

TouchKit driver user guide for Linux

The TouchKit driver package supplies two pre-compiled X modules and a utility for X window. **This driver supports RS232, PS/2 and USB TouchKit controllers only.** All of versions support RS232 and USB TouchKit controllers. **The version of X module later than 1.06 supports PS/2 TouchKit controller.** About PS/2 TouchKit controller support, maybe it needs to rebuild kernel or specific kernel module “serio_raw”. For details about PS/2 TouchKit controller support, please refer to another document **“How to rebuild kernel”**.

1. Installation of the Xorg module:

Regarding installation of Xorg module, all users must copy the Xorg module relating to your version of X window to the X window input modules directory and then configure the Xorg configuration file.

a.) copy the Xorg module

There are two pre-compiled Xorg modules for X window as follows:

- (1) For versions of X window **prior to 6.9 use the `egalax_drv.o` file.**
- (2) For versions of X window **from 6.9 and upwards use the `egalax_drv.so` file.**

User can use **“X -version”** command to check your running version of X window.

The X window input modules directory varies by distribution. Some examples are

`/usr/X11R6/lib/modules/input` (for 64bit: `/usr/X11R6/lib64/modules/input`)
or `/usr/lib/xorg/modules/input` (for 64bit: `/usr/lib64/xorg/modules/input`)

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

```
find / -name mouse_drv.so
```

Output:

```
/usr/lib/xorg/modules/input/mouse_drv.so
```

Copy the correct X module to the correct X window input modules directory.

For example:

```
cp egalax_drv.o /usr/X11R6/lib/modules/input (For X version prior to 6.9)  
or cp egalax_drv.so /usr/X11R6/lib/modules/input (For X version 6.9)  
or cp egalax_drv.so /usr/lib/xorg/modules/input (For X version 7.0 and upwards)
```

b.) configure the xorg configuration file

Edit the xorg configuration file (e.g. `/etc/X11/xorg.conf`) 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, please add multiple InputDevice declarations in the “ServerLayout” section with different names.*

For example:

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

- (2) Configure the Xorg module configuration for TouchKit device.

For each InputDevice section declared in the “ServerLayout” section **you will need to create additional separate configuration** in the `xorg.conf` file.

For only one USB device in the system:

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

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

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

Section "InputDevice" (For RS232 device)

Identifier "EETI1"
Driver "egalax"
Option "Device" "/dev/ttyS0"
Option "Parameters" "/var/lib/egalax1.cal"
Option "ScreenNo" "0"

EndSection

Section "InputDevice" (For USB device)

Identifier "EETI2"
Driver "egalax"
Option "Device" "usbauto"
Option "Parameters" "/var/lib/egalax2.cal"
Option "ScreenNo" "0"

EndSection

Note: *If more than one TocuhKit controllers (2 or more TouchKit touchscreens) are used for the system, please edit each "Parameters" with different file names.*

Option "Device"

The "Device" option must be assigned so that the driver can read the data from the device port. This "Device" is a char device usually found in /dev.

If this "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.

The driver supports three interfaces, serial RS232, PS/2 and USB.

1.) For serial RS232 interface:

This "Device" should be set to correct serial device port name, e. g. /dev/ttyS0 or /dev/ttyS1. Besides, user must ensure that the I/O address and the IRQ are the same as the BIOS setting, please refer to the following command to check these setting.

For example:

```
setserial /dev/ttyS0 -a
```

User can use "man setserial" command to get more information about "setserial" usage.

2.) For **PS/2** interface:

This "Device" should be set to correct PS/2 auxiliary device port name, e. g. **/dev/serio_raw0**. By default, the PS/2 TouchKit device will be directed to mouse device automatically under kernel 2.6 or later. **User must make sure that the kernel supports PS/2 auxiliary port as a char device like kernel 2.4 does and the using version of X module is later than 1.09.** See another document "**How to rebuild kernel**" for details.

3.) For **USB** interface:

There are three kernel modules support USB TouchKit device.

