

Welcome to UPDD version 4.1.x.

This version of the driver is a major update of version 4.0 and has vastly improved PnP integration, addresses the issue of cross-platform compatibility of the driver's utility programs and shares common code across all platforms:

The main features of the UPDD driver are:

- Vast majority of common code across all major OS
- Interface layers for mainstream OS - Windows 2000, XP, Vista, Linux and Mac OS X and CE
- Fully PnP compliant
- Support for non-pnp devices
- Windows Hardware Quality Labs (WHQL) approval to offer digitally signed versions with specific compliant hardware
- Embedded components for CE (all processors) and XPe
- Tested on many Linux distributions. Configurable install scripts to cater for variations across distributions
- Supports 100s of touch controllers, both legacy and current
- USB, Serial and PS/2 device support
- Unlimited number of touch devices supported on one system
- Supports multi-monitor environments
- Supports rotated devices at both software and video hardware rotate levels
- Multi-touch support in API to allow applications to interface with multi-touch devices
- Published Application Programming Interface for touch application integration
- Supports many languages with easy to use language support tools
- Touch surround capabilities (touch areas outside main calibrated area)
- Various calibration algorithms to cater for touch hardware variations
- Advance calibration to automatically assign desktop area to touch device
- Adaptable protocol engine to decode any conceivable data packet
- Selectable touch modes
- Network connection support

Over the coming months we will also be enabling a number of embedded touch utilities that have been written but are not sufficiently stable for release.

Not all version 4 programs are available in this release as it is necessary to release this driver to deliver support and functionality as it is implemented.

The intended UPDD suite of utility programs are as follows:

Program	Status	Description
User Console	Released	Driver's control program aimed at general, non technical, users
Calibration	Released	New calibration program
Daemon process	Released	Handles UPDD background tasks including UPDD System Tray functions under Windows.
Event Selector	Released	Utility to switch between two event states, normally Left and right click. Windows only.
Extensions	Under test	Touch utilities, to be released under 4.1.6
Activator	Under development	Quick access utility to touch related functions
Advanced Console	Under development	Advanced driver and device settings for technical users
SDK assistance	Under development	UPDD SDK API programmers guide and example code
Language files	First release	Common language files are under construction and will be release as available. A completed language file will allow all utility programs to be localized to the language of the system.

The installation program will install the driver, calibration program and the new UPDD Console. It will normally be delivered by email. Touch-Base utilises virus detection software on all of our systems but recipients of the software should pass the files through their own virus checking software before proceeding with installation.

## Software status

### Release software

This software is available in three build phases:

Phase	Description
Production	The latest production software offered as the last known good stable release
Alpha	Copy of production software undergoing changes
Beta	All alpha changes complete and undergoing final tests and acceptance before being placed into production

Alpha and beta releases are made available to test new functionality and by their very nature could be unstable due to the ongoing development.

### Evaluation software

If you are installing evaluation software then mouse clicks will be inactive after you have touched 100 times. A further 100 clicks are available after each reboot or calibration. This restriction is removed on licensed software.

## Operating system specific installations



The driver has been fully tested on Windows 2000, 2003, XP, Vista 32 and 64, Windows 64bit. For XPe you can install the desktop driver on a system running XPe or, if you are a system integrator building your own XPe image, you can embed UPDD components - see XPe entry below for further details.

Windows installation documentation is available [here](#).



For Mac OS X the installation program is delivered as a single file macx.sit. We are aware this is an old Mac OS X compressed format but we use a Windows system to create the drivers and it is currently easier to create .sit files than the new format in our Windows application but we hope in a future release to deliver in the new format – sorry for any inconvenience this may cause.

We are not currently able to create a universal driver that supports both Power PC and Intel processors so we are shipping separate drivers. Please make sure you have the appropriate install for the processor.

Mac OS X installation documentation is available [here](#).



Many Linux distributions have been tested and work fine as long as they conform to certain system requirements. However, we are aware of some Linux distributions that have some issues and these are discussed in detail in the Linux installation documentation. Should the driver not work on your Linux distribution please contact Touch-Base and we will investigate further – often it is subtle file structure differences.

Linux installation documentation is available [here](#).



For Windows CE UPDD the software can only be supplied in component form for Win CE 3.x through 5.x (.Net) with full embedding instructions.

