# Guts 1.0

Generic Unified Touch Screen support software
July 2004

**by Alessandro Rubini** (rubini@linux.it)

# 1 General Information

The package supports serial and USB touch panels manufactured by *A-Touch* and *Onetouch*, under GNU/Linux and XFree86. The latest version of this driver can be downloaded from:

`ftp://ftp.gnudd.com/pub/guts`

`http://www.gnudd.com/software/#guts`

The latest source tree can be accessed from my CVS server, instructions are available at `http://www.gnudd.com/software/#cvs` . The package name to use is `guts`. As an alternative, you can use *viewcvs* at `http://cvs.gnudd.com`. The latest CVS snapshot can also be downloaded from `ftp://ftp.gnudd.com/pub/guts` or the mirror site `ftp://ftp.linux.it/pub/People/rubini/guts` .

# 2 Device Support

The driver has been designed to run under XFree86 version 4.x. It has been tested on 4.1 and 4.2.

In order to use the touch screen in X, you should install in your system the "guts_drv" module, part of this distribution.

The compiled module ('`guts_drv.o`') should be copied in the module directory of your X server, usually or '`/usr/X11R6/lib/modules/input`'. When the file is in place, a proper '`XF86Config`' will arrange for its loading.

In order to recompile the module you need access to the complete X source tree, and you can compiling issuing:

```
make XFREE_SOURCE=<location-of-source> guts_drv.o
```

The "location of source" above should be the full path name to the directory called "xc" after you uncompress the source tar file.

`XFREE_SOURCE` can be defined in your environment if you prefer. Note that if the variable is defined, calling *make* without arguments will compile the module.

To compile *guts* you'll most likely need to compile X first (by issuing `make World` and waiting a huge lot of time). A freshly uncompressed source tree lacks the proper header files for compilation to succeed; since one of the errors happens inside an X header, you can only fix it by compiling X first – I can't fix things in my source.

# 3 XFree Configuration

## 3.1 XFree 4.x

In order to use a Guts device with XFree 4, you need to add an `InputDevice` section to your '`XF86Config`'. The following example shows how it looks like. The options that are shown commented are not implemented in this version of the module, and are ignored if specified.

```
Section "InputDevice"
Identifier "Touchscreen0"
Driver "guts"
Option "Device"            "/dev/ttyS0"
Option "BaudRate"          "9600"
#Option "Protocol"         "a-touch"
#Option "CalibrationFile"  "/etc/guts.calib"
Option "Smoothness"        "0"
Option "TappingDelay"      "0"
Option "JitterDelay"       "50"
Option "DebugLevel"        "0"
Option "SendCoreEvents"
EndSection
```

Moreover, you need to add an `InputDevice` line in the `ServerLayout` section. After the addition, the section will most likely look like this:

```
Section "ServerLayout"
Identifier  "Simple Layout"
Screen      "Screen 1"
InputDevice "Mouse1" "CorePointer"
InputDevice "Keyboard1" "CoreKeyboard"
InputDevice "Touchscreen0"
EndSection
```

Note that if your system has no mouse device, you can remove the `InputDevice "Mouse1"` line and add `"CorePointer"` to the touch screen line.

The meaning of individual options is as follows:

`Identifier "Touchscreen0"`
: The identifier string is mandatory, and appears, literally, in the `InputDevice` directive within the `ServerLayout` section that is usually found at the end of the configuration file.

`Driver "guts"`
: The name of the driver is mandatory. It is used to load the associated module. In this case the file '`guts_drv.o`'.

`Option "Device" "/dev/ttyS0"`
: The device name is mandatory. It states where input data is collected from. For USB, it will be `/dev/ttyUSB0` or similar.

`Option "BaudRate" "9600"`
: The transmission rate for serial ports. It defaults to 9600. You should not specify

`Option "CalibrationFile" "/etc/guts.calib"`
: The option is not currently implemented, the calibration filename is hardwired

`Option "Smoothness" "0"`
: The smoothness of the pointer. The default value is 0. The greater the value the smoother the motion. While I found it useful when testing the hardware without a real screen underneath, I now default to no smoothness as it's the correct setting when you click buttons directly on the screen.

`Option "TappingDelay" "0"`
: Optional selection of tapping mode. By default (tapping delay set to 0), any touch event is reported as a button press. In tapping mode the first touch event is used to move the pointer, and button press is only reported when the user taps on the device. If the delay, measured in milliseconds, between pen-up and pen-down is greater than the selected value, the pen-down even is considered motion. I personally prefer tapping mode when testing the device, as it allows me to use the common point-and-click semantics instead of click-only; this is especially true if your touchscreen is not placed over the image but in side of it (as you guess, I do that).

`Option "JitterDelay" "50"`
: The debouncing time lapse, measured in milliseconds. If a pen-down event occurs immediately after pen-up (within this time lapse), then both events are discarded. Debouncing has been contributed by Chris Howe <chris@howeville.com>, and defaults to 50ms.

