

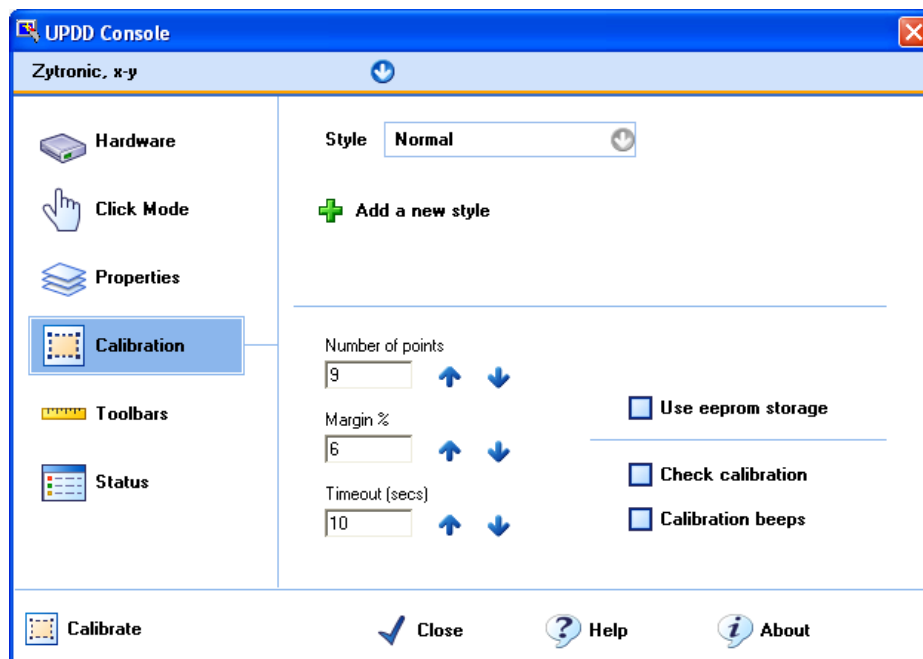
Some pointer device controllers have onboard EEPROM (Electrically Erasable Programmable Read Only Memory). EEPROM can be used to store information pertinent to the controller and also offers a memory storage area for applications and drivers to store data. One of the main uses for this memory with UPDD is to store the calibration data within the controller rather than locally on a system.

In all cases an absolute pointer device needs to be calibrated with the systems video display such that the point of contact is aligned with the video image. [UPDD calibration](#) is performed by drawing points on the video display at known locations that are then touched and the touch co-ordinates are noted for the given point of display. The driver can then use this data to scale and map the touch input to the video location. This information, known as the calibration data, needs to be stored for future reference, such that the device is calibrated at each system power up. This information can be stored locally on the system (e.g. the system's persistent memory, within a file, etc) or externally in the controller's EEPROM.

Given that UPDD supports 100's of different pointer devices, all with different characteristics, our driver generally stores its calibration data locally within the system.

However, there are cases where it is desirable to store calibration data externally. This is especially true of systems where the calibration data cannot be stored locally, such as some embedded system configurations where the entire system is locked down and any changes made are lost when the device is powered off. In other cases it is desirable to calibrate a touch screen prior to usage on a similar system so that it is calibrated when first used and will only require recalibrating if the EEPROM calibration is inaccurate.

As and when we have been requested to support EEPROM calibration for a given controller it has been added. When one of these controllers is defined in the UPDD build (and with [EEPROM enabled](#)) and added in the UPDD Console as an active device an EEPROM check box is shown on the UPDD Console, Calibration screen.



When this EEPROM check box is enabled (or set internally in the UPDD settings), EEPROM calibration storage is employed at the NEXT calibration. **It is important to note that this setting enables the EEPROM calibration procedure such that at the point of calibration the EEPROM calibration procedure is invoked. Depending on the controller in use there may be a further stage needed to retrieve this data at appropriate times, such as a system power up.**

EEPROM calibration falls into two distinct categories, one whereby UPDD stores its own calibration data in EEPROM and one whereby the controller's own firmware calibration procedure is invoked, more commonly known as a firmware calibration.

Where UPDD stores its own data then there is a requirement to retrieve the data at system start up / driver load.

Where firmware calibration is used the controller internally rescales the co-ordinate output to be calibrated with the video system and thus generated co-ordinate data is "pre-calibrated" and therefore a retrieval process is not relevant.

The implementation of each supported controller is discussed below.

### Supported EEPROM controllers

The following controllers are currently supported;

#### Firmware calibration

UPDD invokes the controller's firmware commands to perform calibration such that scaled touch co-ordinates are sent.

In most cases, the calibration function implemented within the controller's firmware dictates the number and location of the calibration points and this will be reflected within the available settings on the calibration dialog.

Following calibration the controller's output is adjusted to map to the video display.

#### DMC FIT 10

When enabled the calibration procedure is restricted to the range of calibration points supported by the controller's firmware calibration procedure.