Win CE embedded documentation is available [here](#).



Under Windows XPe the standard Windows installation files can be used on finalized XPe systems.

However, if you are creating an XPe system from embedded components UPDD can be supplied in component form.

Win XPe embedded documentation is available [here](#).



UPDD is available for VxWorks real-time operating system.

VxWorks Integration guide is available [here](#).

## General notes

The platform installation notes will cover any platform specific issues whereas the following notes cover general issues:

## Serial ports notes

### Serial to USB adaptors

If you are using a serial device via a serial to USB adaptor (needed if the computer system does not have a native serial port) make sure the adaptor has a relevant OS adaptor driver to create the virtual serial port in the system. We use adaptors from Keyspan. Before installing the UPDD serial driver you need to ensure you can identify the virtual serial port in the system.

### Changing port

Following installation the [UPDD Console – hardware dialog](#) can be used to change or define the serial port in use.

### Auto detection and Auto selection of serial controllers

Many serial devices can receive and respond to controller commands. These commands can be set up using [UPDD's macro language](#) and the macro can be invoked automatically as part of UPDD's controller initialisation procedure. See UPDD Advanced Console.

An example of a UPDD macro command is as follows:

```
[HEX]
01 55 FF
[ACK 1000]55 FF[END]
```

In this example UPDD will send 01 55 FF to the controller. If the controller is connected and functioning correctly it is expected to return 55 FF within 1000ms.

If a UPDD macro contains an [ACK] statement then the driver is able to automatically locate serially connected controllers. If an [ACK] statement has been defined for a controller then the Auto-Detect and [Auto-Select](#) will be available.

- **Auto-Detect** - If this function is enabled then all serial ports are tested to locate the controller. The defined serial port is the **first** port tested. This should only be selected if it is known that there are no other serially connected devices that will not be unduly affected by the receipt of the macro command sequence.
- **Auto-Select** - If this function is enabled then the defined serial port is the **only** port tested to locate the controller. This function is typically used when a customer may have more than one type of pointer device connected to the port and the driver is expected to automatically determine the type connected. In this scenario all possible controller types will be defined in UPDD but they will all be associated to the same port.

### Serial port testing

If a serial port device is not correctly working with the driver there are a number of procedures to follow to determine where the fault might lie as discussed in the knowledge base article [here](#).

## USB notes

For USB HID devices it is recommended that the device is first connected to the system prior to installation. This allows the HID driver to load and take control of the device. During the installation procedure UPDD will register as the controller's driver and take control of the device from HID. Once installed UPDD will then be associated with the device.

If, following installation, the device is not working it could be that the wrong driver is installed for the USB device in use. To identify the controller in use refer to the [Controller Identification](#) documentation.

## Calibration

Calibration is a procedure used to align the pointer device with the graphically display area or desktop segment. When using the pointer device the mouse cursor should normally position itself under the stylus when it is in contact with the pointer device. If this is not the case then calibration will be required and this is described in full in the [Calibration document](#).

The UPDD driver also supports Toolbars, which also require calibrating, and this is covered in full in a separate [Toolbar document](#).

## Driver settings – the UPDD Console

The driver and device settings can be adjusted with the UPDD Console program and is described in full in the [UPDD Console documentation](#).

## On-Line help



Help

The Help option can be used at any time to invoke the context sensitive on-line help. See the [Help document](#) for more information

## Toolbars

Toolbars are a concept unique to UPDD driver in that areas of the touch screen can be calibrated separately from the main calibrated area such that an application/function/action can be called when the toolbar is touched. See the [Toolbar documentation](#) for further details.

## Multi-monitor and multi-device support

Multi-monitor and multi pointer devices are supported with this driver and this functionality is covered in full in the [multi monitor and device document](#).

## Display rotation considerations

Most platforms offer some form of display rotation and UPDD caters for rotated displays where possible as explained in detail in the separate [rotate documentation](#).

## Mouse emulation

Most single pointer devices emulate the actions of the mouse; cursor movement, left click, right click, double clicks etc. These actions have to be derived from the triggers generated from the pointer device. In the case of a touch screen these triggers are initial touch, movement, stationary and last touch (you can also have pressure and dual touch (gesture) triggers).

