XP Universal Driver User Manual

Elo Device Interface Functions

Contacting Elo

Americas

Asia-Pacific

Europe (including Africa/Middle East)

P/N008579
Revision A

Introduction

The Elo TouchSystems XP Universal Driver package contains:

- Native 32-bit drivers, for both serial and USB touchscreen controllers, designed for Microsoft Windows XP and Windows 2000
- Setup application
- Video Alignment application, sometimes known as a "Calibration" application
- Control Panel application
- Tool Tray Application
- Right Button Application
- Edge Acceleration Application
- Center Desktop Application

Additional features for this driver, in the form of companion application programs, may be obtained by contacting Elo's Applications Engineering Department (see Contacting Elo).

While this driver and Users Manual are written for both Windows XP and Windows 2000, differences in nomenclature between the two operating systems do exist. This manual uses the nomenclature and descriptions for Windows XP in cases where Windows XP and Windows 2000 differ.

All Elo touchscreen controllers that utilize Elo's SMARTSET command set and data protocol are supported by this driver, with the exception of Elo's older ISA Bus controllers. Supported controllers, and touchmonitors which contain these controllers, currently include:

- All Elo Entuitive brand touchmonitors with an internal USB or serial controller
- All Touch Panel Systems (TPS) touchmonitors with an internal or external serial or USB controller
- Special products from Elo Rochester containing one of the above controllers
- Elo USB Controllers, AccuTouch 3000U, CarrollTouch 4000U, 4500U, and IntelliTouch 2500U, 2600
- Elo Serial Controllers AccuTouch 2210, 2215, CarrollTouch 4000S, IntelliTouch 2500S, 2310B, 2310, 2300
- Monitors from third party vendors containing one of the above controllers

Installing the Driver

Hardware

Touchscreen controllers must be configured for the driver prior to installation. Elo typically ships controllers in a default setup that is compatible with this driver. Controller requirements are:

Serial

- Serial controllers must conform to these minimum requirements:
  - Baud rate-9600 Baud.
  - Data format-SMARTSET
- Non-Elo controllers must support the SMARTSET "T" command, that is, they must send an Elo SMARTSET Touch packet to function with this driver.
  - It is desirable for a full handshaking connection to be established between the touchscreen controller and the computer, but it is not required. The controller will function with this driver in a "two-wire" (RxD and Gnd) configuration. The consequences of a two-wire connection are:
  - Automatic detection of the controllers on serial ports during setup is not possible
  - Controller information will not be registered in the Property Page of the Control Panel

USB

- USB controllers require no configuration prior to installation.
- Elo's USB controllers are "HID" (Human Interface Device) compliant. The Windows 2000 and Windows XP operating systems have native HID drivers that provide low-level support for Elo touchscreen controllers. An Elo touchmonitor, with a USB touchscreen controller installed, will provide the following limited operation with these native HID drivers. Operation of the controller will be limitied without the installation of the software discussed in the next section:
  - The mouse cursor will move on the display in response to touch
  - The direction of motion on the display will depend on the orientation of the touchscreen and the defaults programmed into the controller and the driver
  - No "beep" will be heard when the screen is touched
  - No Control Panel application is available to configure the operation of the driver.

Software

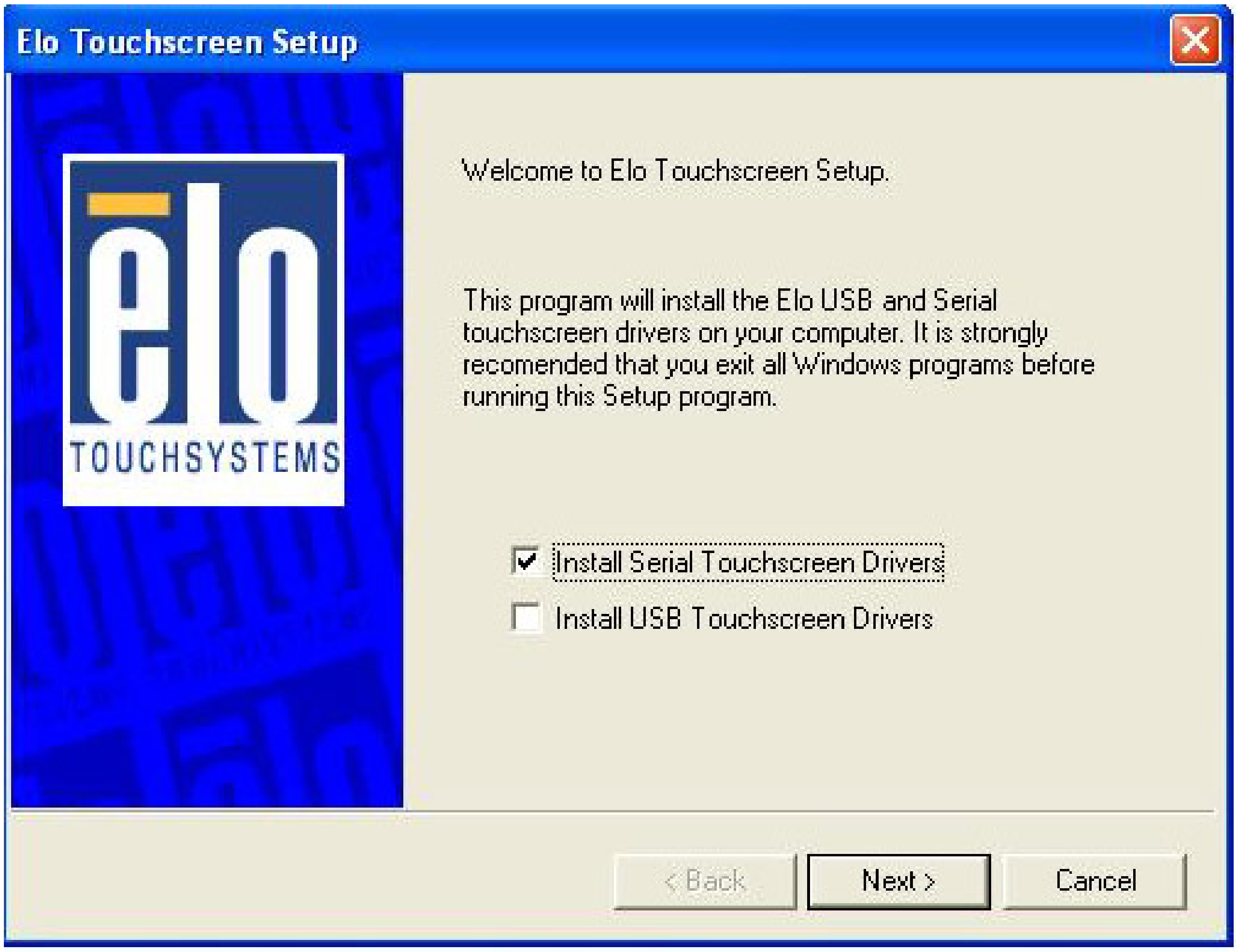
There are many options to consider when installing the driver files. This section will explain the most common ways that the driver is installed. Consult Elo Applications Engineering for situations not covered in this section.

- **Setup from Windows Desktop**

  - Single monitor, serial controller
    - Run the self-extracting zip file (SW500849.exe) to unzip the files, using the Run EloSetup

option

- EloSetup will launch. Check the "Install Serial Touchscreen Drivers" box.


Elo Touchscreen Setup

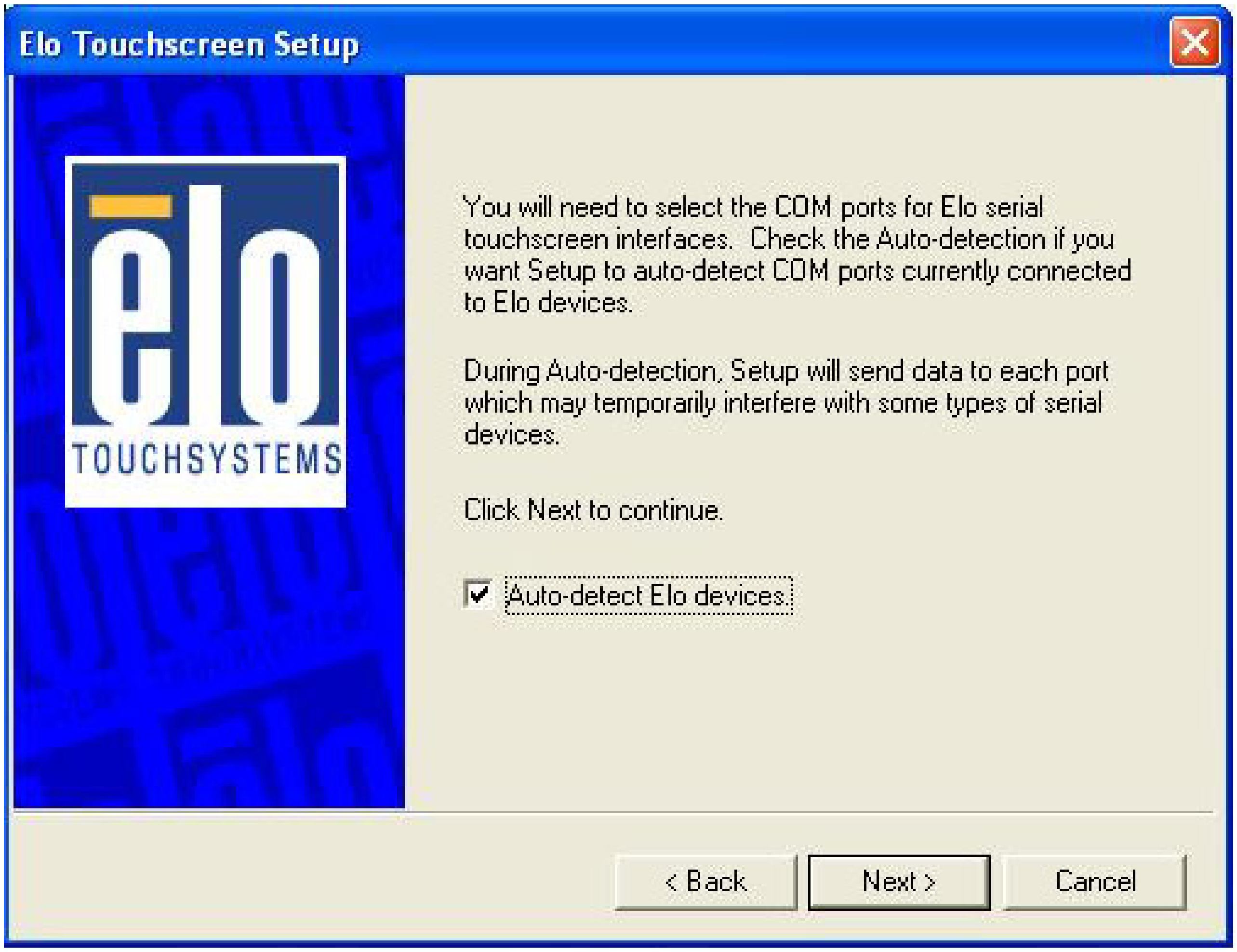
Welcome to Elo Touchscreen Setup.

This program will install the Elo USB and Serial touchscreen drivers on your computer. It is strongly recomended that you exit all Windows programs before running this Setup program.

[✓] Install Serial Touchscreen Drivers

[ ] Install USB Touchscreen Drivers

- Review and accept the License Agreement
- Check "Auto-detect Elo devices" if your touchmonitor serial cable is connected to the computer and the serial cable suppoerts hardware handshaking. If you connect the serial cable later, do not check the "auto-detect" box. Go to the next screen.



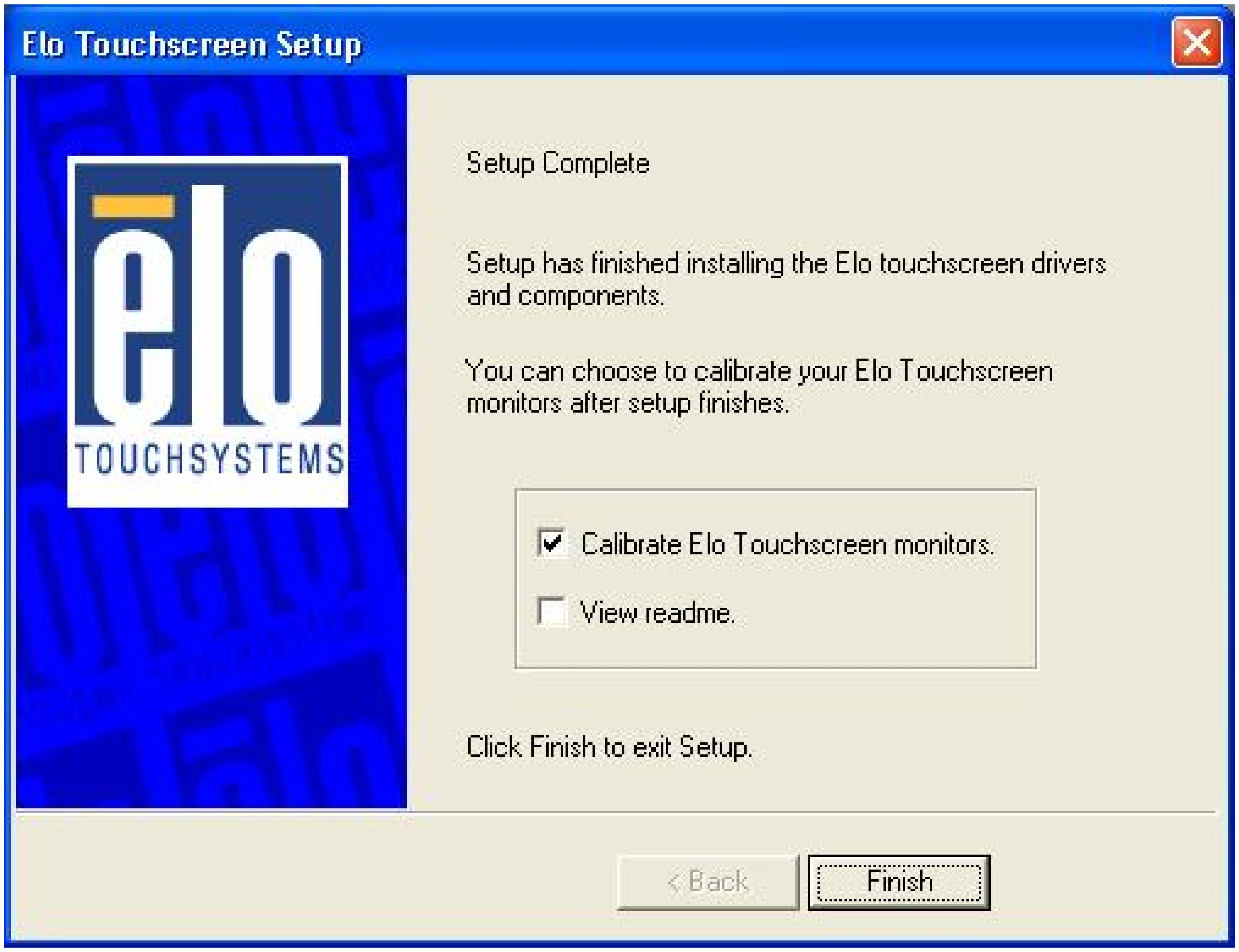
You will need to select the COM ports for Elo serial touchscreen interfaces. Check the Auto-detection if you want Setup to auto-detect COM ports currently connected to Elo devices.

During Auto-detection, Setup will send data to each port which may temporarily interfere with some types of serial devices.

Click Next to continue.

[✓] Auto-detect Elo devices.

- A list of all the serial ports that the system detected is shown. If you checked the auto-detect box in the previous screen, and your touchmonitor serial cable is attached to the computer, the COM port that you selected should be checked in the list. If you agree with the selection, click Next and proceed to the next screen.
- The COM port you selected will be shown in the window of this screen. This is your last opportunity to change your COM port selection before the driver files are installed. Use the Back button(s) to change any of the selections you previously made. Click "Next" to complete the installation of the driver files.
- Depending on the revision of the driver you are installing, Windows may advise you that the "software you are installing has not passed Windows logo testing…" . Select "Continue Anyway" to proceed with the installation. This message can be suppressed in future installations by changing the selection in "Driver Signing Options" at Control Panel>System>Hardware>Device Manager>Driver Signing.
- When the "Setup Complete" screen appears, you may choose to run calibration (the Elo Video Alignment program EloVA) immediately or wait until later. If you choose not to run this program now, you can run it from the Elo Control Panel application.


Elo Touchscreen Setup

Setup Complete

Setup has finished installing the Elo touchscreen drivers and components.

You can choose to calibrate your Elo Touchscreen monitors after setup finishes.

- [✓] Calibrate Elo Touchscreen monitors.
- [ ] View readme.

Click Finish to exit Setup.

- Single monitor, USB controller
  - Run the self-extracting zip file (SW500849.exe) to unzip the files, using the Run EloSetup option.
  - EloSetup will launch. Check the "Install USB Touchscreen Drivers" box.
  - Review and accept the License Agreement.
  - The driver files will install. When the "Setup Complete" screen appears, you may choose to run calibration (the Elo Video Alignment program EloVA) immediately or wait until later. If you choose not to run this program now, you can run it from the Elo Control Panel application.

- Multiple monitors, serial controllers

  - Follow the general procedure for **Single monitor, serial controller** installation, above, auto detecting or selecting all the serial ports that you will use.
  - The driver files will install. When the "Setup Complete" screen appears, you may choose to run calibration (the Elo Video Alignment program EloVA) immediately or wait until later. If you choose not to run this program now, you can run it from the Elo Control Panel application. When EloVA runs for the first time, it will attempt to calibrate all controllers and/or serial ports that were installed. You can use the keyboard Esc key to terminate or skip calibration for any monitor, or allow the program to time out (as indicated by the progress bar reaching maximum).