- (1) Inbuilt HID kernel module: "**usbhid**"
- (2) Inbuilt USB kernel module: "**touchkitusb**" or "**usbtouchscreen**"
- (3) TouchKit USB kernel module: "**tkusb**"

If the version of X module is 1.08 or later and the system has only one USB TouchKit device, the "Device" option can be set to "**usbauto**" so that the X module will attempt to determine the communication device port automatically. For example:

Option "Device" "usbauto**"**

It might be better to manually configure the "Device" declaration by user.

Note: *User can get more information about USB kernel module from two "devices" files, by using following commands:*

```
cat /proc/bus/usb/devices  
or cat /proc/bus/input/devices
```

Part of output:

```
P: Vendor=0eef ProdID=0001 Rev=1.00  
S: Product=USB TouchController  
I: If#= 0 Alt=0 #EPs=1 Cls=03(HID ) Sub=00 Prot=00 Driver=usbhid
```

Part of output:

```
I: Bus=003 Vendor=0eef Product=0001 Version=0210  
N: Name="USB TouchController"  
H: Handlers=mouse2 event4
```

3-1) Use inbuilt HID kernel module:

Linux kernel 2.6 supports HID compliant TouchKit device with inbuilt HID kernel module. If user is working with HID compliant TouchKit device and HID kernel module. Then, there should be a device file for this HID compliant TouchKit device in `/dev` or `/dev/usb`. The user has to identify which `/dev/hiddevX` device represents the HID compliant TouchKit device and set the proper "Device" option.

For example:

Option "Device" `/dev/hiddev0`

Note: The HID compliant TouchKit device should *NOT* work with inbuilt USB kernel module `"touchkitusb"` or `"usbtouchscreen"`. Instead, it should work with `"usbhid"` or TouchKit USB kernel module `"tusb"`.

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

Option "Device" `"hiddev*"`

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

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

3-2) Use inbuilt USB kernel module:

The X module version later than 1.06 supports event device nodes. If the USB TouchKit device finds the working kernel module as `"touchkitusb"` or `"usbtouchscreen"`, there should be an event device file for USB TouchKit device in `/dev/input`, e.g. `/dev/input/event4`.

The user has to identify which event device node represents the USB TouchKit device. Then, set the proper "Device" option.

For example:

Option "Device" `/dev/input/event4`

Note: If the “Device” option is set to event device like *“/dev/input/eventX”*.

User must modify *the mouse setting* in the *xorg.conf* file as well to prevent from the mouse driver read the data from the specified event device node. *Set the “Device” option for mouse to a real device node* like *“/dev/input/mouseX”* instead of default device class *“/dev/input/mice”*. User can use the following command to check which real device node is used for mouse.

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

Part of output:

```
N: Name=“ImPS/2 Generic Wheel Mouse”
```

```
P: Phys=isa0060/serio1/input0
```

```
S: Sysfs=/class/input/input2
```

```
H: Handlers=mouse1 event2
```

Note: If the system *has only one USB TouchKit device with inbuilt kernel module “touchkitusb” or “usbtouchscreen”*, the “Device” option for *X module version 1.06 or later* can be set to

Option “Device” “*event”**

or **Option “Device” “*events*”**

so that *the X module will determine event device node for USB TouchKit device automatically.*

3-3) Use TouchKit USB kernel module:

If vendor provided USB kernel module *“tkusb.ko”* was loaded and the device file *“/dev/tkpanel0”* were created for USB TouchKit device, this “Device” option should be set to *“/dev/tkpanel0”*.

For example:

Option “Device” “*/dev/tkpanel0*”

For details about building the TouchKit USB kernel module *“tkusb.ko”*, user can see another document **“How to build module”**.

Note: *The X module version 1.06 or later supports inbuilt kernel module “touchkitusb” and “usbtouchscreen” for TouchKit USB devices. It is highly recommended to use inbuilt kernel module instead of tkusb, by doing this users do NOT need to compile source code.*

Note: *If user prefers to use “tkusb” kernel module, it is suggested to add “touchkitusb” and “usbtouchscreen” into blacklist in /etc/hotplug to avoid conflicts.*

Option “Parameters”

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/egalax.cal”

Option “ScreenNo”

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”

c.) Restart X window

Restart X window to make sure the X module is loaded. It is enough to logout of X window and log back in.