The driver implements a number of different emulation methods to cater for the different ways mouse emulation is likely to be used with a single touch. These emulation methods are shown in the UPDD Console, Click Mode dialog.


Emulation mode	Initial touch	Movement	Stationary for short period	Last touch	Default Primary Event	Default Secondary Event	UPDD Version 3 reference
Click and drag	Pen down	Move with pen down	None	Pen up	Left click	Right click	Touchdown Left / Touchdown Right
Drag then click	None	Move	None	Pen down/up	Left click	Right click	Liftoff Left / Touchdown Right
Point and click	Pen down	off	None	Pen up	Left click	Right click	N/A
Interactive Touch	Left Pen down	Move with pen down	Left Pen up Right pen down / up	Pen up or no action	Fixed	N/A	Interactive Touch

Each touch device can have two mouse emulation modes associated with it, referred to as the Primary and Secondary emulation mode. When the stylus is used, it will, by default, perform its primary emulation. However, in some circumstances it is required to perform the associated secondary event and this is achieved by using the Event Selector (below) to switch between Primary and Secondary events. More detailed description of Mouse emulation is available [here](#).

## Event Selector

The Event selector is a facility that allows for user to indicate if the primary or secondary mouse emulation is to be performed on the next touch. This is normally used to toggle between Left and Right click emulation respectively.

### Under Windows it can be invoked in various ways:

- Single click the UPDD System Tray Utility  and select 'Event Selector' from the menu items;
- Directly calling the Event Selector from an application, desktop icon or, as shown below, the Windows Program Manager;



## Switching Events

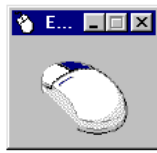
Clicking on either of the event selectors will toggle between the primary (left) and secondary (right) action. By default, the Event Selector automatically switches back to the primary setting after a single secondary action has been performed..

### From the desktop

#### Primary



#### Secondary



### From the system tray

#### Primary

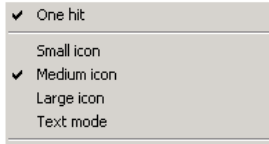


#### Secondary



## Event Selector Settings

Event Selector settings can be changed in the Event Selector menu. The menu can be viewed by clicking on the title bar mouse icon on the desktop Event Selector:



**One hit:** Indicates if the Event Selector automatically switches to the Primary setting after a single secondary use.

**Icon:** Indicates the size of the mouse icon displayed in the desktop event selector.

**Text mode:** Indicates that text be shown instead of a mouse icon.

## Under all Operating Systems:

The Event Selector function is implemented as a Toolbar Action. A Toolbar is created with an action of 'UPDD Action' and the Action being Event Selector. Toolbar images are available to represent the Primary and Secondary states (mouse with left click and mouse with right click).

Once the toolbar and toolbar images are defined the Event Selector will be shown on the screen and will toggle between the two event states:

#### Primary

#### Secondary



The Event Selector settings are defined in UPDD Extensions, Event Selector dialog. The main setting, One Hit, indicates if the Event Selector automatically switches to the Primary setting after a single secondary use.

The Event Selector can be moved by touching the Event Selector for a short period of time and moving it to the desired position.

More detailed description of Mouse emulation and the Event Selector is available [here](#).

## Controller specific notes

### 3M SCXXX range

The 3M range of controllers commonly known as the SC5 (SC40n, SC50n, SC80n) may need initialising before they generate the full range of touch coordinates. With un-initialised controllers, coordinates are not generated across the entire screen which can result in dead touch areas at the edges. If you are using these controllers and find that, after calibration, there is no reaction to touch around the edges then either use 3 point calibration in UPDD Calibration and check the EEPROM check box (this hardware initialises the controller and stores calibration in the controller's EEPROM) or run the Windows initialisation program available on the 3M web site at <http://www.3m.com/3MtouchSystems/downloads/drivers/HIDCalibUtility.zip> and then recalibrate with the UPDD calibration utility.

### OneTouch USB

At the time of writing (21<sup>st</sup> Aug 06) the OneTouch USB controller is actually an onboard Prolific PL2320 serial to USB adaptor so requires an appropriate driver to create a virtual serial port. For Mac OS X, OneTouch sent us file md\_pl2303H\_HX\_X\_dmg\_v1.1.ob1.zip for testing which worked fine. Once this driver is installed use the OneTouch UPDD serial driver to support the device.

