

## UPDD Installation – Win CE embedded

### Software delivery

In this environment UPDD will be supplied as a number of separate components. Software sent via email will be held in the file ZIP file TBUPDDCE.ZIP. This avoids the rejection by many mail servers of .exe files.

### Overview

For OEMs requiring a touch screen, or other pointer interface on Windows CE devices, the Universal Pointer Device Driver suite of software includes a Windows CE 2.xx, 3.x and .net (4.x) embedded driver. This driver utilises the same code base as the Windows NT/9x/2K/XP UPDD product and so has all facilities found in UPDD on those systems, except for minor differences to accommodate variance in the Windows CE implementation. The UPDD application program interface is supported on Windows CE allowing 3<sup>rd</sup> party utilities to be developed.

Currently serial and x86-PS/2 interfaces are supported. Other interfaces such as ISA, USB etc will be added if required. The driver has been tested with X86 and StrongArm processors running CE 3.0 and CE .NET 4.1. Our CE.NET driver has been build in a CE.NET 4.1 development environment. At the time of writing we do not know if this is backward compatible to CE.NET 4.0. We can build drivers for other processors, as supported by the Microsoft CE Platform Builder, on request.

At the time of writing the processors supported by Win CE 2.xx/3.x/CE.NET are:

- MIPS
- StrongArm (also supports ARM processors)
- Power PC (Win CE 3.0/2.xx only)
- Hitachi SH3/SH4
- x86

A complete list of supported hardware is shown on the web at:

<http://www.microsoft.com/windows/Embedded/ce.NET/evaluation/hardware/processors.asp>

Target hardware may have to be supplied for testing if any problems are experienced with the driver.

### Embedding UPDD for Win CE

It is assumed that the developer is using Platform Builder 3.0 or greater and has created a Win CE image for the target hardware. This image should be created with the MAXALL configuration under CE 3.0 or an appropriate platform configuration under CE.NET.

To embed UPDD in a Windows CE image follow the following steps.

1. Copy the binary files (tbcilib.exe and tbupddce.dll) from the supplied zip file to the project tree.
2. Edit the configuration file project.bib, adding the following line:  
tbupddce.dll \$( \_FLATRELEASEDIR )\tbupddce.dll NK SH
3. If USB support is required edit the configuration file project.bib, adding the following line:  
tbupddceusb.dll \$( \_FLATRELEASEDIR )\tbupddceusb.dll NK SH
4. If manual calibration facilities are required, edit the configuration file project.bib, adding the following line:  
tbcilib.exe \$( \_FLATRELEASEDIR )\tbcilib.exe NK
5. Edit the configuration file project.reg, adding the contents of the supplied file tbupddce.reg.

6. If USB support is required edit the configuration file project.reg, manually adding the following entries:

```
[HKEY_LOCAL_MACHINE\Drivers\USB\LoadClients\VID_PID\Default\Default\Tbupddceusb]
```

```
"DLL"="Tbupddceusb.dll"
```

```
"Order"=dword:1
```

```
"Index"=dword:1
```

```
"Prefix"="USB"
```

with the VID and PID tokens being replaced with the appropriate values in base 10 (decimal). The VID and PID values can be found in the supplied tbupddce.reg in hexadecimal format.

```
"Product Id"=dword:0000HHHH
```

```
"Vendor Id"=dword:0000HHHH
```

**Important note.** The VID and PID values MUST match the VID and PID of the controller in use. If the supplied files do not match the controllers VID and PID, manually edit the values to match. If necessary, we can supply a Win 2000 utility to display the VID and PID when the controller is plugged in. e.g. Controller has hex VID = **1234** and PID = **5678** values. Hex 1234 = decimal **4660**. Hex 5678 = decimal **22136**.

Based on the above controller the settings would be as follows:

project.reg

```
[HKEY_LOCAL_MACHINE\Drivers\USB\LoadClients\4660_22136\Default\Default\Tbupddceusb]
```

tbupddce.reg

```
Product Id"=dword:00005678
```

```
"Vendor Id"=dword:00001234
```

7. Make any required software changes to the system components. (See "Port interface issues" below).
8. Rebuild the image using the Platform Builder.

## Calibration issues

The touch screen interface requires that calibration data be used to map screen touches to the corresponding Windows display area. This data is held in the registry. Under Windows CE the registry is reset to its default values whenever the device is reset. This raises some calibration issues that must be considered, these are especially relevant to x86 implementations where the device is reset upon suspend or power off.

UPDD calibration information can be determined in one of 3 ways:

**Auto-calibration.** The calibration information is calculated based on the maximum theoretical range of values (from the number of bits in the touch data packet) assuming that the available touch area is exactly the same size as the visible desktop area.

**Pre-calibration.** The calibration data is determined for a device – or class of device, and stored in the embedded configuration.