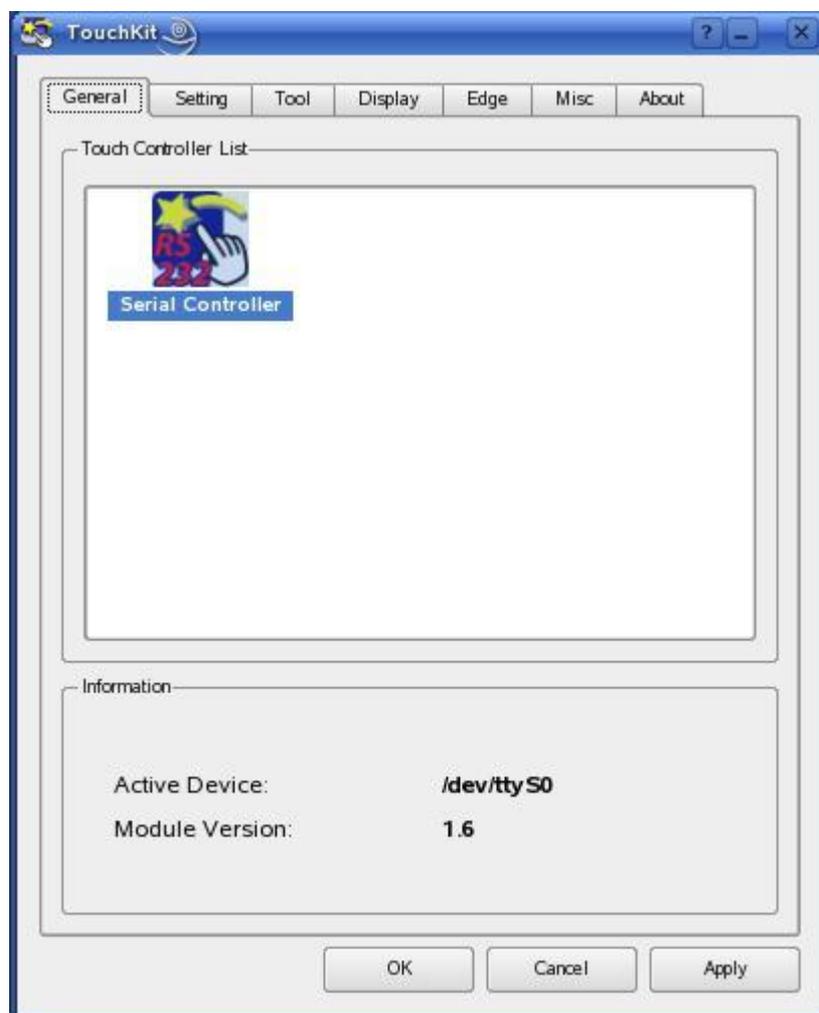
Note: *If the system is running SELinux. SELinux might block access from the driver to the TouchKit device. This can cause many different problems. Before installing the driver issue the “setenforce 0” command to disable SELinux enforcement. Once the driver is installed check your audit logs for denials between Xorg and TouchKit devices. If you see these you will need to create a policy to allow those accesses before re-enabling SELinux with “setenforce 1”.*

2. Utility

TouchKit driver package for Xorg provides user with a configuration tool utility for TouchKit touchscreen. The utility contains property pages **General**, **Setting**, **Tool**, **Display**, **Edge**, **Misc** and **About**.