Multiple monitors, USB controllers

- Follow the procedure for **Single monitor, USB controller** installation above.
- The driver files will install. When the "Setup Complete" screen appears, you may choose to run calibration (the Elo Video Alignment program EloVA) immediately or wait until later. If you choose not to run this program now, you can run it from the Elo Control Panel application. When EloVA runs for the first time, it will attempt to calibrate all controllers and/or serial ports that were installed. You can use the keyboard Esc key to terminate or skip calibration for any monitor, or allow the program to time out (as indicated by the progress bar reaching maximum). The program will continue until all controllers and/or ports have been calibrated.

- Multiple monitors, serial and USB controllers

  - Run the self-extracting zip file (SW500849.exe) to unzip the files, using the Run EloSetup option
  - EloSetup will launch. Check both the "Install Serial Touchscreen Drivers" and "Install USB Touchscreen Drivers" boxes.
  - Review and accept the License Agreement
  - The program will appear to proceed as a single/multiple serial controller installation but USB files will also install.
  - The driver files will install. When the "Setup Complete" screen appears, you may choose to run calibration (the Elo Video Alignment program EloVA) immediately or wait until later. If you choose not to run this program now, you can run it from the Elo Control Panel application. When EloVA runs for the first time, it will attempt to calibrate all controllers and/or serial ports that were installed. You can use the keyboard Esc key to terminate or skip calibration for any monitor, or allow the program to time out (as indicated by the progress bar reaching maximum). The program will continue until all controllers and/or ports have been calibrated.

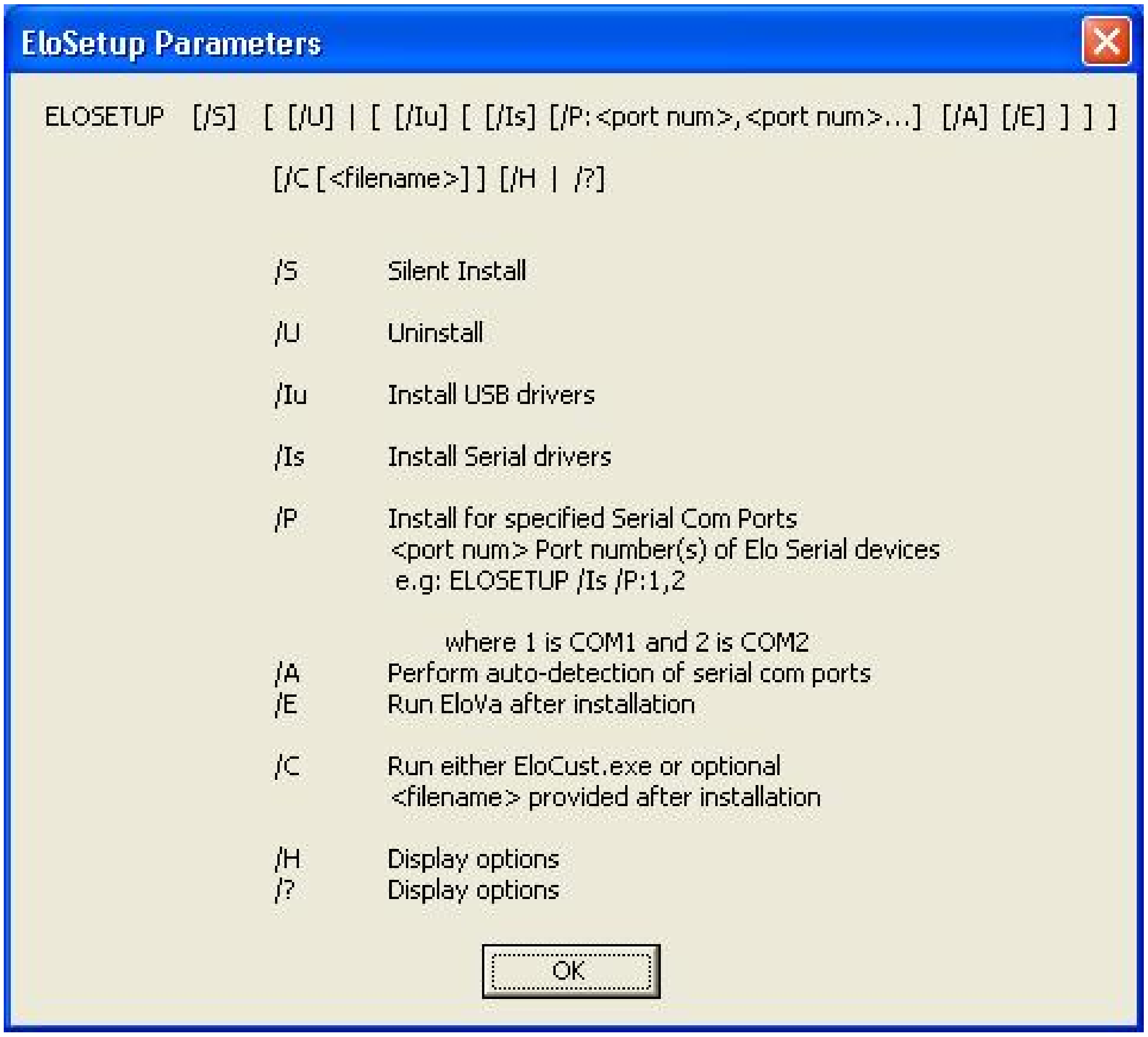
• Silent Install

The EloSetup program for this driver may also be run as an attended or unattended program from a command line, batch file, etc. Example usages:

To Silent install for USB controllers use
EloSetup /Iu /s

To Silent install for a Serial controller on COM1 use
EloSetup /Is /P:1 /s

To view all the options available for this installation method, run EloSetup as **EloSetup /h**.



```
ELOSETUP   [/S]   [ [/U] | [ [/Iu] [ [/Is] [/P:<port num>,<port num>...]  [/A] [/E] ] ] ]
                  [/C [<filename>] ] [/H  |  /?]

                  /S         Silent Install

                  /U         Uninstall

                  /Iu        Install USB drivers

                  /Is        Install Serial drivers

                  /P         Install for specified Serial Com Ports
                             <port num> Port number(s) of Elo Serial devices
                              e.g: ELOSETUP /Is /P:1,2

                                 where 1 is COM1 and 2 is COM2
                  /A         Perform auto-detection of serial com ports
                  /E         Run EloVa after installation

                  /C         Run either EloCust.exe or optional
                             <filename> provided after installation

                  /H         Display options
                  /?         Display options
```

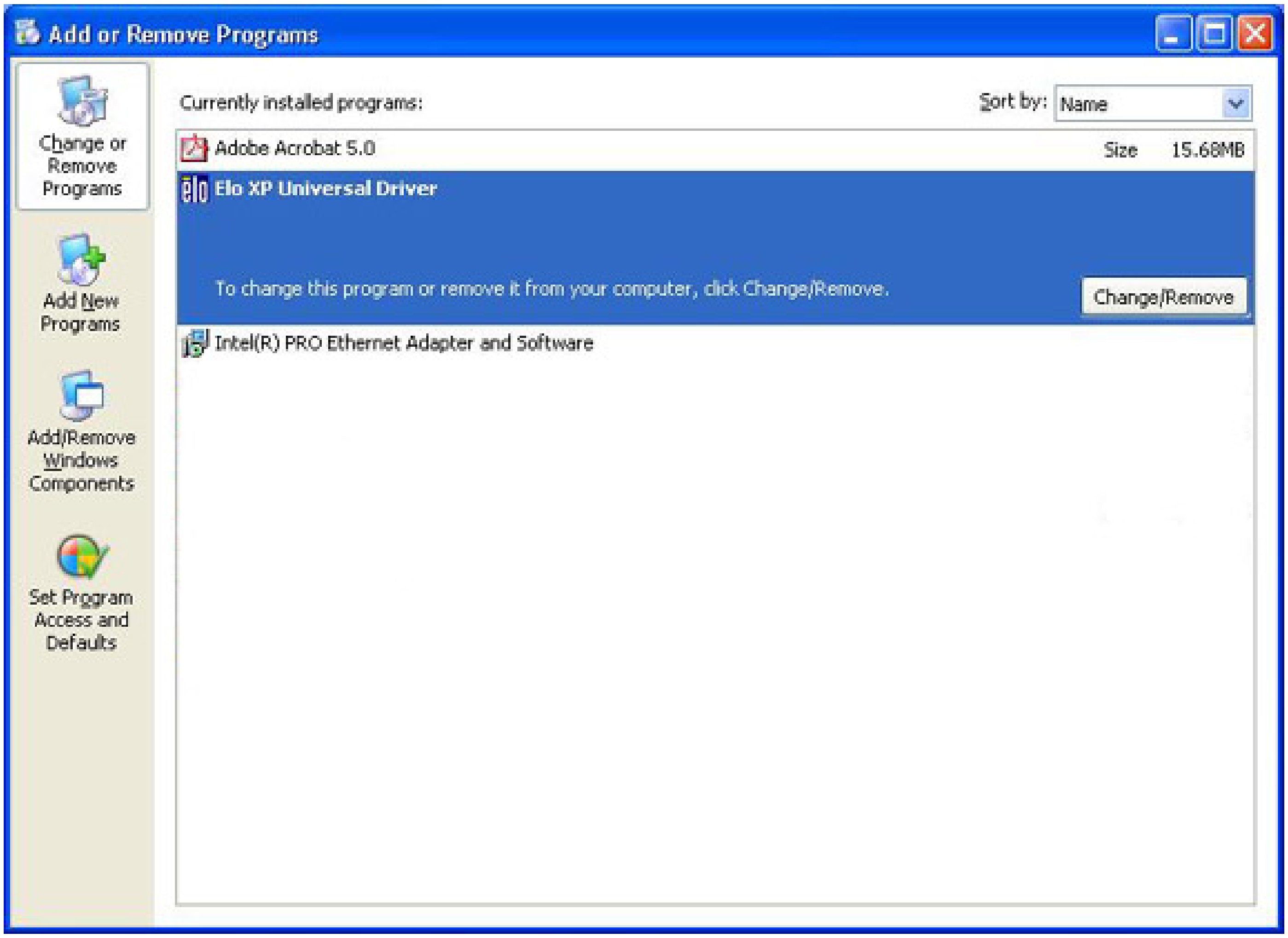
- **Adding Additional Serial Controllers**
  - Run EloSetup again and follow the installation sequence above that matches the type of installation; a single, or multiple serial monitor, that you wish to perform.

- **Adding Additional USB Controllers**
  - If there are no Elo USB devices attached to the system, run install and select both serial and USB check boxes.
  - If there are Elo USB devices already installed, plug the USB cable from the touchmonitor into the computer and run EloVA to calibrate the touchmonitors.

Uninstalling the Driver

Uninstalling the Touch Drivers

- Use the Add/Remove Programs feature in the Control Panel
  - Open the Control Panel
    - Click Start, hover over Settings, click Control Panel
  - Double-click the Add/Remove Programs icon
    - On some machines there will be a moderate wait time for the program list to populate
  - Click the Elo XP Universal Driver item
  - Click the Change/Remove button



Add or Remove Programs
Change or Remove Programs
Add New Programs
Add/Remove Windows Components
Set Program Access and Defaults
Sort by: Name
Change/Remove

Currently installed programs:

Adobe Acrobat 5.0 Size 15.68MB

Elo XP Universal Driver

To change this program or remove it from your computer, click Change/Remove.

Intel(R) PRO Ethernet Adapter and Software

  - Click the Yes button
    - The Elo drivers will be removed. No computer restart is required.
    - Click the Finish button when it appears.
- Run **EloSetup /U** from a command line or batch file. See Installing the Driver, Software, Silent Install for more details.

The following registry keys remain after the driver is uninstalled:

- The Secondary registry folders and keys (see Registry Entries section of this document)

- The USB portion of the Primary registry entries. See the Registry Entries section of this document for the location of the USB registry entries. Only the Vid_04e7&Pid_#### folder applies to Elo registry entries.

To facilitate reinstallation, the file folders remain intact after the driver is uninstalled:

- \Program Files\EloTouchSystems

Disabling the Driver

Disabling Touch Functionality

Touch functionality may be quickly and easily disabled using the **Tool Tray** icon. If the Elo icon is not in the Tool Tray (lower right corner of the display), see the section on enabling the Tool Tray icon, below.

- Click the **Elo icon** in the Tool Tray (lower right corner of the display)
  - The Elo Touchscreen Properties list is displayed
- Click **Disable Touch**
  - Touch functionality is disabled
- Touch functionality is re-enabled in the same manner
  - This is a toggle function (Enable/Disable)

To place the Elo icon in the Tool Tray:

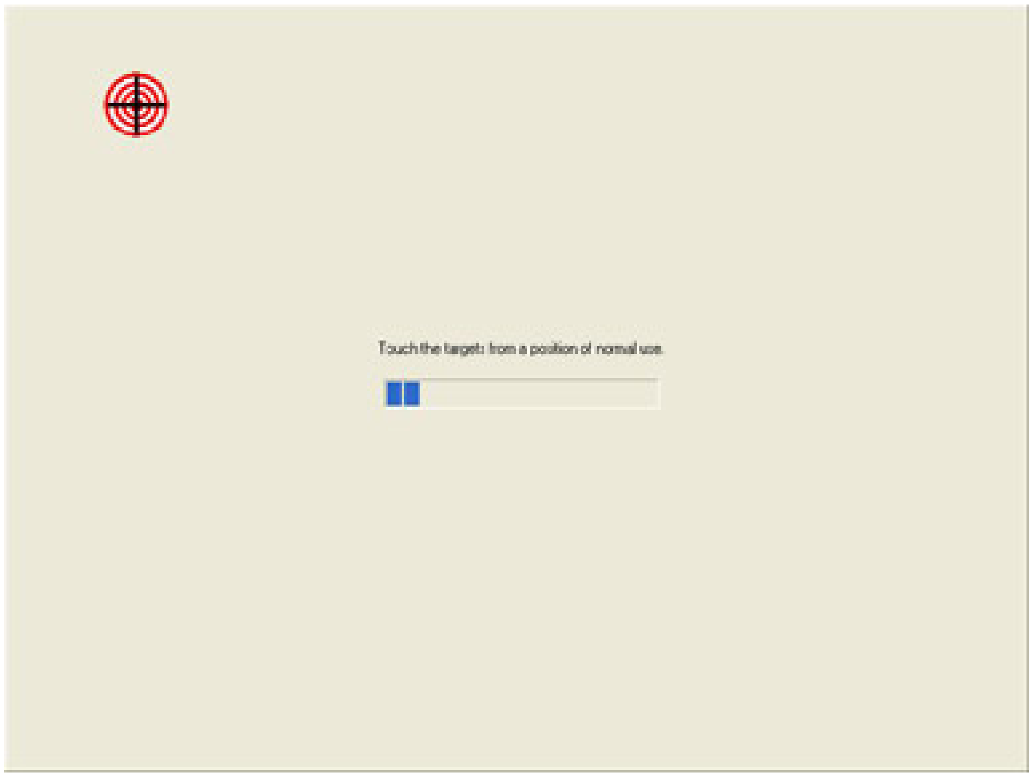
- See Tool Tray section of this document.

Video Alignment (Calibration)

Video alignment or calibration, as it is also called, ensures that the mouse cursor appears at the position of touch.

Elo uses a three-point calibration sequence that will accept touchscreens with any orientation of the X or Y axis, in landscape or portrait mode.

Once calibrated, the touchscreen will be ready to use automatically each time the system is restarted.


Touch the targets from a position of normal use.

Options to Launch

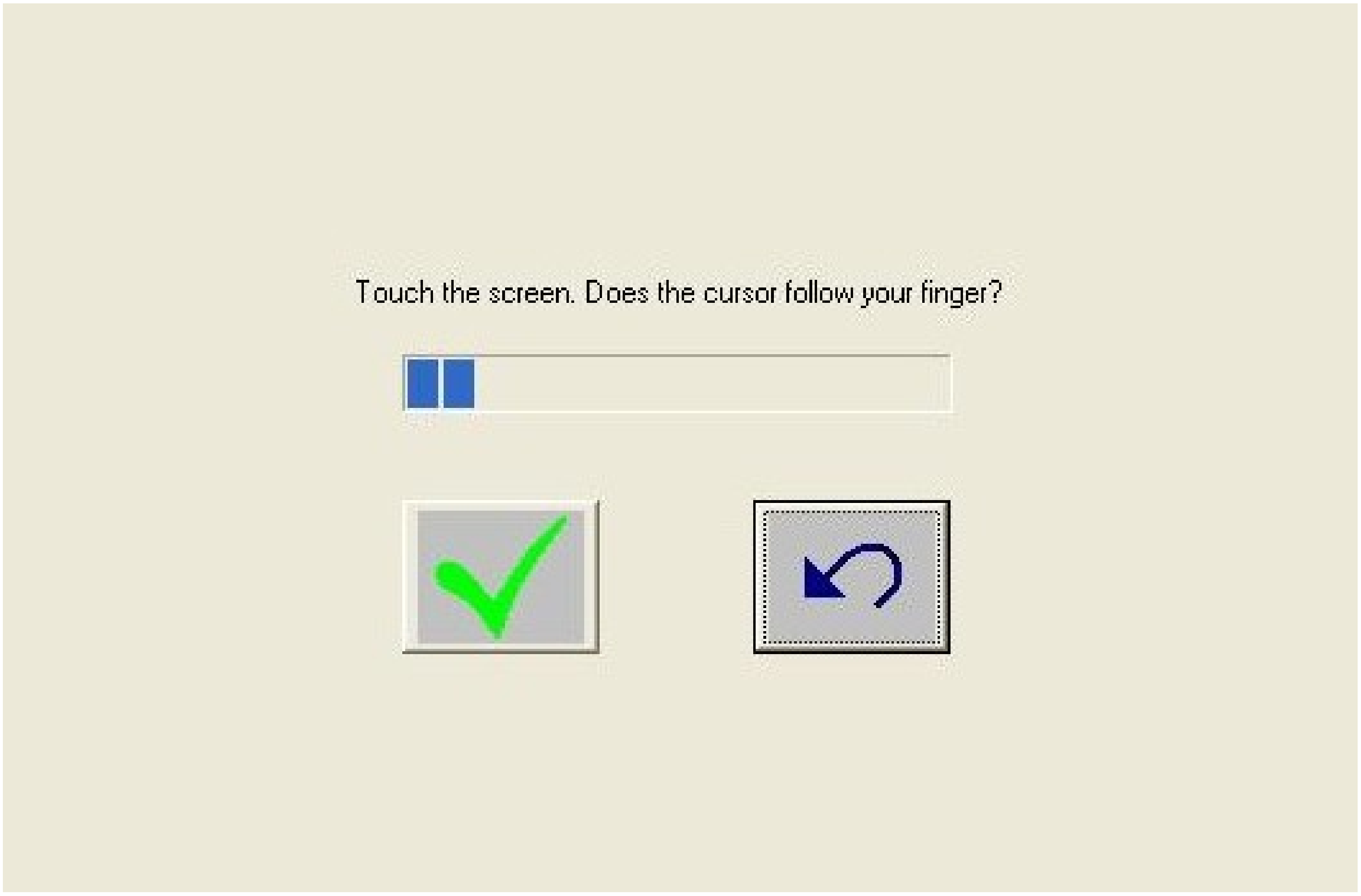
The video alignment program may be launched six different ways:

1. During setup a checkbox may be selected to run the alignment program automatically after setup is completed.
2. Via the Elo Control Panel application by selecting the General tab and clicking the alignment program icon.
3. Via the Elo Control Panel application by selecting the Properties tab for the individual monitor and clicking the alignment program icon.
4. By clicking the Elo icon in the Windows task bar at the bottom right of the display. This will bring up the Elo Control Panel application and the alignment program may be run via option two or three listed above.
5. By running EloVa.exe directly from the Windows command line or \Windows\System32 directory.
6. By calling EloVa.exe directly from an application.