#### 3M Touch Systems, SC400,500,800

UPDD invokes the controller's 3 point calibration procedure. In some older versions of the driver it is important to manually set the number of calibration points to 3 at the same time as enabling EEPROM.

This procedure is also required to initialise / linearise the controller and should be performed at least once irrespective of the need to subsequently utilise EEPROM functionality.

Note: Customers have reported that in some systems with video resolution 800 x 600 this procedure has failed but has worked fine with a 1024 x 768 resolution! Yet to be fully understood!

#### ELO TouchSystems, SmartSet USB

UPDD invokes the controller's 3 point calibration procedure. Internally it actually uses a 2 point algorithm with the 3<sup>rd</sup> point to determine orientation.

#### EEPROM storage of UPDD calibration data

UPDD calibration procedure stores its own calibration data in the controller's EEPROM. To utilize (retrieve) the calibration data stored in the controller the calibration program must be invoked, typically at startup, using the command 'tbcilib eeprom', as detailed in the [Calibration program document](#). If this is not performed then locally stored calibration data is used. Tbcilib returns an [error code](#) if the calibration data checksum is incorrect (e.g. retrieving calibration data from a controller that has no calibration data such as a new controller used for the first time). In locked down environments, such as embedded systems, it is important to retrieve the calibration data from the controller as any data stored locally is likely to be lost over a reboot, reverting back to the data built into the embedded image.

#### Zytronic, x-y

#### Hampshire Touch EEprom capable controllers

Due to current firmware limitations the data is written to the controller, 1 byte at a time, and depending on the number of calibration points can take a significant time to store and retrieve. 4 points takes approx 20 seconds.

#### Data Modul (derivative of eGalax / EETI controller)

#### Enabling EEprom protocols

Given that the EEprom interface to each controller is unique there is separate protocol code written for each controller. When we define a controller we identify the protocol to use. If the EEprom protocol setting is blank then the EEprom option will not be shown. If you are using a driver which supports one of the above controllers but does not show the EEprom option then it is likely the EEprom setting was/is not defined in the definition of the driver at build time. If this is the case contact Touch-Base to request an updated build.

The protocol setting protocol identifiers are as follows:

Controller	Interface	Protocol id 4.1.x only	Description	OS and UPDD version			
				Win	CE 5 and 6	Mac	Linux
DMC Fit 10	Serial	dmc-serial	Firmware calibration for scaled touch data	4.1.0	4.0.6	4.1.8	4.1.8
	USB	dmc-usb	Firmware calibration for scaled touch data	4.1.0	4.0.6	4.1.8	4.1.8
3M SCnnn	Serial	n/a	Hard coded - Firmware calibration for scaled touch data	TBA			
	USB	n/a	Hard coded - Firmware calibration for scaled touch data	TBA			
Zytronic	Serial	zytronic-serial	EEprom storage or UPDD data	4.1.1	4.0.6	4.1.8	4.1.8
	USB	zytronic-usb	EEprom storage or UPDD data	4.1.1	4.0.6	4.1.8	4.1.8
Hampshire	USB	tsharc-usb	EEprom storage of UPDD data	4.1.1	4.0.6	4.1.8	4.1.8
	USB	?	Under development - Firmware calibration for scaled touch data	4.1.8		4.1.8	4.1.8
ELO Smartset	USB	smartset-usb	Firmware calibration for scaled touch data	4.1.6	4.0.6	4.1.8	4.1.1
Data Modul	USB	EETI	EEprom storage or UPDD data	4.1.8	4.1.8	4.1.8	4.1.8

For UPDD 4.1.x the EEprom protocol id setting is held in the UPDD settings file, TBUPDD.INI, within the branch [updd\parameters\n] and called 'eeprom protocol'. Other EEprom settings within the file are redundant in 4.1.x and above.

For earlier UPDD versions the EEprom interface is hard coded and does not utilise the protocol id setting.

### **Adding EEPROM support**

We can add support in UPDD for any EEPROM enabled controller as required once supplied the technical documentation which describes the controller's EEPROM calibration interface.

If you are a controller manufacturer and would like to add EEPROM capabilities to you controller, we are aware of three methods that have been utilised as described below:

#### **Raw Mode**

Firmware commands are used to specify that raw blocks of data can be stored and retrieved from EEPROM. In this mode, UPDD simply stores and retrieves the calibration from the controllers EEPROM.

This is the simplest method to implement for both the firmware developers and our driver as the EEPROM read and write functions are already built into the UPDD calibration program.

#### **Formatted mode**

The controller's firmware dictates the format of the calibration data passed to the controller which is further processed by the controller. The co-ordinate data is then scaled and mapped to a grid co-ordinate range, based on the formatted data. In this case the calibration program has to gather calibration data as dictated by the data format requirement.

#### **Co-operative mode**

The controller's firmware exposes calibration commands such that the calibration program issues a firmware command for each calibration point displayed and the firmware captures the calibration points in real-time. The number of calibration points and the pattern used is dictated by the firmware's requirements. The co-ordinate data is then scaled and mapped to a grid co-ordinate range, based on the capture calibration information.

### **Contact**

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