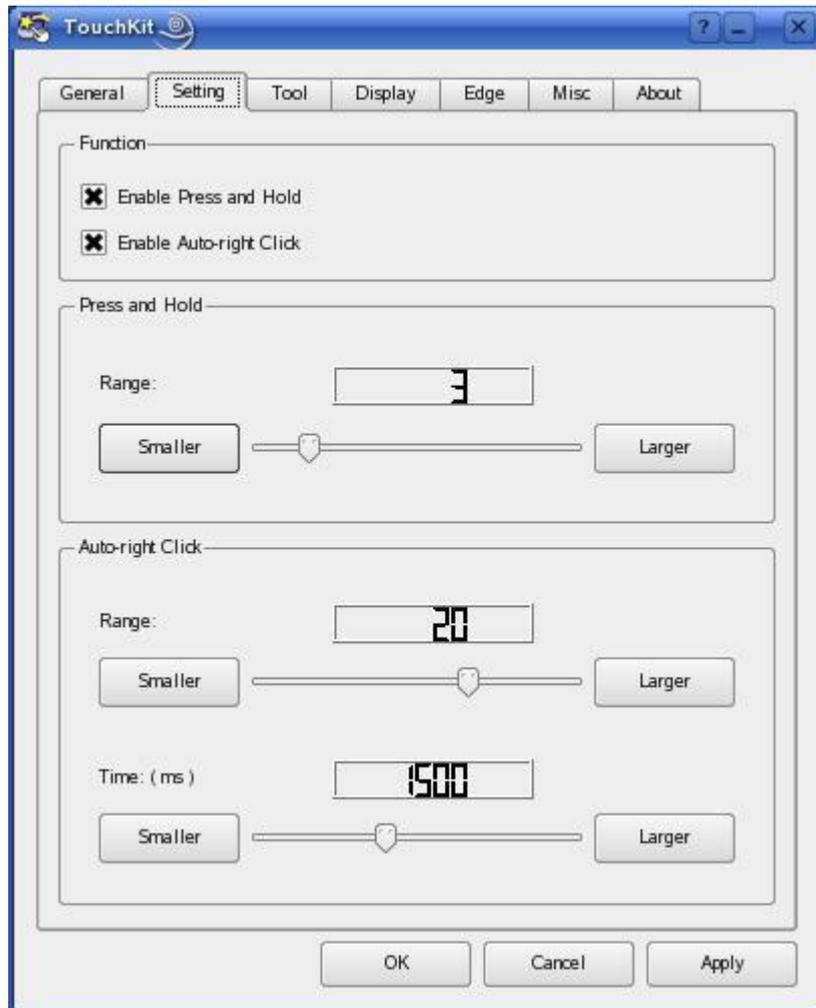
Note: *Make sure the X module is installed correctly and the root permission is required to run this utility. Otherwise, it does not work.*

2.1) General Property:



The utility enumerates TouchKit touchscreen controller installed in this system. **All of the enumerated TouchKit controllers will be list in the "Touch Controller List" Window.** It also shows device name which communication device node the device is connected. In addition, the X module version will be shown in the Information window, too.

2.2) Setting Property:



Some options can be configured for mouse emulation.