Running the Alignment Program

When you run EloVA using one of the options above, touch the three targets as they appear.

You will then be asked to touch various points on the screen to verify that the cursor appears at the position touched. If the cursor appears at the position touched, click the green arrow. If the cursor does not appear at the position touched, click the blue curved arrow and the alignment program will run again.


Touch the screen. Does the cursor follow your finger?

If you are using multiple monitors the alignment program will run on each individual monitor. If one of the monitors is not a touchmonitor, press the Escape key on the keyboard and the alignment program will advance to the next monitor, or wait until the program times out as indicated by completion of the progress bar.

Landscape/Portrait Mode

EloVA may be run in either landscape or portrait mode.

Unusual Combinations of Expanded Desktop with Multiple Monitors

Unusual multiple monitor combinations may create an expanded Windows desktop that is not recognized by the default video resolutions built into the EloVA program. For this reason, an auxiliary file, **EloVideo.txt**, is available to supplement the defaults. If you are running an expanded desktop multiple monitor resolution that is not multiples of standard monitor resolutions, you may experience difficulties in calibrating all of your monitors properly. Contact Elo Applications Engineering for availability and details of the use of this file.

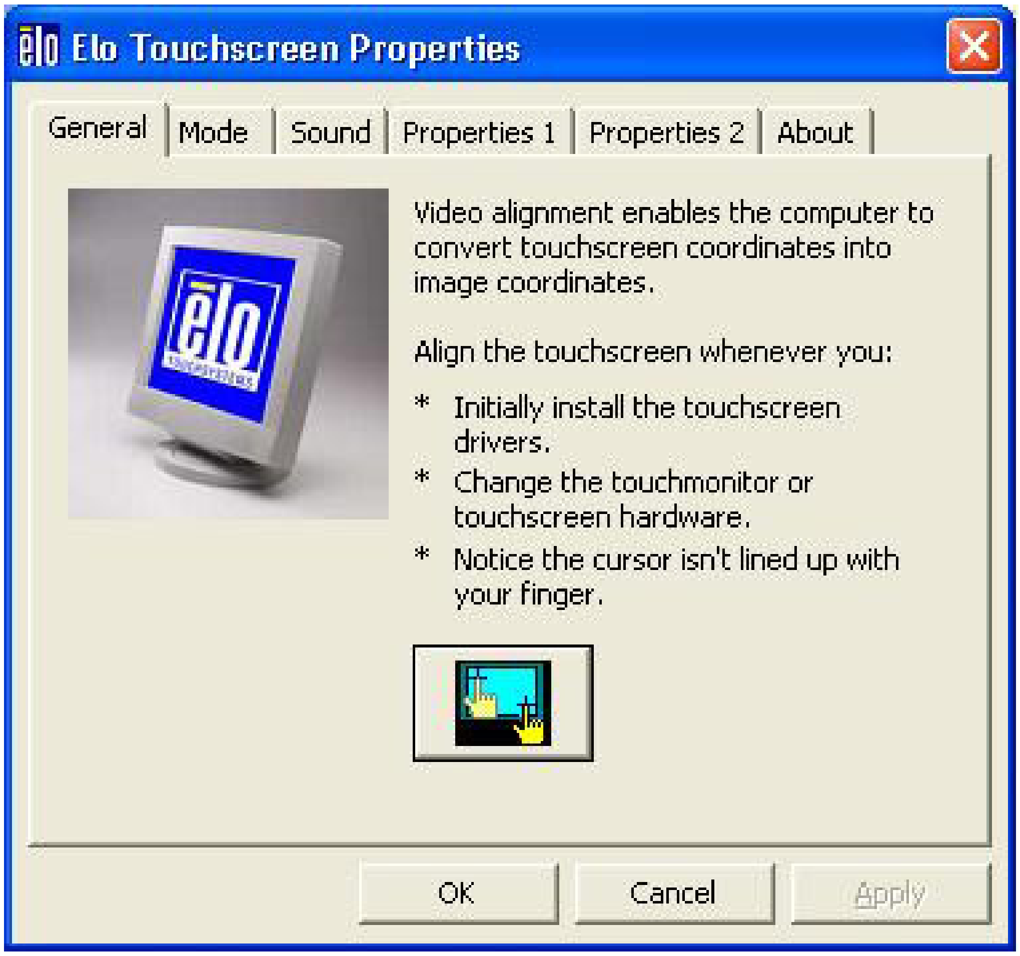
Control Panel (Elo Mouse Properties)

The control panel (CP) application allows configuration of the driver to suit the application programs, and presents system and diagnostic information to the user.

Each of the five or more tabs in the control panel are described below.

General

- The General tab is displayed when the Control Panel is opened. The other tabs described below may be selected, and the video alignment program EloVA may be launched from this tab.



Video alignment enables the computer to convert touchscreen coordinates into image coordinates.

Align the touchscreen whenever you:

* Initially install the touchscreen drivers.

* Change the touchmonitor or touchscreen hardware.

* Notice the cursor isn't lined up with your finger.

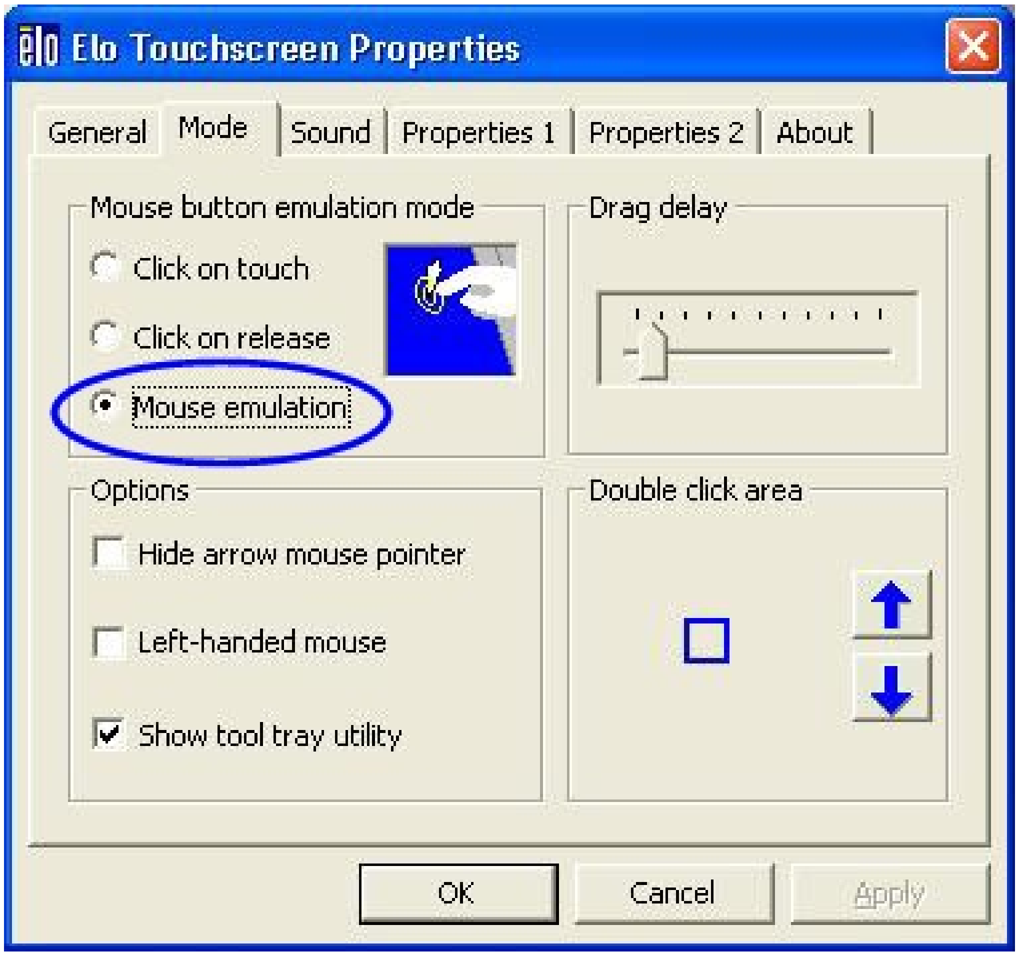
Mode

- The Mode tab selects touchscreen operating mode and configures the appearance and function ability of the desktop. All selections made in the Mode tab must be activated by clicking the "Apply" button at the bottom of the dialog.

•

Mouse Button Emulation Mode

This mode is selected by clicking the appropriate radio button.



Mouse button emulation mode

- ○ Click on touch
- ○ Click on release
- ● Mouse emulation

Options

- [ ] Hide arrow mouse pointer
- [ ] Left-handed mouse
- [x] Show tool tray utility

Click on Touch

- Click on touch sends, immediately upon touch, a mouse down/up message at the point of touch on the touchscreen. The user's finger must be removed from the touchscreen before a new touch at any location will be recognized. The cursor or selected objects CANNOT be "dragged" on the screen in this mode.

Click on Release

- Click on release sends, at the time of release (untouch), a mouse down/up message at the point that the screen was last touched. Dragging across objects on the screen will not highlight or select them unless untouch occurs when the touch is over the object.

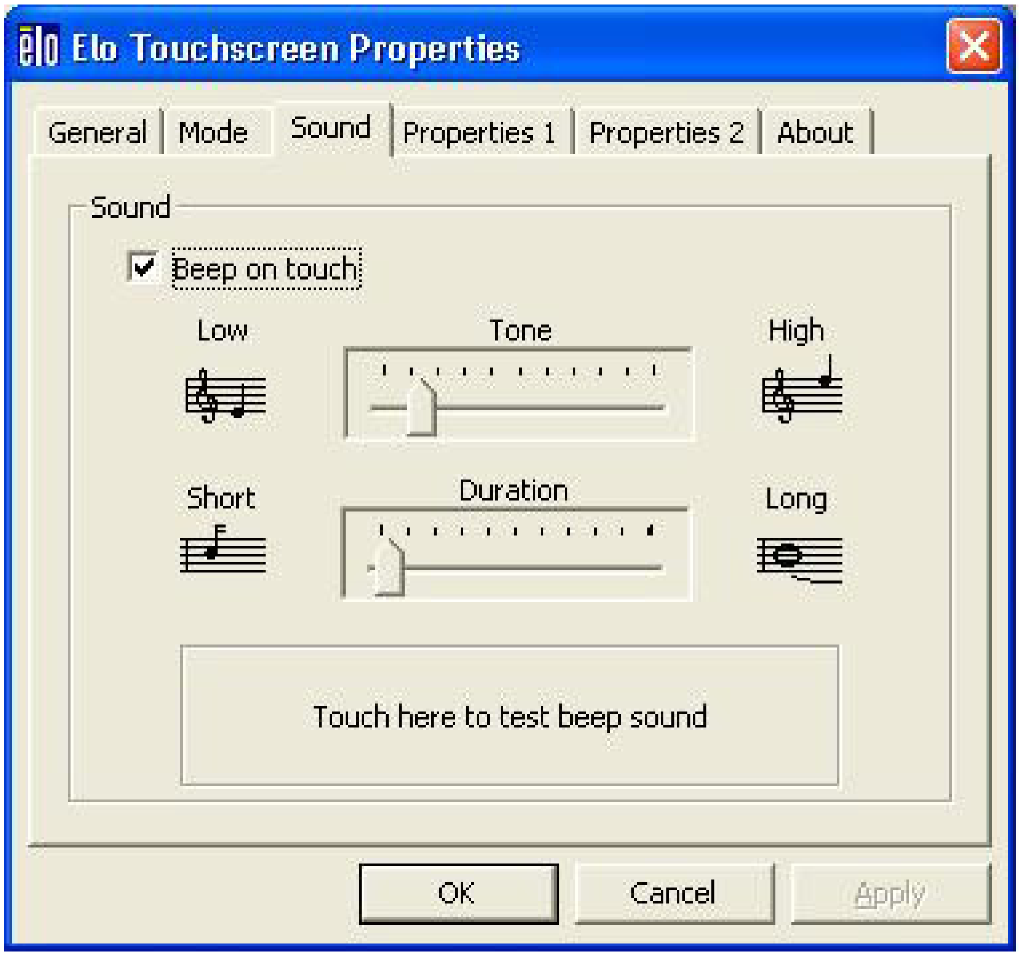
Mouse Emulation (Drag and Double-click)

- Sends a mouse down message at the point of contact.
- Selects an object if it was at the initial point of contact.
- Drags a selected object on the screen.
    - Response to dragging is set by the Drag Delay slider bar.
    - Sends a mouse up message at the point of untouch.
    - Double-clicks on an object when the screen is touched twice in succession at the same location.
    - The appropriate speed to achieve double-click is identical to the speed of a successful double-click with the mouse.
    - Double-click speed is set from the Mouse Properties control panel (Mouse Properties>Buttons>Double-click speed).
    - Double click area graphically sets the dimensions of the location around each clickable icon or object on the screen which will be recognized by Windows as a double-click. The size of the wire-frame square displayed in the Double click area tab is the actual size of double-click area accepted by Windows. The square is increased or decreased in size by touching the appropriate arrows adjacent to the square. Note that the double-click box size is independent of screen resolution.

Options

- Options allow various features of the desktop related to the touchscreen to be configured. Each of the features selected must be activated by clicking the "Apply" button at the bottom of the tab.
    - **Hide Mouse Cursor** turns off the standard mouse cursor.
    - **Left-Hand Mouse** interchanges the standard two-button mouse button assignments.
    - **Show Tool Tray Utility** activates the Tool Tray utility in the Windows Task Bar. See Tool Tray for a complete description of this feature.

Sound



Beep on touch [✓]

Touch here to test beep sound

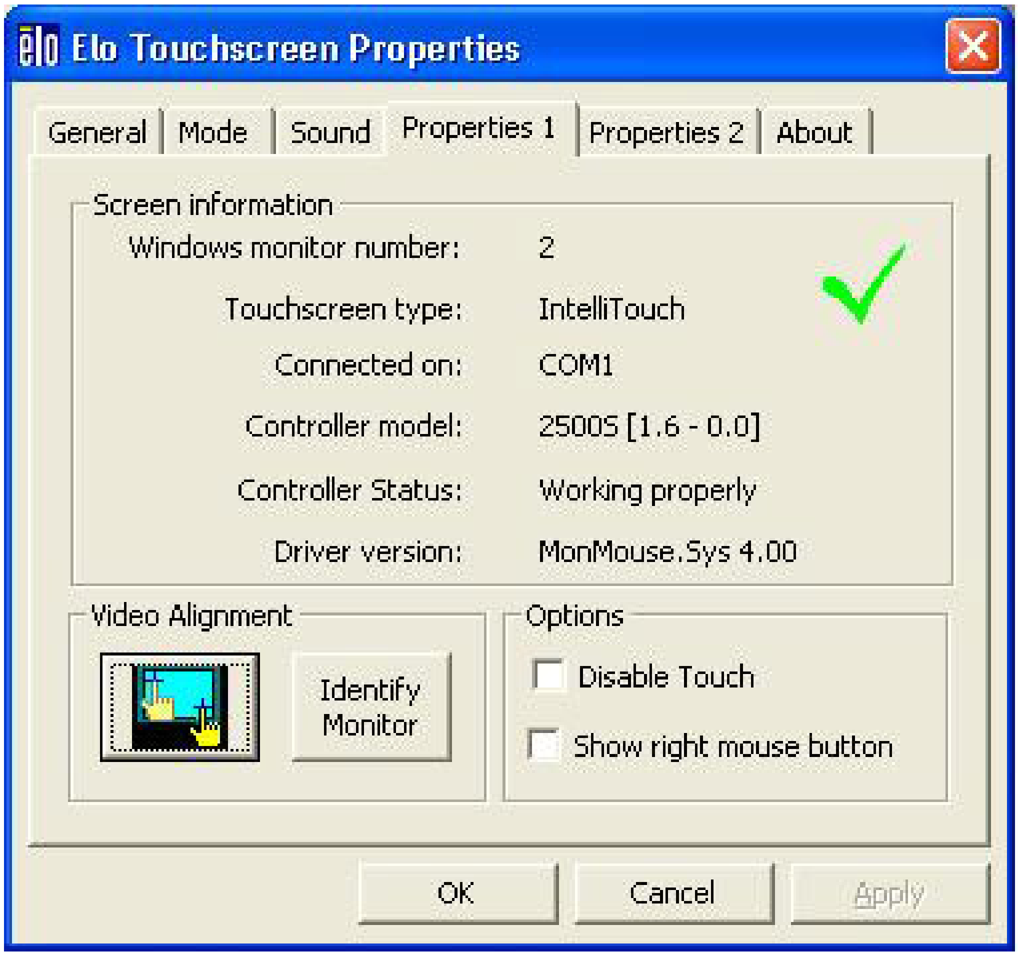
- The Sound tab sends a single-frequency tone, or "Beep" to the system speaker each time that a valid touch occurs.
- The beep is enabled by default when the driver is installed. It may be turned off by unchecking the Beep on touch box in this tab.
- The frequency (Tone) and the Duration of the beep can be adjusted by moving the appropriate slider in this tab with the touchscreen or mouse, or by using the keyboard arrow keys.
- Selections or changes to the settings in this tab take effect when the "Apply" button is clicked.