**Manual Calibration.** The user executes tbcilib.exe and touches a series of displayed points on the screen. Data recorded in this way is lost when the device is reset. An OEM using manual calibration needs to decide on a strategy for initiating the manual calibration. E.g. executing the calibration program at start-up, or placing an icon on the desktop. These and other options are implemented via the platform configuration.

## Calibration Settings

The supplied registry settings will usually be set to indicate auto-calibration. A CE image developer might wish to alter these default settings, for example to provide a pre-calibrated device. This is achieved by changing the settings copied to project.reg. The settings that affect calibration are stored under:

HKLM\Drivers\BuiltIn\TBUPDD\Parameters\{...}\1 - Where {...} is a GUID that identifies the package.  
The specific values used are: -

Number Of Calibration Points  
RefX0, RefX1, RefY0, RefY1 \*  
CalX0, CalX1, CalY0, CalY1\*  
InvertX, InvertY, SwapXY

\*If "Number Of Calibration Points" is greater than 2 then there will be correspondingly more of these values.

The above values may be changed in the following ways.

- **To force auto-calibration**

Number of Calibration Points=2

CalX0=CalX1=CalY0=CalY1=0

InvertX, InvertY, SwapXY=? (These values will have to be determined by experiment for a specific device. The valid values are 0 and 1 (for true / false)).

To define pre-calibration data:

The easiest way to determine the pre-calibration data is to execute tbcilib.exe on the device and perform a manual calibration. The values detailed above are noted from the registry on the target device and manually entered in to project.reg. NB when executing tbcilib.exe the mode of calibration is taken from a different registry location, see "manual calibration" below.

- **To prepare for manual calibration:**

A CE image developer might wish to alter aspects of the manual calibration, such as the number of points or the location of the points. In doing this it is important to bear in mind that the values outlined above are used for the active data (i.e. the previous calibration), and the data used for the next manual calibration are located at: -

HKLM\Drivers\BuiltIn\TBUPDD\Parameters\{...}\1\CalStyles\1

- **Determining calibration reference points:**

If defining pre-calibration data, or preparing for a manual calibration with a non-default number of points, the reference values must be set. The reference values represent the location of the calibration points based on a grid 0-65535 in the x and y planes, with the origin at the top left. So for a three-point calibration with points

At top left middle and bottom right the values would be: -

RefX0=0

RefY0=0

RefX1=32768

RefY1=32768

RefX2=65535

RefY2=65535

## Driver settings

The supplied registry settings will usually be set to the default settings for the controller in use. In a Windows desktop environment (e.g Win 98 etc) these settings are normally changed using the Driver Control Utility (DCU) but under Win CE these settings are set manually in the .reg file prior to generating the CE build. We trust the entries in the .reg structure are self-explanatory but please contact DMC to advise on driver settings changes if required. The active settings are found in the following registry branch:

HKLM\Drivers\BuiltIn\TBUPDD\Parameters\{\...}\1

## Port interface issues – important

### Serial

Touch-screens may be connected to a CE device via a standard serial (COM). A CE image builder should bear in mind that the default CE image generated by platform builder might well make assumptions regarding the usage of such ports. E.g. debug output will be sent to the first physical COM port, preventing its use. By default, CE creates two com port instances, Com 0 and Com 1. Com 0 is used as the debug port and relates to the physical port com 1. Com 1 is therefore the first port that can be used by the touch screen that actually relates to the physical Com 2 port.

Many customers have been unable to get their touch screens working with a default CE build until they have plugged the touch screen into com2 or changed the BIOS so that serial port is referenced as Com2, although the references in the CE build refer to com 1 !!!!

In some circumstances the CE builder will need to amend the CE configuration to alter the default serial port behavior. If you are not familiar with this procedure we have technical bulletin that covers this subject.

### USB

The CE image must be amended to support the USB host controller (this is the system's USB host controller and NOT the USB touch controller). Consult the manufacturer's documentation and or Platform Builder help for details of how to achieve this with the particular model of hardware in use. A 3rd party driver might be required for the host controller, although this has not the case for the hardware we have tested so far. To ensure the CE system's USB host controller is functioning use a HID mouse prior to testing the USB touch controller.

## Software required

In order to build a Win CE binary image the appropriate MS Platform Builder relating to the CE environment is required. E.g. MS Platform Builder 3.0 for Win CE 3.0 builds or Window CE.NET Platform for Windows CE.NET.

## Hardware required

A target Win CE device is required.

Alternatively x86 based images can be run on a standard PC using the LOADCEPC utility.

**Important note.** If you wish to use a Serial connection to the Windows CE device, please note that there is a consideration that is not always made clear in the Microsoft documentation. A null modem cable is required, but this differs from a standard null modem cable in that the RI pin is connected straight through. Without this connection it will be impossible to make a serial connection from the NT host to the Win CE device

## Contact

For further UPDD technical information please contact DMC's regional offices listed on our web site at <http://www.dmccoltd.com>

Document Version 1.3 April 4, 2003

(c) 2003 Touch-Base/DMC Co., Ltd