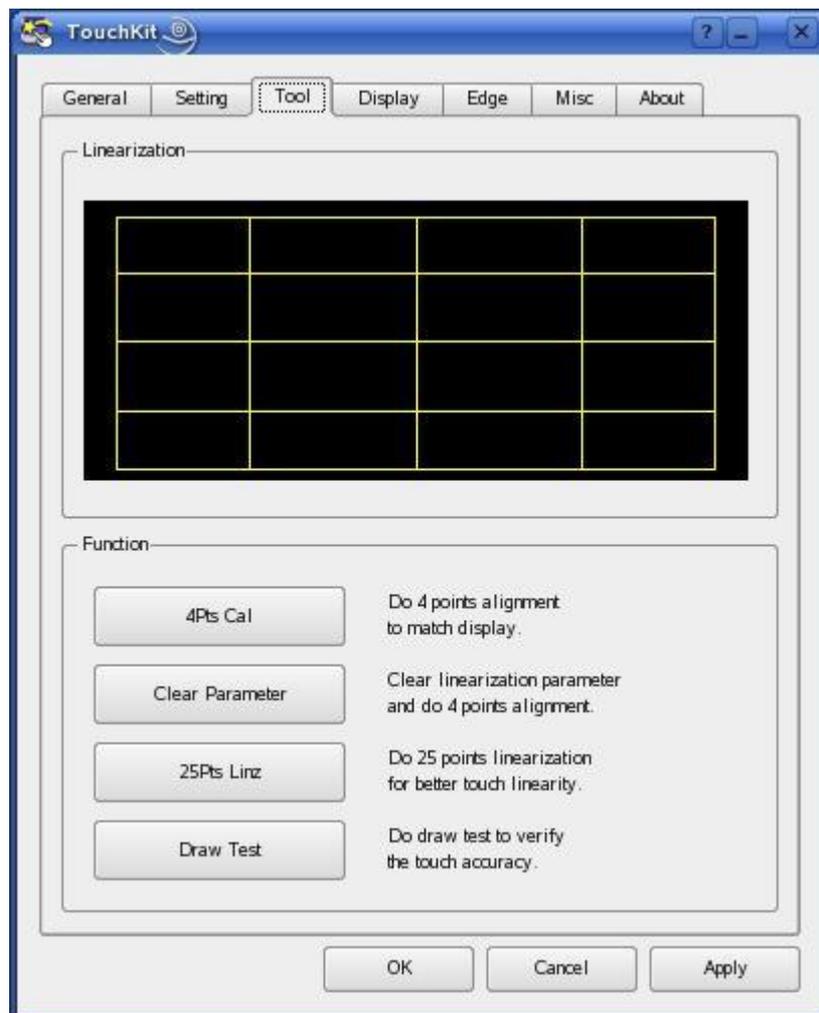
Press and Hold:

Press and hold at the same point. In some application, the application program does not want to receive too many touch points for the touch held at same position, **the user can check this 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 control.

Auto Right Click:

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

2.3) Tool Property:



TouchKit utility provides users with tools for calibration and testing.

Linearization map:

After 25 points calibration, the linearity of the touchscreen will be shown in this linearization map.

4 Pts Cal

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.** User can abort this calibration by pressing **<ESC>** key.

Clear Parameter:

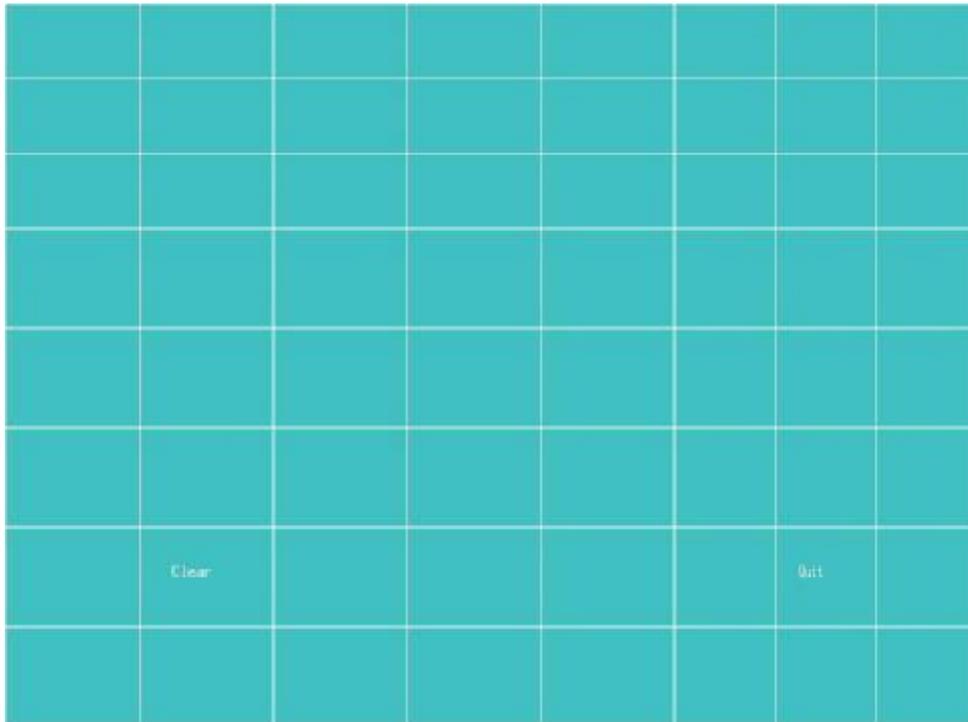
Press this button to **clear the 25 points linearization parameters and do 4 points calibration again.** All of the 25 points linearization parameters will be cleared if the button pressed.