Properties[ ]

- A Properties page will be created for each touchscreen controller installed by EloSetup and for each serial port reserved for a controller, even if that controller is not present. A number will be assigned to each Properties page that is related to the order in which controllers or ports were detected or enumerated. Each Properties page contains information extracted from the touchscreen controller and the system about the monitor, touchscreen, controller and internal driver.

- ## Screen Information

Properties 1 Tab



Screen information

Windows monitor number: 2

Touchscreen type: IntelliTouch

Connected on: COM1

Controller model: 2500S [1.6 - 0.0]

Controller Status: Working properly

Driver version: MonMouse.Sys 4.00

Video Alignment

Options

☐ Disable Touch

☐ Show right mouse button

Windows Monitor Number

- Windows monitor number lists the monitor that corresponds to the "Windows Monitor" from the Windows Display Properties control panel.

Touchscreen Type

- Touchscreen Type lists the Elo touchscreen used by its marketing name, i.e., AccuTouch for Elo's five-wire resistive touchscreen

Connected On

- Connected On indicates USB or lists the Windows COM port that a serial controller is connected to.

Controller Model

- Controller Model lists the marketing model number of the controller attached, and also lists the revision of the firmware loaded in the controller.

Controller Status

- Controller Status indicates either "Working properly" or reflects any errors reported from the controller.

Driver Version

- Driver Version lists the version number of the internal driver being used for serial and USB devices as appropriate.

Each Properties page also contains a **Screen configuration** section that has two functions:

- The **Identify monitor** button will display the Elo logo on the monitor associated with this Properties page for about one second each time this button is touched.

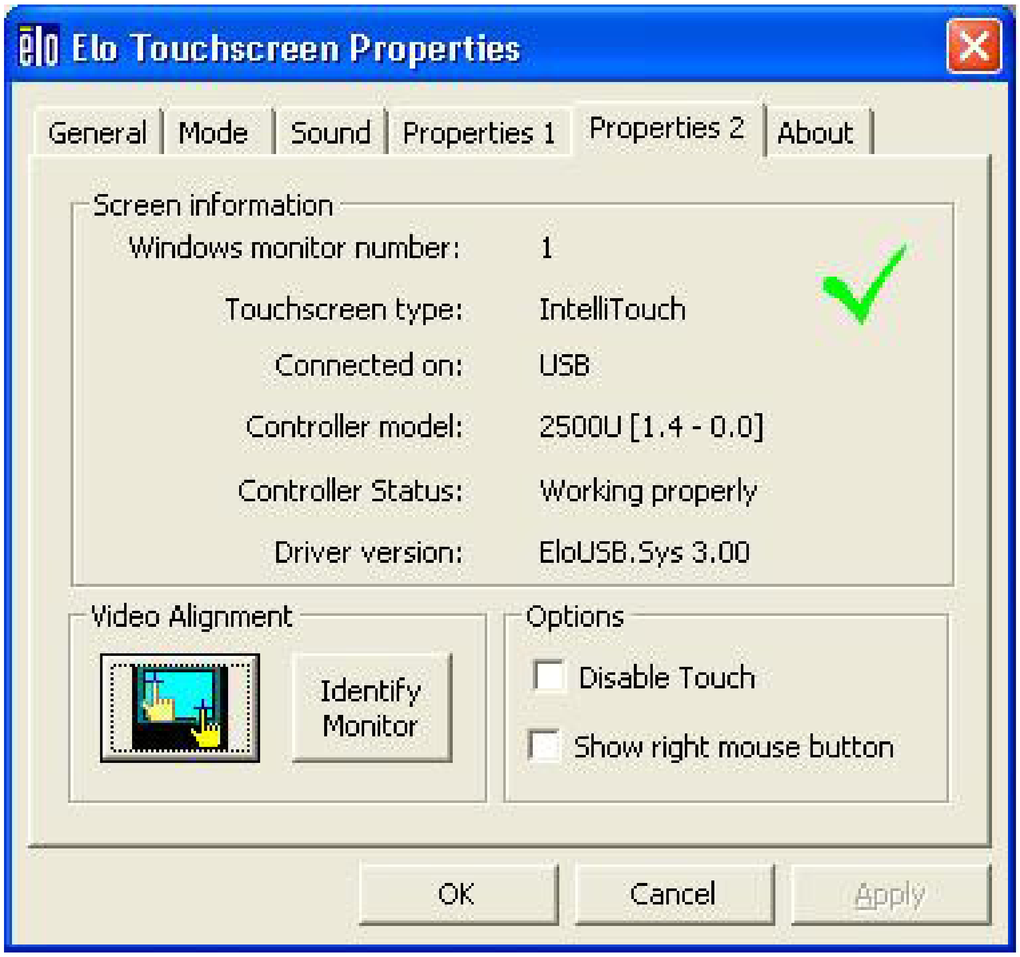
- The **EloVA icon** will launch the video alignment program **only for the monitor associated with the current Properties page**. Running EloVA from the Properties page avoids the need to align all monitors in a multiple monitor application when only a single monitor alignment is desired.

Each Properties page also contains a **Options** section that has two functions.

- **Show Right Mouse Button Tool** launches this feature for this monitor. See Right Mouse Button for a complete description of this feature.
- **Disable Touch** stops touch information from reaching Windows for this touchscreen. This feature may also be activated from the **Tool Tray** utility. Disabling touch has no effect on the operation of the standard mouse.

If a serial port is reserved for a touchscreen controller that is not actually installed, the only information contained in the Screen information section of the associated Properties page will be the number of the Windows COM port. No Screen configuration icons will be displayed.

Properties 2 Tab (USB)

- 

Elo Touchscreen Properties

Screen information

Windows monitor number: 1

Touchscreen type: IntelliTouch

Connected on: USB

Controller model: 2500U [1.4 - 0.0]

Controller Status: Working properly

Driver version: EloUSB.Sys 3.00

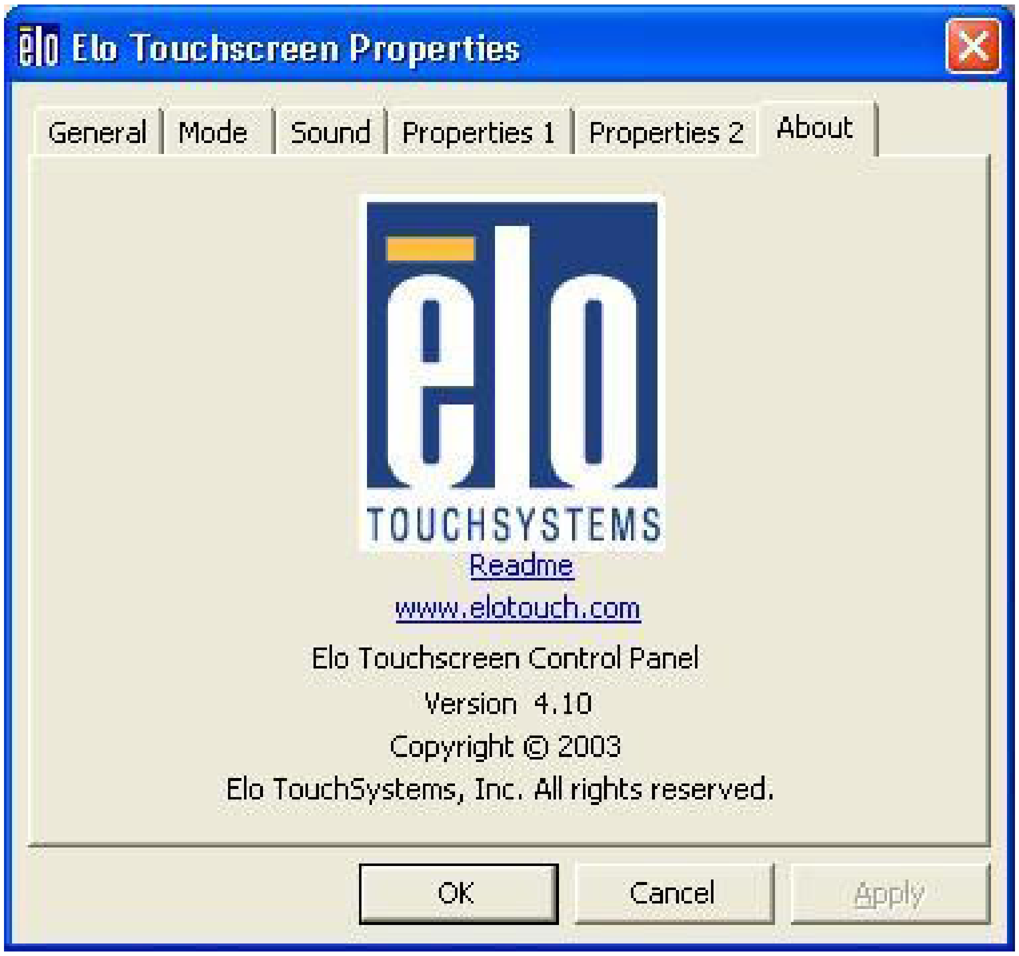
Options

- [ ] Disable Touch
- [ ] Show right mouse button

About

The About tab provides the version of the Control Panel (which is linked to the driver version), and also provides links to

- Readme document
- Elo's web site



Readme

www.elotouch.com

Elo Touchscreen Control Panel

Version 4.10

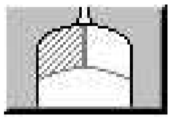
Copyright © 2003

Elo TouchSystems, Inc. All rights reserved.

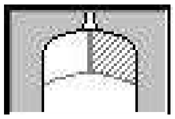
Elo Right Mouse Button Tool (RMBT)

Right Mouse Button tool allows a Windows right mouse button simulation on the touchscreen.

A representation of a typical two-button mouse is displayed in a small window on the desktop, when this application is run. The initial presentation of the RMBT shows the left button shaded, indicating that the left button is active. Any touch on the desktop or an application will produce a left button click consistent with the Button settings in the Control Panel.



A touch in the RMBT will change the shading to show the right mouse button active. Now any touch on this monitor on which RMBT is running will produce a right button click.



Typically, a right button action will activate a new dialog box for some function similar to mouse right button click. When that dialog box is touched, or any other touch on the touch monitor is made, this touch will be a left button event. Simultaneously, the mouse button shading in the RMBT will toggle back to the left button.

After the RMBT has been touched to toggle to the right button state, a second touch in the RMBT will activate a menu allowing the user to:

- Launch the Elo Control Panel (Elo Touchscreen Properties)
- Close the right button utility
- A right button click on the RMBT with the mouse will also activate this menu

The RMBT may be dragged to any location on the desktop by touching it and holding the touch momentarily until the crossed arrows appear. The RMBT may also be dragged with the standard mouse.

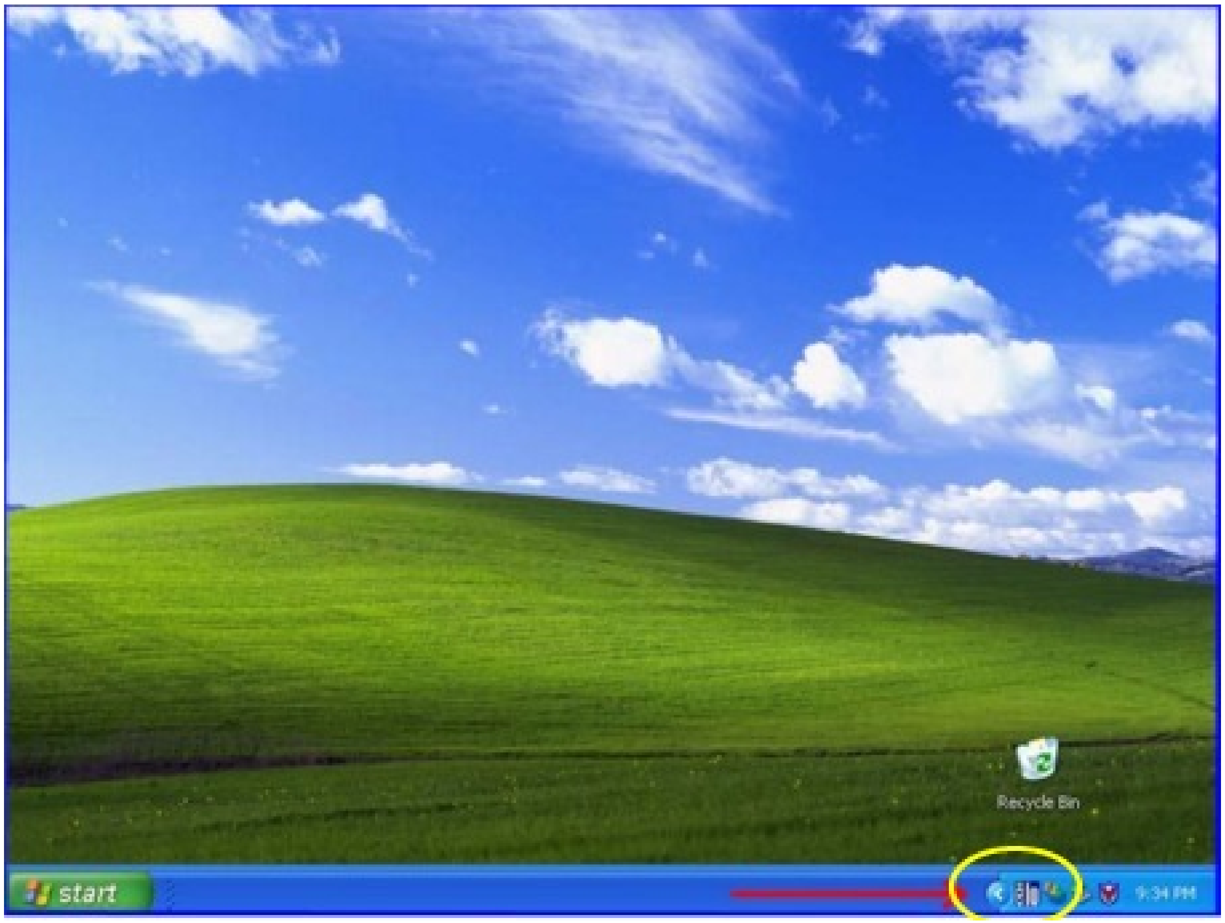
The RMBT cannot be resized.

The RMBT can be run for every touchmonitor from the Elo Control Panel > Properties tab.

Only one RMBT can be active for each touchmonitor.

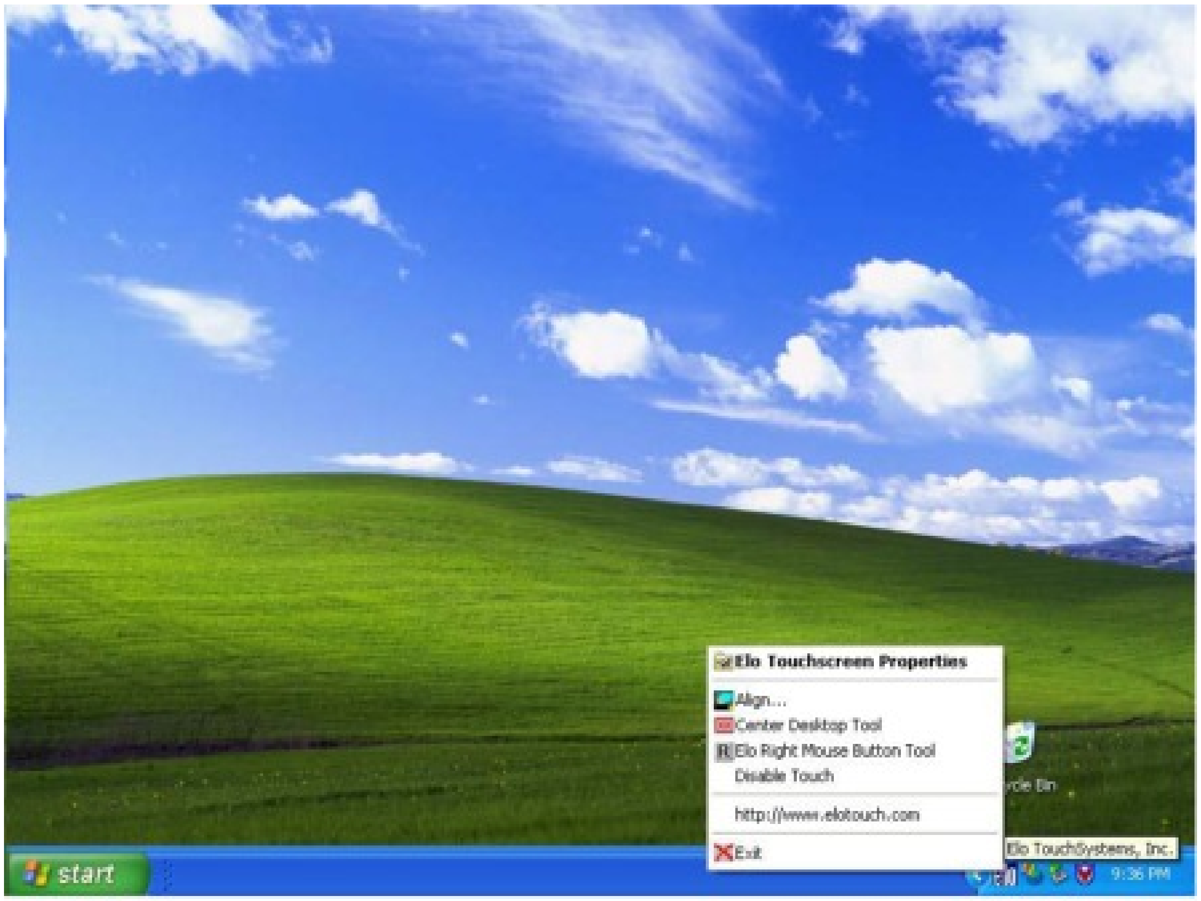
Tool Tray

The Tool Tray application offers a convenient way to access commonly used functions of the driver from the operating system desktop.