### TouchKO

At the time of writing (9<sup>th</sup> Mar 07) the TouchKO USB controller requires virtual coms drivers for the SiLabs/ Cygnal CP2101 USB/Comm bridge to create a virtual serial port.

The CP210x USB to UART Bridge Virtual COM Port (VCP) drivers are required for device operation as a Virtual COM Port to facilitate host communication with CP210x products, available from

[http://www.silabs.com/tqwWebApp/public/web\\_content/products/Microcontrollers/USB/en/mcu\\_vcp.htm](http://www.silabs.com/tqwWebApp/public/web_content/products/Microcontrollers/USB/en/mcu_vcp.htm)

Once this driver is installed use the TouchKO UPDD serial driver to support the device.

### Zytronic X-Y

A UPDD Console firmware dialog is enabled if this controller is configured. If EEPROM storage is used the calibration data is stored in EEPROM and must be retrieved with the TBcalib command 'tbcalib eeprom' at system startup. This is mainly used for embedded systems. See [EEPROM documentation](#) for more details.

### ELO 2216 USB ELO 7010 USB

In USB mode the 2216 does not continue to send identical repeating coordinates and therefore coordinates stop if the stylus is held steady. In this situation UPDD will generate a pen up unless the lift off time processing is disabled. This is achieved by setting the lift off time to 0 in the UPDD Console, properties page.

### ELO 7000 USB

This is not SmartSet compatible and is the only ELO controller not supported by UPDD due to lack of technical documentation.

## Software Development Kit and Application Programming Interface

UPDD supports an application program interface (API) on all supported platforms. This allows user written programs to interact directly with the driver or pointer device handled by UPDD. It is assumed that the reader is familiar with the various functions and parameters of TBUPDD, since that information is not duplicated here.

The API calls work exactly the same in all environments except obvious operating system specific calls.

Registry get and set setting calls work exactly the same in all environments. In Windows they access the registry whereas in other operating systems they access the files holding the registry structure.

Depending on which operating system and client language is used the user has a choice of linking to the API statically or dynamically. For example on Windows both static and dynamic libraries are available, however Visual Basic only supports dynamic linking.

Depending on the OS in use the following files implement the UPDD API:



and



TBapi.h	function declarations.
TBbundle.h	contains bundle-specific data (file supplied with driver).
Tbapi.lib	statically linked implementation.
Tbapi.dll	dynamically linked implementation.
Tbapi.bas *	contains DLL function declarations, constants and data types and helper functions for VB.
Tbundle.bas *	contains VB bundle specific data.

\* both these files are required in a VB project



TBapi.h	function declarations.
TBbundle.h	contains bundle-specific data (file supplied with driver).
ITBApi_[proc].lib	statically linked implementation for a given target processor

[Proc] = x86, mips, arm etc



TBapi.h	function declarations.
TBbundle.h	contains bundle-specific data (file supplied with driver).
libTBApi.a	statically linked implementation.



TBapi.h	function declarations.
TBbundle.h	contains bundle-specific data (file supplied with driver).
libTBApi.so	dynamically linked implementation.

Java applications will need to use the Java Native Interface (JNI). This can link using either the DLL or the static library. A partial JNI interface is available, but not yet supported - this is supplied in source form to simplify JNI implementation.

Full details of the version 4 SDK and API are currently being documented in the advanced technical SDK dialog. Please contact Touch-Base if you require further details in the interim period.

### UPDD Interface utility

The UPDD calibration program exports a number of UPDD interface calls which can be called by an application program or Apple script which can be used to invoke certain UPDD driver functions without the need to directly use the UPDD API interface with your own application code. See [Calibration document -Interface calls](#) for more information.

### Software platform differences

Although it is our aim to make the software as compatible as possible it is not always possible to implement all functions across all platforms, either because of time / resource constraints, inappropriateness or incompatibility. Features that have not been implemented in a specific platform or missing functionality, as compared to the Windows driver, is documented in the platform specific notes.

### Support

If you encounter any difficulties with installation or usage of our software please review the [support document](#) and/or submit a support request via our on-line support form.

### Contact

For further information or technical assistance please email the technical support team at [technical@touch-base.com](mailto:technical@touch-base.com)