25Pts Linz:

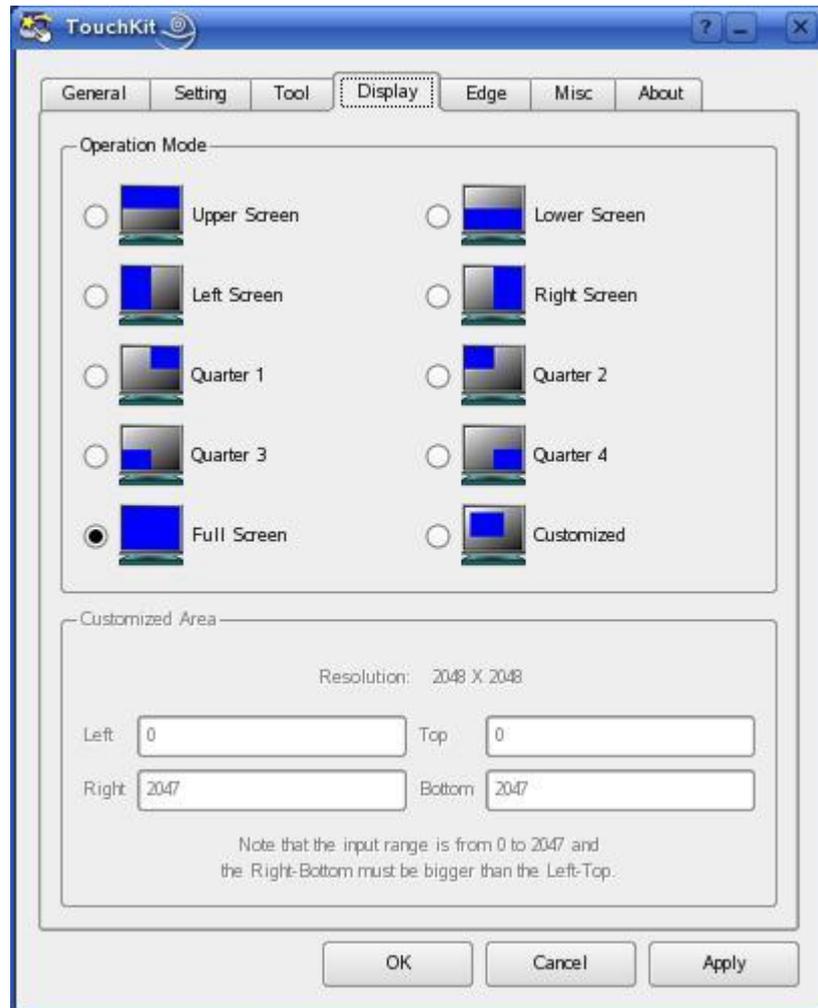
Press this button to do 25 points calibration. **After calibration, the previous 25 points linearization parameters will be overwritten by the new parameters.**

Draw Test:

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



2.4) Display Property:

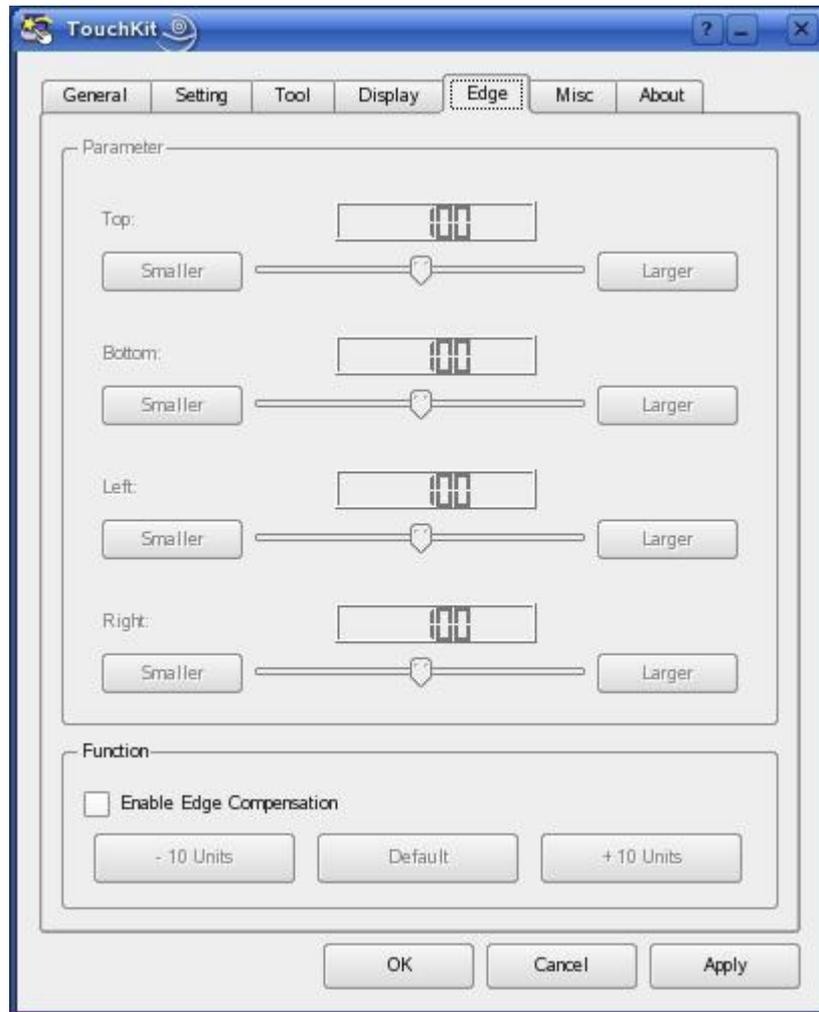


TouchKit utility supports split display feature.

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

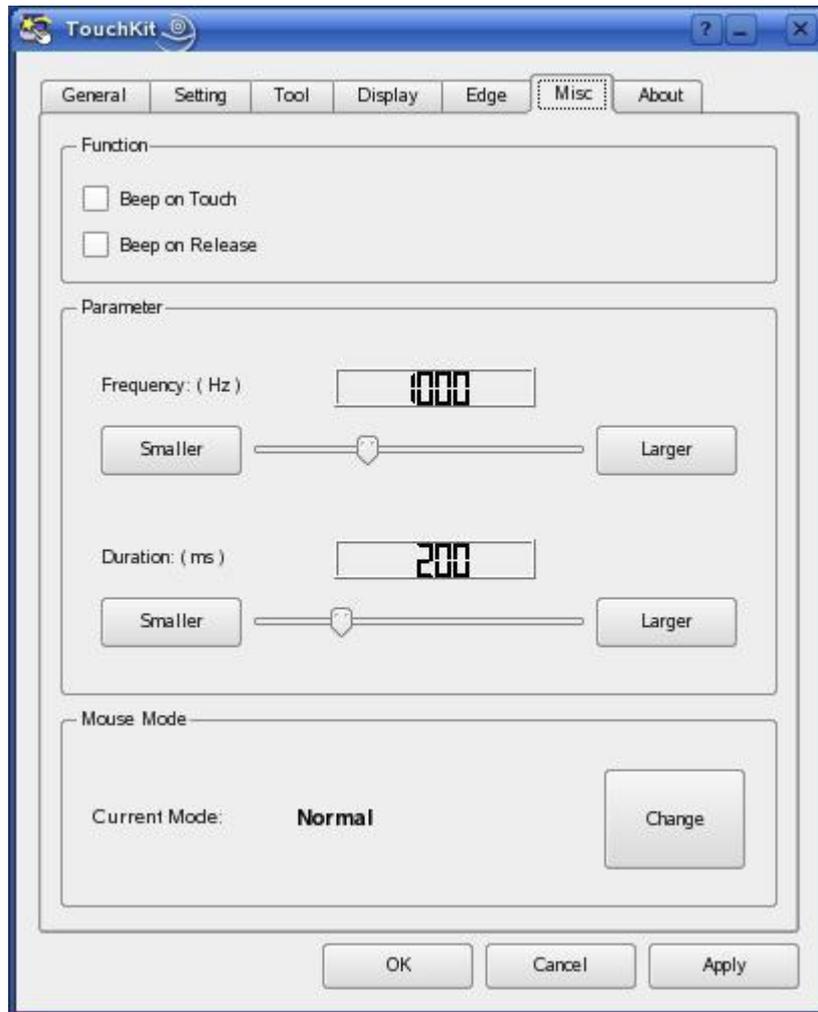
User can choose any options to define where the touchscreen will be mapped. However, if the “**Customized**” is selected, it needs to enter the area to map to. **TouchKit always assume the resolution is 2048 X 2048. If the video resolution is not 2048 X 2048, the user has to calculate the area manually.**