The Tool Tray icon is enabled/disabled from the Control Panel in the Button tab.

The Tool Tray application is launched with a single left or right click, and pops up a menu with several selectable functions.



The Tool Tray functions are:

Elo Touchscreen Properties

- A one click method to launch the Elo Control Panel.

Align

- Launches EloVA, the Elo Video Alignment program.

Elo Right Mouse Button Tool (RMBT)

- Launches Right Mouse Button Tool for all touch monitors that have been calibrated.

Center Desktop Tool

- Launches Center Desktop Tool

Disable/Enable Touch

- When driver is loaded and running, clicking this segment of the Tool Tray menu will disable the driver, stopping all touch reports from being sent from the driver to the mouse handler (MouClass)
- When disabled through the Tool Tray, touches made on the touchscreen are not buffered by the driver, and will not be sent from the driver to MouClass after the touchscreen is re-enabled.
- After the touchscreen has been disabled, the state of this segment of the Tool Tray menu changes from "Disable" to "Enable." Since the touchscreen is now disabled, it can only be enabled by using the mouse or keyboard. Clicking "Enable" in the Tool Tray will now restore touchscreen operation.
- It is also possible to re-enable the touchscreen from the Control Panel after touch has been disabled from the Tool Tray. Open the Control Panel with mouse or keyboard, uncheck the "Disable Touch" box in the Mode>Options tab, click "Apply" and touch is restored.

http://www.elotouch.com

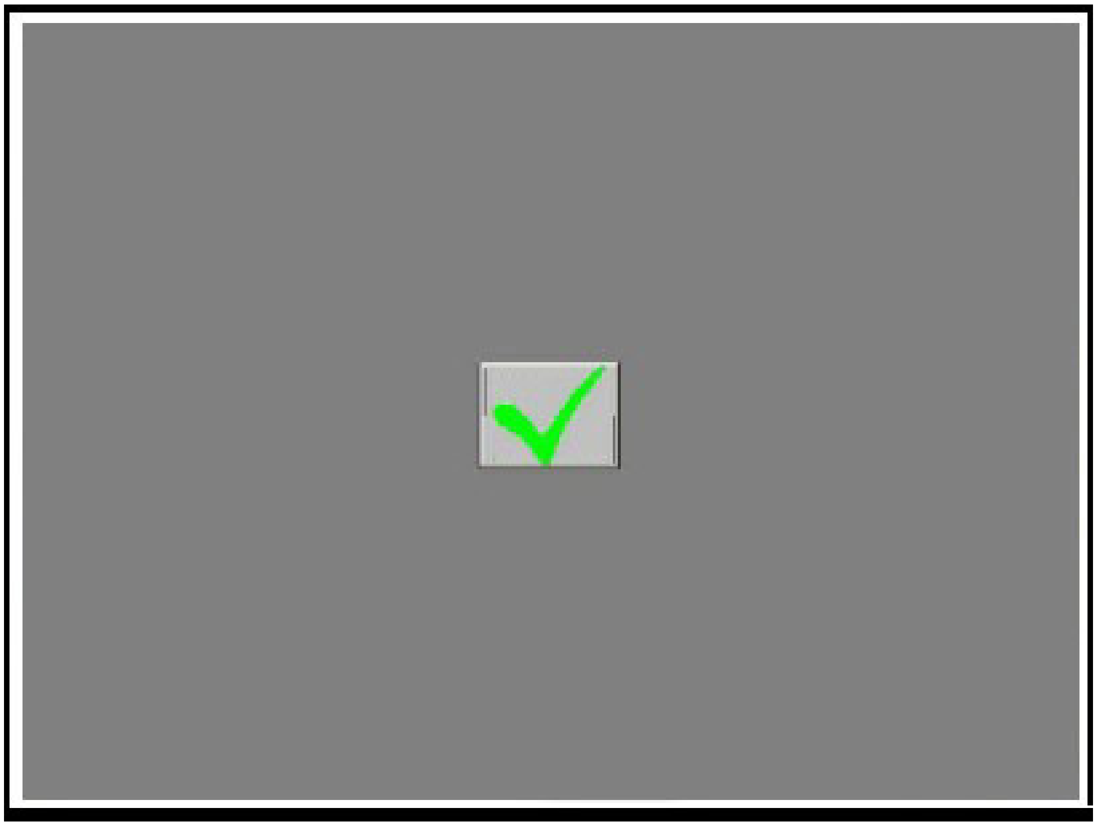
- A quick link to Elo's website

Exit

- Clicking this button exits the Tool Tray application and also removes it from the Windows Taskbar.
- The Tool Tray application may also be removed by unchecking it in the Control Panel and clicking "Apply."

Center Desktop Tool

- Presents a full screen image for the selected monitor to allow proper adjustment of the video prior to running the EloVA video alignment program
- Configurable parameters are
  - Border width, in pixels
  - Which monitor to adjust in a multiple monitor configuration
- Can be launched from the Tool Tray (for single monitor) or command line (for multiple monitors)
- Configuration syntax menu is available by running eloalmon /h. Only one monitor may be adjusted for each execution of the program.
- Program terminates when green "check" mark is touched or clicked with the mouse, or from the keyboard by pressing Esc, Enter or Space. See graphic below.



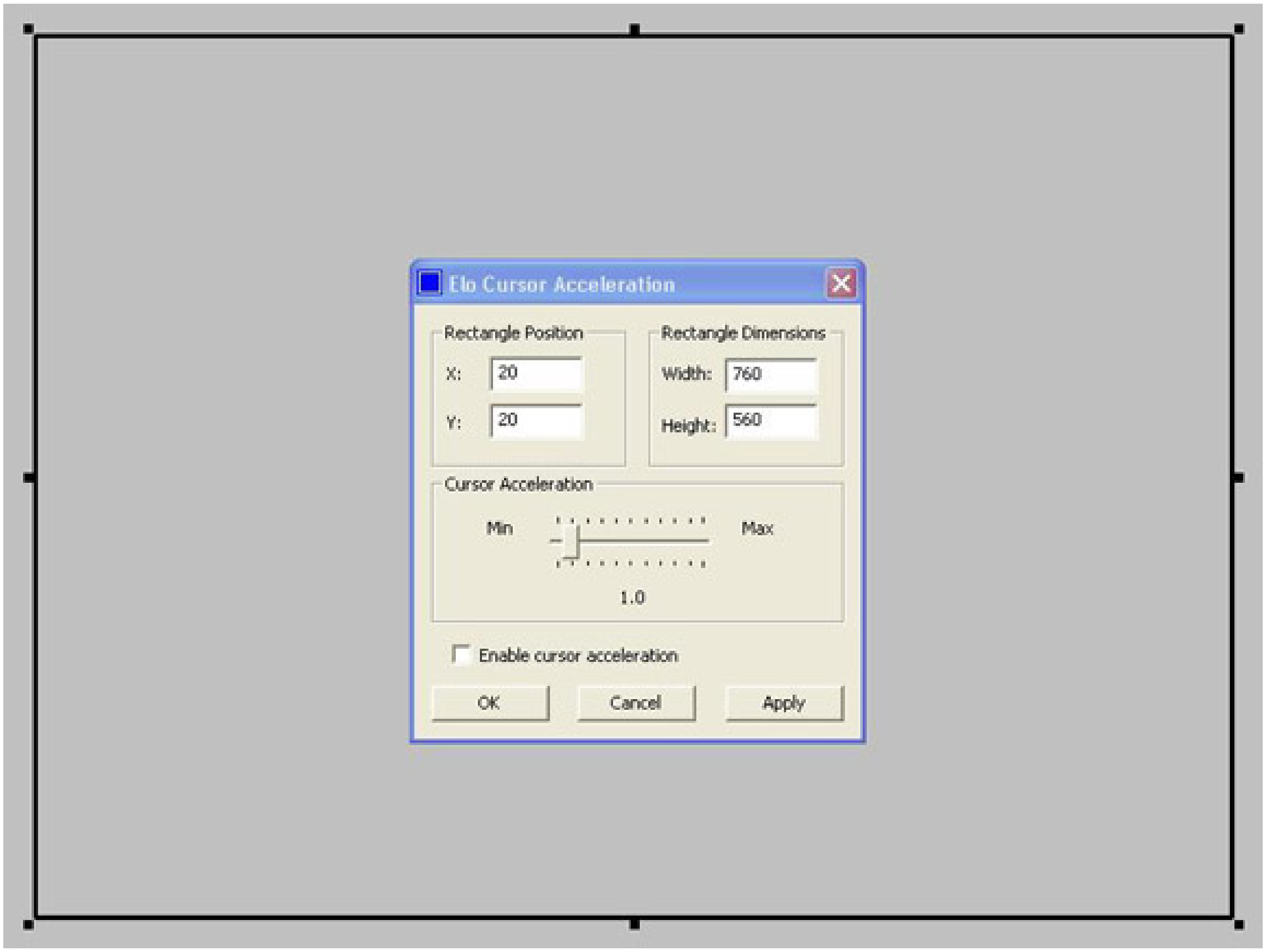
- The center desktop tool solves a common problem in video alignment programs-how to properly size the video image. The image presented is a black screen with a white border, which makes the edge of the image visible and allows it to be properly adjusted on CRT displays as well as flat panel displays of all technologies.

Edge Acceleration Tool (EAT)

Edge Acceleration tool can be used to configure cursor acceleration towards the edge of the touchscreen. Edge Acceleration is a special feature provided to allow touches towards the edges.

EAT (i.e. EloAccel.exe) can be launched from Program Files> EloTouchsystems folder.

Presents a full screen image with the edge acceleration rectangle bounds and a **Cursor Acceleration Configuration** dialog, displaying the configuration parameters for Edge Acceleration.



X: 20
Y: 20
Width: 760
Height: 560
1.0
Enable cursor acceleration

Configurable parameters are:

- Cursor Acceleration scale. This scale can be set from a value of 0-10 where, 1 corresponds to no acceleration.
- Edge acceleration rectangle bounds dimensions.
- Edge acceleration rectangle bounds position

- Enable Edge acceleration

Cursor is accelerated beyond the rectangle bounds.

These parameters can be configured using mouse, to resize or move the bounds rectangle or, by entering the values directly in the dialog box. Changes to the settings in the Cursor Acceleration dialog take effect when the "Apply" button is clicked.

This applies to **all touchmonitors** unless run with appropriate command line options. To view the options available, run EloAccel using **EloAccel /h**.

File List

The following list identifies all the files in the basic driver package, and the installed location of each file. Most of the files are common to both the serial and USB drivers. Those files that are associated with only one of the serial or USB drivers are so identified.

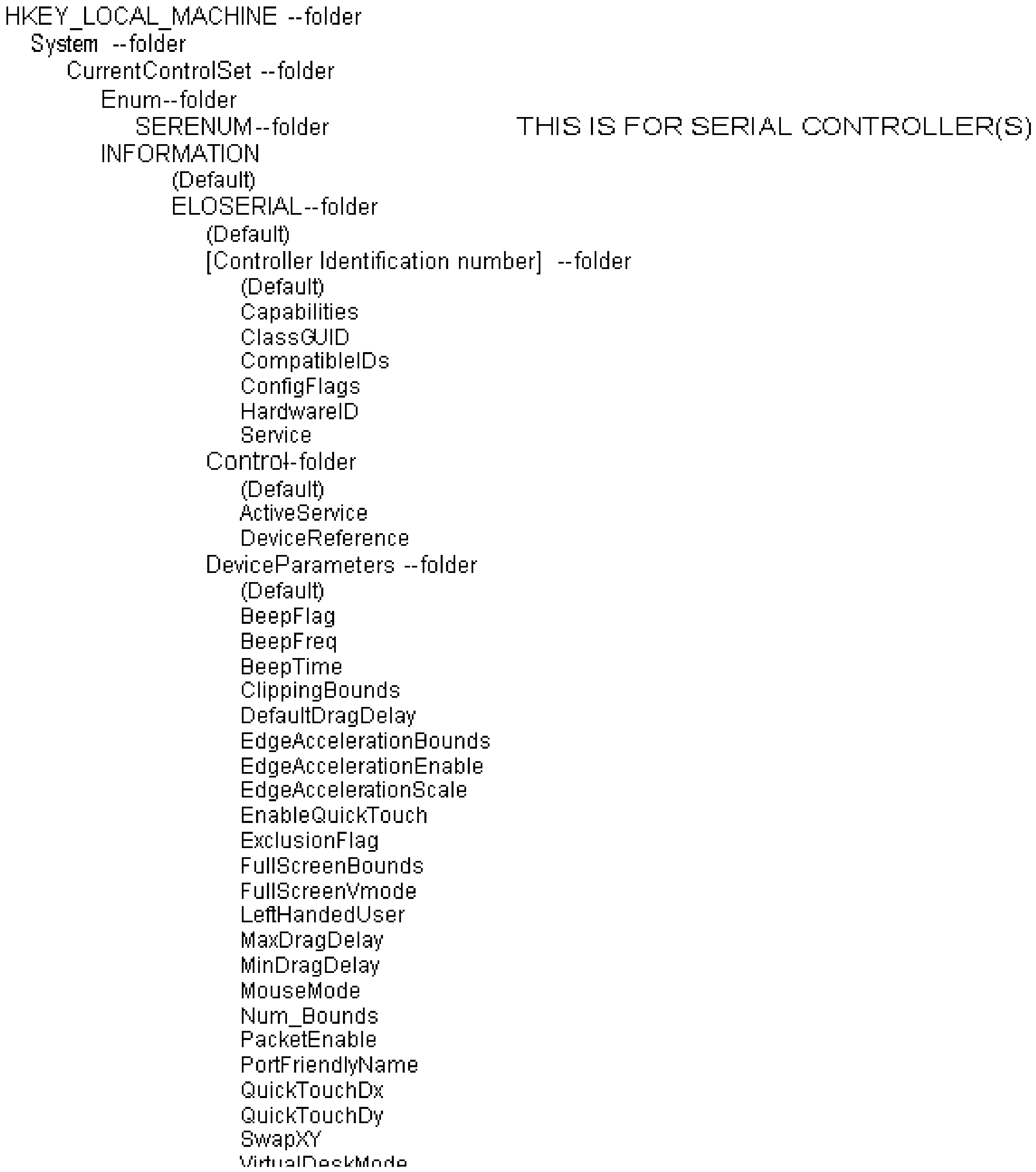
File	**Installed Location**
Common Files	
EloAIMon.exe	\Windows\System32
EloDkMon.exe	\Windows\System32
EloIntf.dll	\Windows\System32
EloLCoin.dll	\Windows\System32
EloLnchr.exe	\Windows\System32
EloProp.dll	\Windows\System32
EloRtBtn.exe	\Windows\System32
EloSerCo.dll	\Windows\System32
EloSetup.exe	\Program Files\EloTouchSystems
EloSrvce.exe	\Windows\System32
EloSrvCt.exe	\Windows\System32
EloTouch.cpl	\Windows\System32
EloTTray.exe	\Windows\System32
EloUCoin.dll	\Windows\System32
EloVA.exe	\Windows\System32
License.txt	\Program Files\EloTouchSystems
Null.cur	\Windows\CURSORS
Readme.txt	\Program Files\EloTouchSystems
Readme-S.txt	\Program Files\EloTouchSystems
Readme-U.txt	\Program Files\EloTouchSystems
Serial	
EloBus.sys	\Windows\System32\DRIVERS
EloSer.sys	\Windows\System32\DRIVERS
WinSerXP.inf	\Program Files\EloTouchSystems

USB

EloFiltr.cat	\Program Files\EloTouchSystems
EloFiltr.inf	\Program Files\EloTouchSystems
EloFiltr.sys	\Windows\System32\DRIVERS
EloUSB.sys	\Windows\System32\DRIVERS
Elouxpsi.cat	\Program Files\EloTouchSystems
EloUXpSI.inf	\Program Files\EloTouchSystems

Registry Entries

Registry (Primary registry entries):



```
HKEY_LOCAL_MACHINE --folder
   System  --folder
       CurrentControlSet --folder
           Enum--folder
               SERENUM--folder                      THIS IS FOR SERIAL CONTROLLER(S)
           INFORMATION
                   (Default)
                   ELOSERIAL--folder
                       (Default)
                       [Controller Identification number]  --folder
                           (Default)
                           Capabilities
                           ClassGUID
                           CompatibleIDs
                           ConfigFlags
                           HardwareID
                           Service
                       Control--folder
                           (Default)
                           ActiveService
                           DeviceReference
                       DeviceParameters --folder
                           (Default)
                           BeepFlag
                           BeepFreq
                           BeepTime
                           ClippingBounds
                           DefaultDragDelay
                           EdgeAccelerationBounds
                           EdgeAccelerationEnable
                           EdgeAccelerationScale
                           EnableQuickTouch
                           ExclusionFlag
                           FullScreenBounds
                           FullScreenVmode
                           LeftHandedUser
                           MaxDragDelay
                           MinDragDelay
                           MouseMode
                           Num_Bounds
                           PacketEnable
                           PortFriendlyName
                           QuickTouchDx
                           QuickTouchDy
                           SwapXY
                           VirtualDeskMode
```

```
                SwapXY
                VirtualDeskMode
                WindowsMonitorNumber
                X_EloDx
                X_Offset
                X_Resolution
                X_ScrDx
                xMonLocn
                xVirtScrnCoord
                xVirtScrSize
                Y_EloDx
                Y_Offset
                Y_Resolution
                Y_ScrDx
                yMonLocn
                yVirtScrnCoord
                yVirtScrSize
HKEY_LOCAL_MACHINE  --folder
  System --folder
    CurrentControlSet --folder
      Enum --folder
        USB --folder                          THIS IS FOR USB CONTROLLER(S)
      INFORMATION
            Vid_04e7&Pid_#### --- folder
              Default
              [Controller Identification number]  --folder
                (Default)
                Capabilities
                ClassGUID
                CompatibleIDs
                ConfigFlags
                DeviceDesc
                Driver
                HardwareID
                LocationInformation
                Mfg
                ParentIdPrefix
                Service
              DeviceParameters --folder
                (Default)
                SymbolicName
                BeepFreq
              LogConfig--folder
                (Default)
```

Second Registry Entries:

```
HKEY_LOCAL_MACHINE
   System
       CurrentControlSet
           Services
               elobus
                   Default
                   DisplayName
                   ErrorControl
                   ImagePath
                   Start
                   Type
```

Primary Registry Entries:

```
                   Enum
                       Default
                       0
                       Count
                       NextInstance
                   Security
                       Default
                       Security
               EloSer
                                                       Same keys as elobus, above
                   Enum
                       Same keys as elobus Enum, above
                   Security
                       Same keys as elobus Security, above
               EloSystemService
                   Same keys as elobus, above
                   Enum
                       Same keys as elobus Enum,above
                   Security
                       Same keys as elobus Security, above
```

Troubleshooting

Is it a hardware or a software problem? How to decide?