`Option "DebugLevel" "0"`
: The level of messages spit out by the driver. The directive is optional and it defaults to 0.

`Option "SendCoreEvents"`
: The directive instructs X to use the touch screen as a core input device (like the main mouse). This module can currently only work as a core device.

## 3.2 Generic X Configuration

Please note that with XFree you can still use a normal mouse together with the touch panel. Also, can use the touch screen as its only pointer device.

If, when calling `startx`, it fails with a message of `Invalid Subsection Name`, please check the previous error lines, as they explain what is wrong. They usually look like:

```
(--) no ModulePath specified using default: /usr/X11R6/lib/modules
xf86Guts.so: Unknown error loading module
```

The messages are pretty clear, if your are careful in reading them (I am not that careful, and lose half an hour in trying to figure out what was wrong).

# 4 Calibration

To calibrate the touch panel, run the '`guts_calib`' script, under X.

The program needs to find '`guts_control`' and '`guts_to_ascii`' in the command search path or the current directory. This usually means you can simply "`make install`" before running the calibration, but you can also run the calibration program from the source directory, provided "`.`" is in your `PATH`.

The control program is used to turn off the touchscreen in X, so the calibrator can read input data (see Section 5.1 [guts_control], page 4). The conversion tool is used so that the calibrator can read ASCII data instead of binary data (see Section 5.3 [guts_to_ascii], page 5).

The new 5-point calibration, as of release 1.2 of the package, is a full-screen application that takes complete control of your desktop. Since it disables the touchscreen in X in order to directly read the serial port, you'll need to invoke "`guts_control on`" or "`guts_control raw`" if the program terminates unexpectedly. Premature death of the application is not foreseeable, but you may want to send a termination signal to the calibrator for whatever reason. See Section 5.1 [guts_control], page 4.

# 5 Support Tools

The package includes three tools that work by connecting with the X server:

## 5.1 guts_control

The program receives a single command-line argument and uses it as a command to be performed by communicating with the X server.

The following commands are supported:

`devinfo`

Report device information to standard output. The information is about the input devices active in the current X server. In order for the other commands to succeed, the server must have exactly one touch panel configured. See Chapter 3 [XFree Configuration], page 1.

`off`

Deactivate the input device. The command makes the driver stop reading the serial port, so the calibration program (or otherwise) can read touchscreen data.

`on`

Activate the input device. The command tells the driver to start reading the serial port again and activate the current calibration. If no calibration file is found the touchscreen will work in raw mode.

`raw`

Activate the input device in raw mode. Even if a calibration file exists, it will not be used until the *on* command is issued. The command is meant to stop using wrong calibration information that sometimes may be generated by error.

`b1`

Generate button-1 press and release events (the default).

`b2`
`b3`

Generate button-2 (middle button) events from now on, likewise for button-3 (right button).

`b2once`
`b3once`

Generate a button-2 event for the next press and release events, and then turn back to reporting button-1 events. The program won't terminate until the button-2 events have been reported. Likewise for button-3.

`getleds`

Return to standard output information about the current status of device leds. There are no physical leds on the touchscreen device, but the "led" abstraction is what has been used to send commands to the driver. This command is useful to me but very unlikely to be useful to anyone else.

## 5.2 guts_panel

The program is a simple graphical application to control generation of button-2 and button-3 mouse events. Running the application is optional, and you won't need to run it if your setup only uses button-1 events.

The program window is made up of two buttons, labelled "2" and "3". They are inactive by default.

After clicking on one of the buttons, it becomes yellow and the touch screen will report a button-2 or button-3 event once. After the event is reported the button turns back gray and the touchscreen will go back to report button-1 events.

After clicking twice on one of the buttons, the button becomes red and button-2 or button-3 events will be generated from now on. To go back to button-1 events just click on the button again to inactivate it.

## 5.3 guts_to_ascii

The program reads from the serial port specified on the command-line and converts binary packet to ascii information. Besides being used by the calibration program, the program can be used for basic diagnosis of hardware problems.

## 5.4 warp

The program moves the mouse pointer. It can be used to move the mouse when there is no active mouse. It accepts a series of letters on the command line: each uppercase N, S, W, E moves the pointer by 100 pixels to the north, south, west, east. Each lowercase n, s, w, e moves the pointer by 10 pixels.

# Table of Contents


**1 General Information . . . . . 1**

**2 Device Support . . . . . 1**

**3 XFree Configuration . . . . . 1**

3.1 XFree 4.x . . . . . 2
3.2 Generic X Configuration . . . . . 3

**4 Calibration . . . . . 4**

**5 Support Tools . . . . . 4**

5.1 guts_control . . . . . 4
5.2 guts_panel . . . . . 5
5.3 guts_to_ascii . . . . . 5
5.4 warp . . . . . 5