2.5) Edge Property



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

2.6) Misc Property



Beep On Touch:

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

Beep On Release:

Which this function enabled, 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 beep sound frequency.

Duration:

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

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.

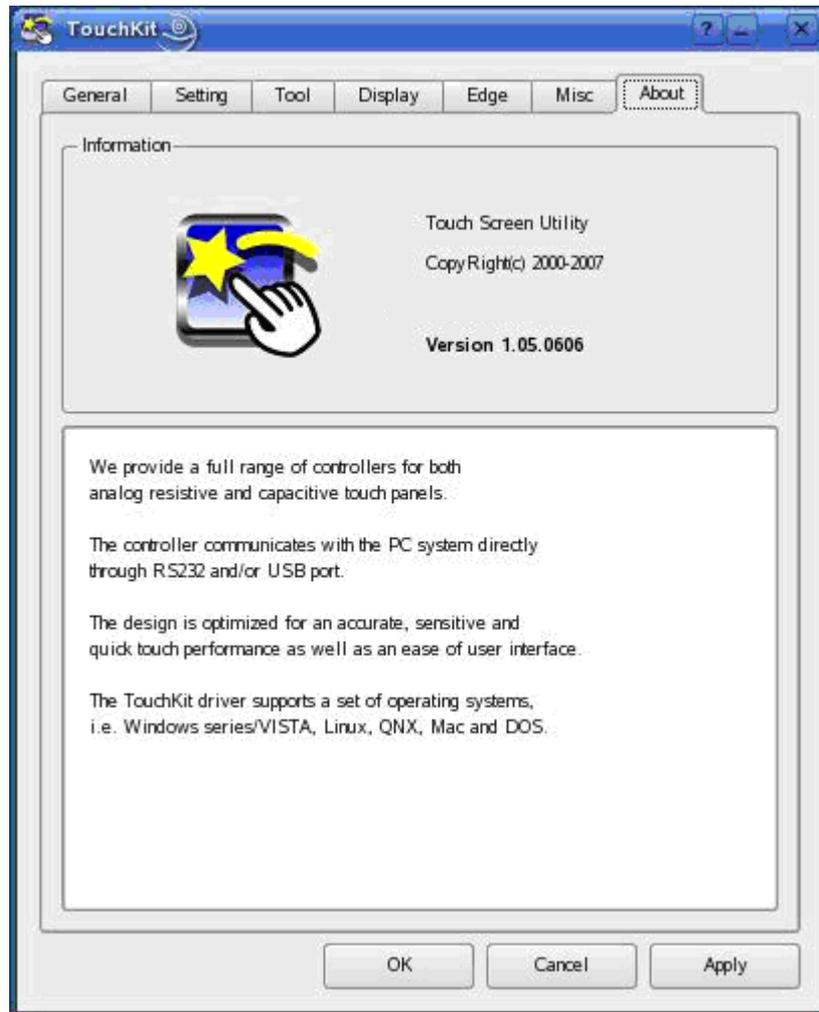
2.) Click On Touch:

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 when it detects a lift off.

2.7) About Property



Information about TouchKit.