All touchscreen controllers that are supported by this driver are serial or USB.

If you have a serial controller, and are unsure about whether or not the controller or touchscreen is functioning correctly, use Elo's serial port utility, COMDUMP.EXE, available from Elo's web site or any Elo TouchTools CD. The syntax for COMDUMP is:

COMDUMP [COM port]

This program can be run from a command line. Select a serial port that is NOT being used by the driver, or uninstall the driver first.

As an alternative, you may be able to see the status light on the serial controller, and determine whether or not the touchscreen and controller are operating properly from the status light function. If the light is visible, perform the following steps:

- Disconnect the serial cable from the computer.
- Observe the status light, a small green or yellow LED on the controller PCB.
    - If the light blinks at approximately once per second with no touch applied, touch the touchscreen and observe the light again while your finger is still touching the screen.
        - If the status light illuminates continuously while the screen is touched, stop touching.
        - If the light reverts to the once per second blink when no longer touched, the touchscreen and controller are probably functioning properly.
    - If the status light blinks at approximately 2-3 times per second with no touch, cycle power on the display or controller first.
        - If cycling power causes the status light to revert to blinking once per second, follow the steps in the first case to evaluate performance.
        - If cycling the power produces no change in the blink rate, there is a hardware fault in the controller or touchscreen that may interfere with the operation of the system.
        - Touch the touchscreen and observe the light again while your finger is still touching the screen.
        - If the status light illuminates continuously while the screen is touched, stop touching.
        - If the light reverts to the 2-3 times per second blink when no longer touched, the touchscreen or controller may have a fault, but are probably functioning well enough to work with the driver.

If you are working with a USB controller, the best hardware troubleshooting method is to observe cursor motion on the touchscreen display with only the Windows native HID drivers installed. Even if the cursor does not move in the same direction that your finger does, when you drag your finger on the touchscreen, cursor motion itself is sufficient to tell you that your hardware is working correctly.

There are status lights for USB controllers as well:

- Connect a USB cable to the system
- A touch status light blinks at a once per second rate with no touch applied. Operation is similar to that of the serial controller status lights described above.
- A second status light (located very close to the USB connector) is continuously illuminated except when touch is applied. With touch, this light blinks at 2-3 times per second. This USB status light is only monitoring the status of the USB bus, and is not concerned with the interaction between the touchscreen and controller.

If your hardware is functioning correctly, and you believe that your problem is driver software related, start by:

- Uninstalling the driver
- If you are unsure as to the success of the uninstall process, check the file lists and the discussion in the Uninstalling the Driver section of this manual. You can successfully remove the files manually if necessary.
- Reinstall the driver.
- If problems persist, contact Elo Applications Engineering at one of the locations mentioned in the section, Contacting Elo.

Appendix A

Device Specific Registry Keys

Introduction

This document explains the use of all the registry keys used by the "XP Universal Driver Package".

In order for device specific changes to be effective, the device must be restarted (i.e. disable the device and then enable it) in Device Manager, or the system must be rebooted.

For user specific changes to become effective, it is necessary to log off and log back on again.

Registry Keys for Serial

Device Parameters for Elo serial touchscreen are located in subkeys of

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\SERENUM\

ELOSERIAL*******\Device Parameters key.

The "******" is the id used by the Windows system.

Touch Mode

Name:	MouseMode
Data Type:	DWORD (REG_DWORD)

Explanation:

Touchscreen mode

Possible Values:

Default is 6

0: Click on touch

1: Click on release

6: Mouse emulation

Name: EnableQuickTouch

Data Type: DWORD (REG_DWORD)

Explanation:

Enables / disables quick touch mode

Possible Values:

Default is 0

0: Disabled

1: Enabled

Name: QuickTouchDx

Data Type: DWORD (REG_DWORD)

Explanation:

X distance in Windows virtual coordinates outside which quick touch is enabled. Within this distance two quick touches are interpreted as regular touches.

Name: QuickTouchDy

Data Type: DWORD (REG_DWORD)

Explanation:

Y distance in Windows virtual coordinates outside which quick touch is enabled. Within this distance two quick touches are interpreted as regular touches.

Name: UntouchDelay

Data Type: DWORD (REG_DWORD)

Explanation:

"UntouchDelay" defines the time in seconds for the timeout. If there is constant touch at the same location a Button Up event will be generated automatically after the timeout period expires.

Possible Values:

Default value is 10 seconds.

Name: SwapButton

Data Type: DWORD (REG_DWORD)

Explanation:

Swaps the mouse left and right buttons. The buttons remain swapped for the count of touches this value is set to. It remains swapped until this count is decremented to 0. After that, the buttons are automatically swapped back to their original state.

Possible Values:

By default this key is not present in the registry. When the DLL API feature is used the mouse buttons are swapped and this value is added.

Name: LeftHandedUser

Data Type: DWORD (REG_DWORD)

Explanation:

This is applicable for left-handed mouse users.

Possible Values:

Default is 0

0: Disabled

1: Enabled

Name: RightButtonActive

Data Type: DWORD (REG_DWORD)

Explanation:

Indicates that the right button application should be run at user logon for this screen.

Possible Values:

1: Run right button at user logon.

2: Do not run right button for this screen at user logon.

Serial Port Configuration

Name: PortFriendlyName

Data Type: String (REG_SZ)

Explanation:

PortFriendlyName identifies the serial port the touchscreen is connected to. The friendly name is the COM port identifier string used by Device Manager and can be found by expanding the Ports node of the Device Manager hardware tree.

To change the COM port associated with a touchscreen, replace the existing value in HKLM\System\CurrentControlSet\Services\EloTouchscreen\LegacyPorts\**\FriendlyName with the friendly name of the desired COM port and restart the system.

Possible Values:

Communications Port (COM1)

Communications Port (COM2)

Name: BaudRate

Data Type: DWORD (REG_DWORD)

Explanation:

Baud rate used for serial port communication. Controller must be externally configured to support any baudrate other than 9600.

Possible Values:

Default is 9600.

Allowed: 1200, 2400, 4800, 9600

Name: HardwareHandshaking

Data Type: DWORD (REG_DWORD)

Explanation:

This key is used only for serial cables which are built using 2 wires and do not support hardware handshake.

Possible Values:

By default this key is not present in the registry. It has to be explicitly created. By default hardware handshaking is always turned on. To turn it off, explicitly set this key to 0.

Enable/Disable Touch

Name: PacketEnable

Data Type: DWORD (REG_DWORD)

Explanation:

Touch is enabled / disabled depending on this value.

Possible Values:

1: Enabled

2: Disabled

3: Reserved

Beep on Touch

Name: BeepFlag

Data Type: DWORD (REG_DWORD)

Explanation:

Beep is enabled / disabled depending on this value.

Possible Values:

0: Disabled

1: Enabled

Name: BeepTime

Data Type: DWORD (REG_DWORD)

Explanation:

Defines the duration of beep on touch in milliseconds.

Possible Values:

Default is 0x0014.

Name: BeepFreq

Data Type: DWORD (REG_DWORD)

Explanation:

Frequency of beep on touch in Hz.

Possible Values:

Default is 0x00300

Drag Delay

Name: DefaultDragDelay

Data Type: DWORD (REG_DWORD)

Explanation:

Drag delay value in milliseconds.

Possible Values:

Default is 20

Name: MinDragDelay

Data Type: DWORD (REG_DWORD)

Explanation:

Minimum drag delay value on the scale in Control Panel in milliseconds. Reserved for use by control panel and driver.

Name: MaxDragDelay

Data Type: DWORD (REG_DWORD)

Explanation:

Maximun drag delay value on the scale in Control Panel in milliseconds. Reserved for use by control panel and driver.

Full Screen Bounding Rectangle and Bounding Mode

Name FullScreenVmode

Data Type: DWORD (REG_DWORD)

Explanation:

Defines the mode for the touchscreen boundary in the virtual desktop mode

Possible Values:

Default is 2

0: Disable virtual desktop. Touch on all screens will reflect on the primary monitor.

1: Enable virtual desktop. In this mode no individual screen bounds are used for the monitor. Touches toward the edge may generate clicks on adjacent monitors.

2: Enable virtual desktop. Screen bounds are enabled for each monitor. Touches on the screen always generate clicks on the associated monitor. A touch outside the bounding rectangle will cause the cursor to move to a point along the boundary that is nearest to the point of touch.

3: Enable virtual desktop. Screen bounds are enabled for each monitor. Touches on the screen always generate clicks on the associated monitor. Touches outside the bounding rectangle are not sent to the system.

Name: FullScreenBounds

Data Type: Array of DWORDs (REG_BINARY)

Explanation:

Defines the full screen bounding rectangle in Windows virtual coordinates for the screen.

Calibration

The following defines the calibration data used to convert touches from the Elo coordinate system to Windows virtual coordinate system.

The driver uses the following equation to convert a raw touch coordinate point from Elo coordinate system to the Windows screen coordinate system. Calibration equation is:

Xcal = a + m*Xuncal,

where,
Xuncal, is in Elo coordinate system
Xcal, is in Windows coordinate system
m = X_ScrDx / X_EloDx
"a", is the X offset value entry
X_ScrDx = distance between targets in virtual coordinates
X_EloDx = distance between targets in Elo coordinates.

Name: X_EloDx, X_ScrDx, X_Offset, Y_EloDx, Y_ScrDx, Y_Offset, Z_EloDx, Z_ScrDx, Z_Offset

Data Type: DWORD (REG_BINARY)

Explanation:

Data required by calibration. Use the equation above to calculate these values.

Name: SwapXY

Data Type: DWORD (REG_DWORD)

Explanation:

Specifies if the touchscreen is rotated 90 degrees or 270 degrees.

Possible Values:

Set to 1 if touchscreen is rotated 90 degrees or 270 degrees else set to 0

Name: WindowsMonitorNumber

Data Type: DWORD (REG_DWORD)

Explanation:

Windows monitor number.

Possible Values:

Windows monitor numbers are 1 based.

Name:	X_Resolution
Data Type:	DWORD (REG_DWORD)

Explanation:

Screen width in pixels.

Name:	Y_Resolution
Data Type:	DWORD (REG_DWORD)

Explanation:

Screen height in pixels.

Name:	xVirtScrSize
Data Type:	DWORD (REG_DWORD)

Explanation:

Width in pixels of the Windows virtual screen.

Name:	yVirtScrSize
Data Type:	DWORD (REG_DWORD)

Explanation:

Height in pixels of the Windows virtual screen.

Name: xVirtScrCoord

Data Type: DWORD (REG_DWORD)

Explanation:

Top left X coordinate of the Windows virtual screen.

Name: yVirtScrCoord

Data Type: DWORD (REG_DWORD)

Explanation:

Top left Y coordinate of the Windows virtual screen.

Name: xMonLocn

Data Type: DWORD (REG_DWORD)

Explanation:

Monitor location (X) for this monitor on the virtual desktop.

Name: yMonLocn

Data Type: DWORD (REG_DWORD)

Explanation:

Monitor location (Y) for this monitor on the virtual desktop.

Edge of Touchscreen Cursor Acceleration

Name: EdgeAccelerationEnable

Data Type: DWORD (REG_DWORD)

Explanation:

Enables / disables acceleration.

Possible Values:

Default is 0

0: Disable acceleration.

1: Enable acceleration

Name: EdgeAccelerationBounds

Data Type: Array of DWORDS (REG_BINARY)

Explanation:

Rectangle bounds (XMin, YMin, XMax, YMax) beyond which cursor movement is accelerated. Within the defined rectangle touch is interpreted normally. XMin, YMin, XMax, YMax are defined in Windows virtual screen coordinates.

Name: EdgeAccelerationScale

Data Type: DWORD (REG_DWORD)

Explanation:

Acceleration value.

Possible Values:

Default is unused.

It can be anywhere in the range of 0-100. 10 indicates no acceleration.

Registry Keys for USB Device

Device Parameters for USB touchscreen are located in the sub keys of the HID key and the USB key.

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\USB\Vid_04e7&Pid_???\*******\Device Parameters key for USB.

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\HID\Vid_04e7&Pid_???\*******\Device Parameters key for HID.

where, "???" is the product id.
And the "******" is the serial number for the touchscreen.

Enable / Disable Touch

Name:	PacketEnable
Data Type:	DWORD (REG_DWORD)

Explanation:

Touch is enabled / disabled depending on this value.

Possible Values:

1: Enabled

2: Disabled

Full Screen Bounding Rectangle and Bounding Mode

Name:	FullScreenVmode
Data Type:	DWORD (REG_DWORD)

Explanation:

Defines the mode for the touchscreen boundary in the virtual desktop mode.

Possible Values:

Default is 2.

0: Disable virtual desktop. Touch on all screens will reflect on the primary monitor.

1: Enable virtual desktop. In this mode no individual screen bounds are used for the monitor. Touches toward the edge may generate clicks on adjacent monitors.

2: Enable virtual desktop. Screen bounds are enabled for each monitor. Touches on the screen always generate clicks on the associated monitor. A touch outside the bounding rectangle will cause the cursor to move to a point along the boundary that is nearest to the point of touch.

3: Enable virtual desktop. Screen bounds are enabled for each monitor. Touches on the screen always generate clicks on the associated monitor. Touches outside the bounding rectangle are not sent to the system.

Name:	FullScreenBounds

Data Type:	Array of DWORDs (REG_BINARY)

Explanation:

Defines the full screen bounding rectangle in Windows virtual coordinates for the screen.

Calibration

The following defines the calibration data used to convert touches from the Elo coordinate system to Windows virtual coordinate system.

The driver uses the following equation to convert a raw touch coordinate point from Elo coordinate system to the Windows screen coordinate system.

Calibration equation is
Xcal = a + m*Xuncal,

where,
Xuncal, is in Elo coordinate system
Xcal, is in Windows coordinate system
m = X_ScrDx / X_EloDx
"a", is the X offset value entry
X_ScrDx = distance between targets in virtual coordinates
X_EloDx = distance between targets in Elo coordinates.

Name:	X_EloDx, X_ScrDx, X_Offset, Y_EloDx, Y_ScrDx, Y_Offset, Z_EloDx, Z_ScrDx, Z_Offset
Data Type:	DWORD (REG_DWORD)

Explanation:

Data required by calibration. Use the equation above to calculate these values.

Name:	SwapXY
Data Type:	DWORD (REG_DWORD)

Explanation:

Specifies if the touchscreen is rotated 90 degrees or 270 degrees.

Possible Values:

Set to 1 if touchscreen is rotated 90 degrees or 270 degrees else set to 0

Name: WindowsMonitorNumber

Data Type: DWORD (REG_DWORD)

Explanation:

Windows monitor number.

Possible Values:

Windows monitor numbers are 1 based.

Name: X_Resolution

Data Type: DWORD (REG_DWORD)

Explanation:

Screen width in pixels.

Name: Y_Resolution

Data Type: DWORD (REG_DWORD)

Explanation:

Screen height in pixels.

Name: xVirtScrSize

Data Type: DWORD (REG_DWORD)

Explanation:

Width in pixels of the Windows virtual screen.

Name: yVirtScrSize

Data Type: DWORD (REG_DWORD)

Explanation:

Height in pixels of the Windows virtual screen.

Name: xVirtScrCoord

Data Type: DWORD (REG_DWORD)

Explanation:

Top left X coordinate of the Windows virtual screen.

Name: yVirtScrCoord

Data Type: DWORD (REG_DWORD)

Explanation:

Top left Y coordinate of the Windows virtual screen.

Name: xMonLocn

Data Type: DWORD (REG_DWORD)

Explanation:

Monitor location (X) for this monitor on the virtual desktop.

Name: yMonLocn

Data Type: DWORD (REG_DWORD)

Explanation:

Monitor location (Y) for this monitor on the virtual desktop.

Edge of Touchscreen Cursor Acceleration

Name:	EdgeAccelerationEnable
Data Type:	DWORD (REG_DWORD)

Explanation:

Enables / disables acceleration.

Possible Values:

Default is 0.

0: Disable acceleration.

1: Enable acceleration.

Name:	EdgeAccelerationBounds
Data Type:	Array of DWORDS (REG_BINARY)

Explanation:

Rectangle bounds (XMin, YMin, XMax, YMax) beyond which cursor movement is accelerated. Within the defined rectangle touch is interpreted normally. XMin, YMin, XMax, YMax are defined in Windows virtual screen coordinates.

Name:	EdgeAccelerationScale
Data Type:	DWORD (REG_DWORD)

Explanation:

Acceleration value.

Possible Values:

Default is unused.

It can be anywhere in the range of 0-100. 10 indicates no acceleration.

Registry Keys for HID

Touch Mode

Name: MouseMode

Data Type: DWORD (REG_DWORD)

Explanation:

Touchscreen mode.

Possible Values:

Default is 6.

0: Click on touch.

1: Click on release.

6: Mouse emulation.

Name: EnableQuickTouch

Data Type: DWORD (REG_DWORD)

Explanation:

Enables / disables quick touch mode

Possible Values:

Default is 0.

0: Disabled

1: Enabled

Name: QuickTouchDx

Data Type: DWORD (REG_DWORD)

Explanation:

X distance in Windows virtual coordinates outside which quick touch is enabled. Within this distance two quick touches are interpreted as regular touches.

Name: QuickTouchDy

Data Type: DWORD (REG_DWORD)

Explanation:

Y distance in Windows virtual coordinates outside which quick touch is enabled. Within this distance two quick touches are interpreted as regular touches.

Name:

UntouchDelay

Data Type:

DWORD (REG_DWORD)

Explanation:

"UntouchDelay" defines the time in seconds for the timeout. If there is constant touch at the same location a Button Up event will be generated automatically after the timeout period expires.

Possible Values:

Default value is 10 seconds.

Name:

SwapButton

Data Type:

DWORD (REG_DWORD)

Explanation:

Swaps the mouse left and right buttons. The buttons remain swapped for the count of touches this value is set to. It remains swapped until this count is decremented to 0. After that, the buttons are automatically swapped back to their original state.

Possible Values:

By default this key is not present in the registry. When the DLL API feature is used the mouse buttons are swapped and this value is added.

Name:

LeftHandedUser

Data Type:

DWORD (REG_DWORD)

Explanation:

This is applicable for left-handed mouse users.

Possible Values:

Default is 0.

0: Disabled

1: Enabled

Name: RightButtonActive

Data Type: DWORD (REG_DWORD)

Explanation:

Indicates that the right button application, should be run at user logon for this screen.

Possible Values:

1: Run right button at user logon.

2: Do not run right button for this screen at user logon.

Beep On Touch

Name: BeepFlag

Data Type: DWORD (REG_DWORD)

Explanation:

Beep is enabled / disabled depending on this value.

Possible Values:

0: Disabled

1: Enabled

Name: BeepTime

Data Type: DWORD (REG_DWORD)

Explanation:

Defines the duration of beep on touch in milliseconds.

Possible Values:

Default is 0x0014

Name:	BeepFreq
Data Type:	DWORD (REG_DWORD)

Explanation:

Frequency of beep on touch in Hz.

Possible Values:

Default is 0x00300

Drag Delay

Name:	DefaultDragDelay
Data Type:	DWORD (REG_DWORD)

Explanation:

Drag delay value in milliseconds.

Possible Values:

Default is 20.

Name:	MinDragDelay
Data Type:	DWORD (REG_DWORD)

Explanation:

Minimum drag delay value on the scale in Control Panel in milliseconds.
Reserved for use by control panel and driver.

Name:	MaxDragDelay
Data Type:	DWORD (REG_DWORD)

Explanation:

Maximum drag delay value on the scale in Control Panel in milliseconds.
Reserved for use by control panel and driver.

User Specific Registry Keys

User specific configuration data are saved in this key. User key is located in

HKEY_CURRENT_USER\Software\EloTouchscreen

Name:	DoubleClickHW
Data Type:	DWORD (REG_DWORD)

Explanation:

Defines the double click width and height.

Possible Values:

Default value is 25

Name:	EloTTray
Data Type:	DWORD (REG_DWORD)

Explanation:

This determines if Tool tray application should be run for this user when the user logs in.

Possible Values:

Default value is 1.

1: Run Elo tool tray app

2: Do not run Elo tool tray app.

Name:	RButton
Data Type:	Array of DWORDs(REG_BINARY)

Explanation:

Saves x,y Location of Right button application in Windows virtual coordinates.

Appendix B

Elo Driver Interface Usage

Introduction

EloIntf.dll provides a common interface for communicating with the Elo Serial and USB drivers. In order to, communicate with the either of the drivers load this dynamic library with your application and use the appropriate interface in your program.

All applications using this library must first call the EloIF_EnumTouchScreenEx function to initialize the DLL.

Please refer to the Appendix C, EnableDisable Sample Code for interface usage.

Elo Device Interface Functions

Function Name: EloIF_EnumTouchScreenEx

int EloIF_EnumTouchScreenEx(DWORD dwMonNum[32])

Parameters:
dwMonNum: Array of DWORD to receive the Windows monitor number associated with the touchscreens.

Return Values: Returns the total number of Elo touchscreens found. Returns EloFailure if no driver is found.

Remarks:
Enumerates and initializes all Elo Serial and USB devices.

This must be the first call made to the DLL to enumerate and initialize the touchscreens.

It returns the list of Windows monitor numbers associated with the touchscreens where, the index is the touchscreen number and the value is the Windows monitor number.

Touchscreens are 0 based and Windows monitor numbers are 1 based.

Maximum of 32 touchscreens are supported.

Function Name: EloIF_GetTouch

BOOL EloIF_GetTouch (int nScrNo,TOUCHPOINT* xy, BOOL xlated, GETPOINTS_CODE getCode)

Parameters:
nScrNo: 0 based touchscreen number.

xy: Pointer to TOUCHPOINT structure to receive the touch coordinates from the touchscreen. The application is responsible for allocating / freeing memory. Please see structure definition section for TOUCHPOINT.

xlated: If TRUE, the coordinates are returned translated for Windows coordinate system. If FALSE, raw coordinate data are returned.

getCode: This can be one of the following values: ReturnImmediately, ReturnOnTouch, ReturnOnUntouch. Please see definition for GETPOINTS_CODE.

Return Values: Returns TRUE if the call succeeds else it returns FALSE.

Remarks:
This is a blocking call used to get the touch coordinates for touch. This call does not return until user touches the screen specified.

To cancel the blocking wait use EloIF_Cancel API function.

Function Name: EloIF_Cancel

void EloIF_Cancel()

Parameters:
None.

Return Values: None.

Remarks:
Cancels any pending request with the driver. Typically used to cancel a blocking call made to the driver.

Function Name: EloIF_GetMouseMode

```
BOOL EloIF_GetMouseMode (WORD *wMode,UINT nScrNo)
```

Parameters:
wMode: Returns the touchscreen mode.

nScrNo: 0 based touchscreen number.

Return Values: Returns TRUE if the call succeeds else returns FALSE.

Remarks:
Retrieves the touch mode for the specified touchscreen.

Function Name: EloIF_SetMouseMode

```
BOOL EloIF_SetMouseMode (WORD wMode)
```

Parameters:
wMode: Touchscreen mode.

Return Values: Returns TRUE if the call succeeds else returns FALSE.

Remarks:
Sets the touch mode for all touchscreens to wMode.

Function Name: EloIF_GetTouchState

```
void EloIF_GetTouchState (BOOL *bFlag , UINT tsNumber)
```

Parameters:
bFlag : Touch state enabled / disabled.

tsNumber: 0 based touchscreen number.

Return Values: None.

Remarks:
Gets the touch state for touchscreen specified. It saves it in bFlag.
If bFlag is TRUE the touchscreen is disabled else it is enabled.

Function Name: EloIF_DisableTouchEx

```
void EloIF_DisableTouchEx (BOOL bFlag, int iScrNo)
```

Parameters:
bFlag : Touch state enabled / disabled.

iScrNo: 0 based touchscreen number.

Return Values: None

Remarks:
Enables or Disables touch depending on bFlag. If bFlag is TRUE the touchscreen is disabled else it is enabled.

If iScrNo is -1 then touch state is changed for all touchscreen.

Function Name: EloIF_GetDragDelay

BOOL EloIF_GetDragDelay (PDRAGDELAYDATA dwDragDelay,UINT nScrNo)

Parameters:
dwDragDelay: Pointer to DRAGDELAYDATA to receive Drag Delay . Please see structure definition section for PDRAGDELAYDATA.

nScrNo: 0 based touchscreen number.

Return Values: Returns TRUE if the call succeeds else returns FALSE.

Remarks:
Gets the dragdelay parameters for the specified touchscreen.

Function Name: EloIF_SetDragDelay

BOOL EloIF_SetDragDelay (DRAGDELAYDATA DragDelay)

Parameters:
DragDelay: Dragdelay data. Please see structure definition section for DRAGDELAYDATA.

Return Values: Returns TRUE if the call succeeds else returns FALSE.

Remarks:
Sets the drag delay for all touchscreens.

Function Name: EloIF_GetSound

BOOL EloIF_GetSound (PSOUNDDATA sndVal, UINT nScrNo)

Parameters:
sndVal: Pointer to SOUNDDATA structure to receive data. The application is responsible for allocating / freeing memory. Please see structure definition section for SOUNDDATA.

nScrNo: 0 based touchscreen number.

Return Values: Returns TRUE if the call succeeds else returns FALSE.

Remarks:
Gets the touchscreen beep data for the specified touchscreen.

Function Name: EloIF_SetSound

BOOL EloIF_SetSound (PSOUNDDATA sndVal)

Parameters:
sndVal: Pointer to sound structure to receive data. The application is responsible for allocating / freeing memory. Please see structure definition section for SOUNDDATA definition.

Return Values: Returns TRUE if the call succeeds else returns FALSE.

Remarks:
Sets the touchscreen beep data for all touchscreens.

Function Name: EloIF_GetVirtualFullScreen

BOOL EloIF_GetVirtualFullScreen (PFULLSCREEN pFullScreen)

Parameters:
pFullScreen: Pointer to FULLSCREEN structure containing full screen boundary data in pixels. The application is responsible for allocating / freeing memory. Please see structure definition section for FULLSCREEN.

Return Values: Returns TRUE if the call succeeds else returns FALSE.

Remarks:
Gets the full screen bounding rectangle and bounding mode for the specified touchscreen.

Function Name: EloIF_SetVirtualFullScreen

BOOL EloIF_SetVirtualFullScreen (PFULLSCREEN pFullScreen)

Parameters:
pFullScreen: Pointer to FULLSCREEN structure containing full screen boundary data in pixels. The application is responsible for allocating / freeing memory. Please see structure definition section for FULLSCREEN.

Return Values: Returns TRUE if the call succeeds else returns FALSE.

Remarks:
Sets the full screen bounding rectangle and bounding mode for the specified touchscreen.

Function Name: EloIF_GetCalibrationData

DWORD EloIF_GetCalibrationData (CalibData *pCalData,
DWORD dwScrNo)

Parameters:
pCalData: Pointer to CalibData structure to receive the calibration data. The application is responsible for allocating / freeing memory. Please see structure definition section for CalibData.

dwScrNo: 0 based touchscreen number.

Return Values: Returns TRUE if the call succeeds else returns FALSE.

Remarks:
Retrieves the calibration data for the specified touchscreen.

Function Name: EloIF_UpdateCalibrationData

BOOL EloIF_UpdateCalibrationData (CalibData
*pCalData,DWORD nScrNo)

Parameters:
pCalData: Pointer to CalibData structure which stores the calibration data for the touchscreen. The application is responsible for allocating / freeing memory. Please see structure definition section for CalibData.

nScrNo: 0 based touchscreen number.

Return Values: Returns TRUE if the call succeeds else returns FALSE.

Remarks:
Sets the calibration data for the specified touchscreen.

Function Name: EloIF_SwapButton

```
BOOL EloIF_SwapButton (HWND hWnd,DWORD
dwScrNo,DWORD dwCnt)
```

Parameters:
hWnd: Handle to window to receive the WM_RELEASEBUTTON message.

dwScrNo: 0 based touchscreen number.

dwCnt: Touch count for which the button should be swapped.

Return Values: Returns TRUE if the call succeeds else returns FALSE.

Remarks:
Swaps the button for the touch monitor for the touch count specified in dwCnt. Once the count is exhausted the DLL notifies the application, by sending WM_RELEASEBUTTON message to application window specified in hWnd.

WM_RELEASEBUTTON must be defined in the user application as Windows user defined message of value WM_USER + 4698.

Function Name: EloIF_SetLeftHandedMouse

```
int EloIF_SetLeftHandedMouse (BOOL bFlag)
```

Parameters:
bFlag: If TRUE it sets touchscreen mode to left-handed mouse.

Return Values: Returns EloSuccess if the call succeeds else returns EloFailure.

Remarks:
Sets the touchscreen mode to left-handed mouse. This does not affect the Windows left-handed mouse settings.

Function Name: EloIF_GetDiagnosticsData

```
BOOL EloIF_GetDiagnosticsData(LPPROPERTIESDATA pData,
int nScrNo)
```

Parameters:
pData: Pointer to PROPERTIESDATA structure to receive the diagnostics for the touchscreen. The application is responsible for allocating / freeing memory. Please see structure definition section for PROPERTIESDATA.

nScrNo: 0 based touchscreen number.

Return Values: Returns TRUE if the call succeeds else it returns FALSE.

Remarks:
Gets the diagnostics for the specified touchscreen. This information is also displayed in the Elo Control panel>Properties tab.

Function Name: EloIF_GetQuickTouch

```
int EloIF_GetQuickTouch (PQUICK_TOUCH_DATA pQTouch,
UINT nScrNo)
```

Parameters:
pQTouch: Pointer to QUICK_TOUCH_DATA structure to receive the quick touch configuration data from the selected touchscreen. The application is responsible for allocating / freeing memory. Please see structure definition section for QUICK_TOUCH_DATA.

nScrNo: 0 based touchscreen number.

Return Values: Returns EloSuccess if the call succeeds else returns EloFailure.

Remarks:
Gets the Quick Touch configuration parameters.

Function Name: EloIF_SetQuickTouch

```
int EloIF_SetQuickTouch (PQUICK_TOUCH_DATA pQTouch)
```

Parameters:
pQTouch: Pointer to QUICK_TOUCH_DATA structure containing the quick touch configuration data for the touchscreen. The application is responsible for allocating / freeing memory. Please see structure definition section for QUICK_TOUCH_DATA.

Return Values: Returns EloSuccess if the call succeeds else returns EloFailure.

Remarks:
Sets the Quick Touch configuration parameters for all touchscreens.

Function Name: EloIF_GetAcceleration

```
int EloIF_GetAcceleration(PACCELDATA pAccel)
```

Parameters:
pAccel: Pointer to ACCELDATA structure to receive the edge accleration data for the touchscreen. The application is responsible for allocating / freeing memory. Please see structure definition section for ACCELDATA.

Return Values: Returns EloSuccess if the call succeeds else returns EloFailure.

Remarks:
Gets the edge acceleration boundary and acceleration value for the given touchscreen.

Function Name: EloIF_SetAcceleration

```
int EloIF_SetAcceleration(PACCELDATA pAccel)
```

Parameters:
pAccel: Pointer to ACCELDATA structure to containing the edge accleration data for the touchscreen. The application is responsible for allocating / freeing memory. Please see structure definition section for ACCELDATA.

Return Values: Returns EloSuccess if the call succeeds else returns EloFailure.

Remarks:
Sets the edge acceleration boundary and acceleration scale for the given touchscreen.

Interface Data Structures

Structure Name: PROPERTIESDATA

The PROPERTIESDATA structure contains the diagnostics data for the touchscreen.

```
typedef struct _PropertiesData
{

        int iWindowsMonNo ;
        ULONG Type;
        char Port[32];
        char SerialNumber[18];
        DWORD HardwareHandshaking ;
        CONTRL_STAT ctrl_status;
        LONG BaudRate;
        char crevminor;
        char crevmajor;
        char trevminor;
        char trevmajor;
        char diagcodes[8];
        char id[8];
        char cnt_id[8];
        char driver_id[32];

}PROPERTIESDATA, *LPPROPERTIESDATA ;
```

Members

iWindowsMonNo: Windows monitor number associated with the touchscreen.

Type: Defines the type of touchscreen. Expected values are

USB: 0x01
PNP_SERIAL: 0x02
NT_SERIAL: 0x04
LEGACY_SERIAL: 0x08
RESERVED: 0xFF

Port:The serial port on which this serial touchscreen device is connected. This is blank for USB.

SerialNumber: Serial number for connected touchscreen. Serial numbers uniquely identify a touchscreen.

HardwareHandshaking: This is used only for serial touchscreens. If it is set to 1 hardware handshaking is turned on else if it is set to 0 it is turned off.

ctrl_status: Controller status returned at the time of diagnostics. Please see constants definition for CONTRL_STAT.

BaudRate: This specifies the baud rate for the serial port, valid only for serial touchscreens.

crevminor: Minor revision of controller.

crevmajor: Major revision of controller.

trevminor: Unused.

trevmajor: Unused.

diagcodes: The response received for the diagnostics smartest command sent to the controller.

id: Elo OEM identification returned from the controller.

cnt_id: Contains the response to controller id smartest command to the controller.

driver_id: Driver identification / version string.

Structure Name: TOUCHPOINT

The TOUCHPOINT structure defines a touch coordinate.

```
typedef struct tagTOUCHPOINT
{

          LONG x;
          LONG y;
          LONG z;
          }TOUCHPOINT, *LPTOUCHPOINT ;
```

Members
x: x co-ordinate for touch.

y: y co-ordinate for touch.

z: z co-ordinate for touch.

Structure Name: CalibData

The CalibData structure defines the calibration data used to convert touches from the Elo coordinate system to Windows virtual coordinate system.

The driver uses the following equation to convert a raw touch coordinate point from Elo coordinate system to the Windows screen coordinate system.
Calibration equation is

Xcal = a + m*Xuncal,

where,
Xuncal, is in Elo coordinate system
Xcal, is in Windows coordinate system
m = nScrDx / nEloDx
"a", is the X offset value entry
nScrDx = distance between targets in Windows virtual coordinates
nEloDx = distance between targets in Elo coordinates.

```
typedef struct _CalibData
{

        LONG VDeskMode ;
        LONG nScrDx ;
        LONG nEloDx ;
        LONG nOffsetX ;
        LONG nScrDy ;
        LONG nEloDy ;
        LONG nOffsetY ;
        LONG xyswap;
        LONG MonitorNumber ;
        RegistryOperation CalMode ;
        LONG xRes ;
        LONG yRes ;
        LONG xMonLocn ;
        LONG yMonLocn ;
        LONG xVirtScrSize ;
        LONG yVirtScrSize ;
        LONG xVirtScrCoord ;
        LONG yVirtScrCoord ;

} CalibData;
```

Members
VDeskMode: Defines the mode for the touchscreen boundary on the virtual desktop. Possible values are as follows:

0: Disable virtual desktop. Touch on all screens will reflect on the primary monitor.
1: Enable virtual desktop. In this mode no individual screen bounds are used for the monitor. Touches toward the edge may generate clicks on adjacent monitors.
2: Enable virtual desktop. Screen bounds are enabled for each monitor. Touches on the screen always generate clicks on the associated monitor. A touch outside the bounding rectangle will cause the cursor to move to a point along the boundary that is nearest to the point of touch.
3: Enable virtual desktop. Screen bounds are enabled for each monitor. Touches on the screen always generate clicks on the

associated monitor. Touches outside the bounding rectangle are not sent to the system.

nScrDx:
nEloDx:
nOffsetX:
nScrDy:
nEloDy:
nOffsetY:
Data required by calibration. Use the equation above to calculate these values.

xyswap: Specifies if the touchscreen is rotated 90 degrees or 270 degrees. Set to 1 if touchscreen is rotated 90 degrees or 270 degrees else set to 0.

MonitorNumber: Windows monitor numbers are 1 based.

CalMode: Set to TempWrite if the value does not have to be saved over system reboot. For values to be saved over system reboot set this value to Write. Please see constants definition for RegistryOperation.

xRes: Screen width in pixels.

yRes: Screen height in pixels.

xMonLocn: Monitor location (X) for this monitor on the virtual desktop.

yMonLocn: Monitor location (Y) for this monitor on the virtual desktop.

xVirtScrSize: Width in pixels of the windows virtual screen.

yVirtScrSize: Height in pixels of the windows virtual screen.

xVirtScrCoord: Top left X coordinate of the Windows virtual screen.

yVirtScrCoord: Top left Y coordinate of the Windows virtual screen.

Structure Name: DRAGDELAYDATA

The DRAGDELAYDATA structure is used to get or set drag delay.

```
typedef struct _DragDelayData{

          DWORD MinDragDelay ;
```

```
                    DWORD MaxDragDelay ;

                    DWORD DragDelay ;

} DRAGDELAYDATA, *PDRAGDELAYDATA ;
```

Members
MinDragDelay:
MaxDragDelay:
Reserved.

DragDelay: Drag delay value in milliseconds.

Structure Name: SOUNDDATA

The SOUNDDATA structure is used to get or set beep parameter data.

```
typedef struct _SoundData{

          BOOL BeepFlag;

          DWORD BeepFreq ;

          DWORD BeepTime ;

} SOUNDDATA, *PSOUNDDATA ;
```

Members
BeepFlag: If TRUE touchscreen beeps on touch.

BeepFreq: Frequency of beep in Hz.

BeepTime: Duration of beep in milliseconds.

Structure Name: FULLSCREEN

The FULLSCREEN structure defines the full screen bounding rectangle and bounding mode for the touchscreen.

```
typedef struct _FullScreen
```

```
{

        LONG ScreenNumber ;

        ClippingBounds Bounds;

        LONG ClippingMode;

} FULLSCREEN, *PFULLSCREEN ;
```

Members
ScreenNumber: 0 based touchscreen number.

Bounds: Defines the full screen bounding rectangle for the screen. See structure ClippingBounds. For interpretation of these bounds see ClippingMode below.

ClippingMode: Defines the mode for the touchscreen boundary on the virtual desktop. Possible values are as follows:
0: Disable virtual desktop. Touch on all screens will reflect on the primary monitor.
1: Enable virtual desktop. In this mode no individual screen bounds are used for the monitor. Touches toward the edge may generate clicks on adjacent monitors.
2: Enable virtual desktop. Screen bounds are enabled for each monitor. Touches on the screen always generate clicks on the associated monitor. A touch outside the bounding rectangle will cause the cursor to move to a point along the boundary that is nearest to the point of touch.
3: Enable virtual desktop. Screen bounds are enabled for each monitor. Touches on the screen always generate clicks on the associated monitor. Touches outside the bounding rectangle are not sent to the system.

Structure Name: QUICK_TOUCH_DATA

The QUICK_TOUCH_DATA data structure defines the Quick touch configuration parameters.

```
typedef struct _QuickTouchData

{

        DWORD bEnable ;
        ULONG Dx ;

        ULONG Dy ;

    } QUICK_TOUCH_DATA, *PQUICK_TOUCH_DATA;
```

Members
bEnable: 0 disables quick touch , 1 enables it.

Dx: X distance in pixels. Touches outside this distance will generate quick touches.

Dy: Y distance in pixels. Touches outside this distance will generate quick touches.

Structure Name: ClippingBounds

The ClippingBounds structure is used to define the bounding rectangle.

```
    typedef struct _ClippingBounds{

        ULONG X_Min ;

        ULONG Y_Min ;

        ULONG X_Max ;

        ULONG Y_Max ;

        ULONG Z_Max ;

        ULONG Z_Min ;

    } ClippingBounds, *PClippingBounds ;
```

Members
X_Min:
Specifies the X coordinate of the upper-left corner of the rectangle in pixels.

Y_Min:
Specifies the Y coordinate of the upper-left corner of the rectangle in pixels.

X_Max:
Specifies the X coordinate of the lower-right corner of the rectangle in pixels.

Y_Max:
Specifies the Y coordinate of the lower-right corner of the rectangle in pixels.

Z_Max:
Z_Min:
Reserved.

Constants: GETPOINTS_CODE

The blocked call returns data depending on this value.

```
typedef enum _GETPOINTS_CODE

{

        ReturnImmediately=1,

        ReturnOnTouch,

        ReturnOnUntouch

}GETPOINTS_CODE ;
```

Values
ReturnImmediately: Returns immediately with the touch data.

ReturnOnTouch: Waits for user to touch and then returns data.

ReturnOnUntouch: Waits for user to release touch and then returns data.

Constants: CONTRL_STAT

Controller status is returned as

```
typedef enum _CONTRL_STAT
```

```
          {
                    CS_OK = 0,
                    CS_ConstantTouch,
                    CS_CanNotFindController,
                    CS_NoResponse,
                    CS_InvalidResponse,
                    CS_CanNotSetBaudRate,
                    CS_CommandNotSent,
                    CS_SystemError,
                    CS_InvalidCommPort,
                    CS_CommPortFailedOpen,
                    CS_CommPortCommandError,
                    CS_CommPortNoController,
                    CS_UndefinedController
 } CONTRL_STAT;
```

Values

CS_OK: Everything works fine.

CS_ConstantTouch: There is constant touch detected on the touchscreen.

CS_CanNotFindController: No controller found.

CS_NoResponse: No response from controller.

CS_InvalidResponse: Incorrect response, maybe due to out of sync.

CS_CanNotSetBaudRate: Baud rate cannot be set.

CS_CommandNotSent: Smartset command could not be sent to the controller.

CS_SystemError: System Error.

CS_InvalidCommPort: No touchscreen connected on the serial port specified.

CS_CommPortFailedOpen: Error opening serial port.

CS_CommPortCommandError: Communications error.

CS_CommPortNoController: No controller.

CS_UndefinedController: Unidentified controller.

Constants: RegistryOperation

Driver writes the values to the registry if Write flag is set else it initializes the local buffer and does not save it to the registry.

```
typedef enum

{

        TempWrite=1,

        Write

}RegistryOperation;
```

Values

TempWrite: Writes data to local buffer. Does not write it to the registry.

Write: Writes data to local buffer and registry.

Error codes: Returned from the Interface DLL

EloSuccess: 0

EloFailure: -1

Appendix C
Enable/Disable Sample Code

```
/*-----------------------------------------------------------------------------
EnableDisable.cpp : Defines the entry point for the sample EnableDisable touch
            application.

Copyright(c) 2003  Elo TouchSystems, Inc., all rights reserved.

Sample application to Enable/Disable touch using Elo Interface DLL. This prgram can be
compiled using Microsoft Visual Studio 7.0 or 6.0.

Usage: EnableDisableSample -[e/d] -n:<mon num>[,<mon num>...]
                  -e    : Enable touch
                  -d    : Disable touch
                  -n:<mon num>[,<mon num>...] : Monitors to enable or disable
---------------------------------------------------------------------------- */


#include <windows.h>
#include <stdio.h>
#include <tchar.h>
#include <stdlib.h>

// ----------------------------------------------------------------------------
// Interface function calls
// ----------------------------------------------------------------------------
// Error codes
#define EloSuccess      0
#define EloFailure      -1
#define MAX_MON_SUPPORTED     32

typedef int (*ELOIF_ENUMTOUCHSCREENEX)(DWORD dwMonNo[MAX_MON_SUPPORTED]);
ELOIF_ENUMTOUCHSCREENEX EloIF_EnumTouchScreenEx ;

typedef  void (*ELOIF_DISABLETOUCHEX)( BOOL bFlag, int iScrNo ) ;
ELOIF_DISABLETOUCHEX EloIF_DisableTouchEx ;

BOOL bDisable = FALSE;
int dwMonParamCnt=0;
DWORD dwMonParam[MAX_MON_SUPPORTED] ;

HINSTANCE hInstInterfaceLib ;
int LoadInterfaceLibrary() ;
void ProcessCmdLine( int argc, char** argv) ;

// ----------------------------------------------------------------------------
// main entry point
int _tmain(int argc, char** argv)
{
      DWORD dwEnumMon[MAX_MON_SUPPORTED] ;

      if( argc < 3 )
      {
            printf( "Usage: EloEnableDisableSample -[e/d] \
                        -[n:<mon num>,[,<mon num>...]]\n\n \
                        -e \t\t\t\t: Enable touch\n \
                        -d \t\t\t\t: Disable Touch\n \
                        -n:<mon num>[,<mon num>...] \
                        \t: Monitors to enable or disable \n" ) ;
            return EloFailure ;
      }

      // Load the interface DLL
      if( LoadInterfaceLibrary() != 0 )
            return EloFailure ;

      ZeroMemory( dwEnumMon, MAX_MON_SUPPORTED ) ;
      ZeroMemory( dwMonParam, MAX_MON_SUPPORTED ) ;

      // Enumerates & initializes all Elo Serial & USB
      // The return value is the total number of Elo touch screens found.
      int iCnt = EloIF_EnumTouchScreenEx(dwEnumMon) ;
      if(iCnt < 1 )
      {
            printf( "No Elo touchscreens found\n" ) ;
            FreeLibrary(hInstInterfaceLib) ;
            return EloFailure;
      }

      // Process Commandline
      ProcessCmdLine( argc, argv ) ;

      // For all screens of matching monitor number enable / disable touch
      for( int i=0; i<dwMonParamCnt; i++ )
      {
            // where j is the screen number for the monitor number
            for( int j=0; j<iCnt; j++ )
            {
                  if( dwMonParam[i] == dwEnumMon[j] )
                  {
                        // Enables / Disables touch depending on the bFlag
                        // where iScrNo is the screen number
                        EloIF_DisableTouchEx( bDisable , j ) ;
                  }
            }
      }

      FreeLibrary(hInstInterfaceLib) ;
    return EloSuccess;
}

// ------------------------------------------------------------------------------
// Loads the Interface library
int LoadInterfaceLibrary()
{
	hInstInterfaceLib = LoadLibrary( "EloIntf" ) ;

	if( hInstInterfaceLib == NULL ) return EloFailure ;

	EloIF_EnumTouchScreenEx = (ELOIF_ENUMTOUCHSCREENEX)GetProcAddress(
	                        hInstInterfaceLib,"EloIF_EnumTouchScreenEx" ) ;

	if( EloIF_EnumTouchScreenEx == NULL )
	{
		FreeLibrary(hInstInterfaceLib) ;
		return EloFailure ;
	}

	EloIF_DisableTouchEx = (ELOIF_DISABLETOUCHEX)GetProcAddress(
	                              hInstInterfaceLib,"EloIF_DisableTouchEx" ) ;
	if( EloIF_DisableTouchEx == NULL )
	{
		FreeLibrary(hInstInterfaceLib) ;
		return EloFailure ;
	}

	return EloSuccess ;
}

// -----------------------------------------------------------------------------
// Processes command line options
void ProcessCmdLine( int argc, char** argv)
{
	char *lpArg, *lpArg2 ;
	char argstr[256]="";
	LPTSTR lpCmdLine = argstr ;
	strcpy( argstr,"" ) ;
	for( int i=0; i<argc ; i++ )
	{
		strcat( argstr," " ) ;
		strcat( argstr, argv[i] ) ;
	}

	lpArg = strstr(lpCmdLine,"-") ;

	while (lpArg != NULL)
	{
		switch (lpArg[1])
		{
			case 'e':
				// Enables / Disables touch depending on the Flag
				// FALSE to enable , TRUE to disable
				bDisable = FALSE;
				break;
			case 'd':
				// Enables / Disables touch depending on the Flag
				// FALSE to enable , TRUE to disable
				bDisable = TRUE;
				break;
			case 'n':
				// monitor number
				if (lpArg[2] == ':')
					lpArg++ ;
				lpArg2 = strstr(lpArg+1,"-") ;
				if (lpArg2 != NULL)
					lpArg2-- ;
```

```
                        dwMonParam[dwMonParamCnt] = atoi(lpArg+2) ;
                        if (dwMonParam[dwMonParamCnt] > 0)
                              dwMonParamCnt++ ;
                        lpArg = strstr(lpArg+1,",") ;

                        while(lpArg != 0)
                        {
                              if(lpArg2 && (lpArg >= lpArg2)) break ;
                              dwMonParam[dwMonParamCnt] = atoi(lpArg+1) ;
                              if (dwMonParam[dwMonParamCnt] > 0)
                                    dwMonParamCnt++ ;
                              lpArg = strstr(lpArg+1,",") ;
                        }
                        lpArg = lpArg2 ;
                        break ;
            default:
                  printf( "Usage: EloEnableDisableSample -[e/d] \
                        -[n:<mon num>,[,<mon num>...]]\n\n \
                        -e \t\t\t\t: Enable touch\n \
                        -d \t\t\t\t: Disable Touch\n \
                        -n:<mon num>[,<mon num>...] \
                        \t: Monitors to enable or disable \n") ;
                  exit(0);
                  break ;

            }
            if (lpArg != 0)
                  lpArg = strstr(lpArg+1,"-") ;
      }
}

// ------------------------------------------EOF---------------------------------
```

Contacting Elo

Elo has technical support offices around the world. By telephone, email or fax, you can usually find an office that is open and staffed with personnel to assist you with questions or problems with Elo drivers. Consult the list below for the office which can best serve you:

Americas

North America

U.S.A	Tel: 1-800-557-1458 (toll free) Fax: 865-694-1731 Online form: request technical support

Latin America

English-speaking customers	Tel: 1-800-557-1458 (**toll free**) Fax: 865-694-1731 Online form: request technical support
Portuguese-speaking customers	Tel: (55 11) 3611-1311, ext 249 or 144 Fax: (55 11) 3611-4365 E-mail: elotechbr@elotouch.com
Spanish-speaking customers	Tel: 54 11 - 47332238 Fax: 54 11 - 47332245 Online form: request technical support

Asia-Pacific

Japan	Tel: (81) (0) 45-478-2166 Fax: (81) (0) 45-478-2181 E-mail: info@tps.co.jp
Taiwan	Tel: 886 2 26629788, ext 416 Fax: 886 2 26642725 E-mail: eric.chang@tycoelectronics.com

Europe (including Africa/Middle East)

Africa	Tel: +32 (0)16 35 19 01
Belgium	Fax: +32 (0)16 35 21 01
Greece	E-mail: elotecheu@elotouch.com
Italy	
Luxemburg	
Netherlands	
Portugal	
Spain	

Austria	Tel: +49 (0)89 60 822 193
Czech Republic	Fax: +49 (0)89 60 822 180
Germany	E-mail: elotecheu@elotouch.com
Hungary	
Kroatia	
Poland	
Romania	
Russia	
Slovakia	
Slovenia	
Switzerland	

Denmark	Tel: +44 (0)1793 57 33 46
Finland	Fax: +44 (0)1793 57 33 45
Iceland	E-mail: elotecheu@elotouch.com
Israel	
Ireland	
Norway	
Sweden	
United Kingdom	
France	Tel: 0825 825 497 (toll free)
	E-mail: elotecheu@elotouch.com