

PENMOUNT DEVICE DRIVER USERS' GUIDE FOR MICROSOFT WINDOWS CE

Version 2.3

30/Sep/'16



Preface

Disclaimer

The information in this document is subject to change without notice. The manufacturer makes no representations or warranties regarding the contents of this manual and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Furthermore, the manufacturer reserves the right to revise this publication or make changes in the specifications of the product described within it at any time without notice and without obligation to notify any person of such revision.

Trademarks

PenMount is a registered trademark of **SALT International Corp.** Microsoft and Windows are registered trademarks of Microsoft Corp. Other product names used in this manual are the properties of their respective owners and are acknowledged.

Copyright

This publication, including all photographs, illustrations and software, is protected under international copyright laws, with all rights reserved. Neither this manual, nor any of the material contained herein, may be reproduced without the express written consent of the manufacturer.

©Copyright 2016 **PenMount Touch Solutions.**

Revision Table

Date	Revision	Changes
16/Aug/2012	1.0	Initial Release.
13/Sep/2013	2.0	Content Layout Revised
17/Jul/2015	2.1	Add the BSP framework chapter.
20/Jan/2016	2.2	Textual refinement and editing Chapter3: Descriptions for PCI Utilities are maintained in separate document Chapter 4 :Add description for pmTchSdk_GetVersion().
03/Jun/2016	2.2A	Revise 1.1 Device Requirements
30/Sep/2016	2.3	Updated information of device driver V4.6

Table of Content

Preface.....	i
Disclaimer	i
Trademarks.....	i
Copyright	i
Revision Table.....	ii
1. Introduction.....	5
1.1 System Requirements	5
1.2 Device Requirements	5
2. Installation.....	7
2.1 Adding PenMount Support in Platform Builder	7
2.2 Configurations.....	9
2.2.1 Stylus Settings.....	9
2.2.2 RS-232 Interface Settings.....	10
2.2.3 I2C Interface Settings (V4 Only).....	12
3. Utilities.....	14
3.1 Touch Calibration	14
3.2 Draw	15
4. SDK.....	16
4.1 Summary	16
4.2 Get SDK Revision	17
4.3 Disable Touch Report	17
4.4 Enable Touch Report.....	18
4.5 Get Firmware Version	19
4.6 Adjust Panel Orientation.....	20
4.7 Get Edge Offset	20
4.8 Set Edge Offset.....	21
4.9 Reset Device.....	22
4.10 Get Touch Sensitivity.....	23

4.11	Set Touch Sensitivity	24
4.12	Update touch calibration data	24
5.	BSP Support	26
5.1	Initialize I2C Settings	26
5.2	Initialize Interrupt	26
5.3	Send Data to Device	27
5.4	Wait for Incoming Data	28
5.5	Get Data from Device.....	28

1. Introduction

This document provides information regarding using PenMount device drivers and utilities for Microsoft Windows CE and the touch API for programmers to control PenMount devices.

1.1 System Requirements

The PenMount device driver supports the following systems and hardware architectures.

Device Driver Versions	CE 5	CE 6	WEC 7	WEC2013
V3.3		O	O	
V3.4	O	O	O	
V4.2		O	O	
V4.4			O	O
V4.5		O	O	O
V4.6			O	O

1.2 Device Requirements

The PenMount device driver for Windows CE supports the following devices and interfaces:

Series	Product Name	USB	RS-232 / UART	I ² C ¹
PenMount P2-02 Series	PM1100		v	
	PM2101		v	v
PenMount P2-03 Series	PM1200	v	v	
	PM1201	v	v	v
	PM2201	v	v	v
PenMount P2-04 Series	PM1300A	v		
	PM1302	v	v	v
	PM1400A	v	v	
	PM1401	v	v	
	PM1401A	v		v
	PM1500	v	v	v
PenMount P2-06 Series	PM1110A		v	
	PM1210	v	v	v
	PM2103	v	v	v
	PM2203 PM2203B PM2203C	v	v	v
	PM2300	v	v	v
PenMount P2-08 Series	PM1310	v	v	
	PM1410	v	v	

¹ I2C interface support needs special customization using the pm-bsp introduced in chapter 5

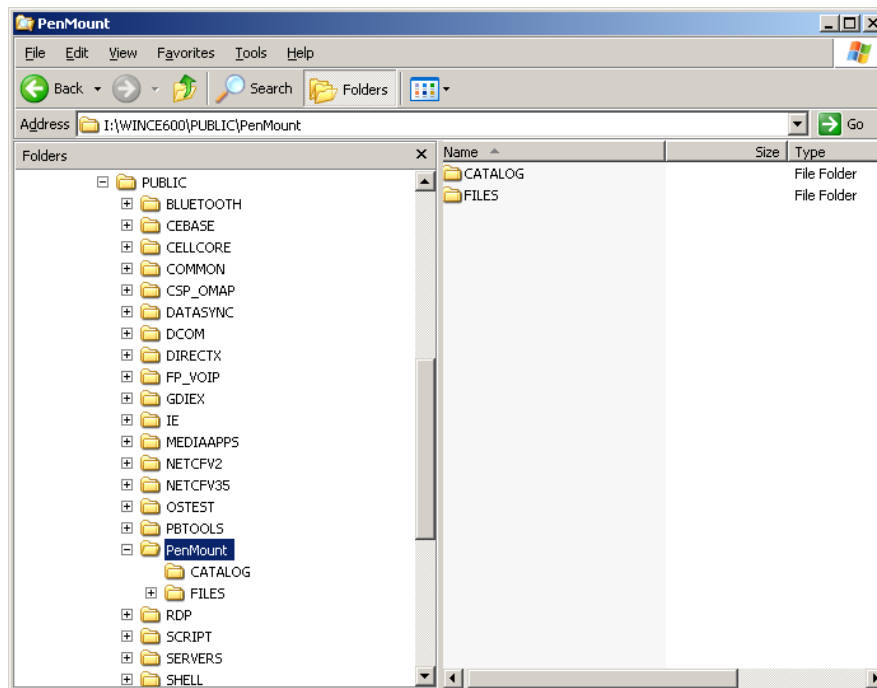
	PM1710	v	v	
	PM1711	v		v
PenMount 5000 Series	PM5126	v		v
	PM51A5	v		
PenMount 6000 Series	PM6200	v	v	
	PM6202	v	v	
	PM6300	v		
	PM6500	v	v	
	PM6005	v	v	
PenMount 9000 Series	PM9026		v	
	PM9036		v	

2. Installation

The PenMount device driver for Windows CE is organized in a folder named “PenMount”, which is a Windows CE Platform Builder component. System integrators need to manually add and configure the component before building a Windows CE system image.

Suppose that the Windows CE is installed to **%_WINCEROOT%**, please copy the PenMount folder and the files to the following location:

%_WINCEROOT% \ PUBLIC



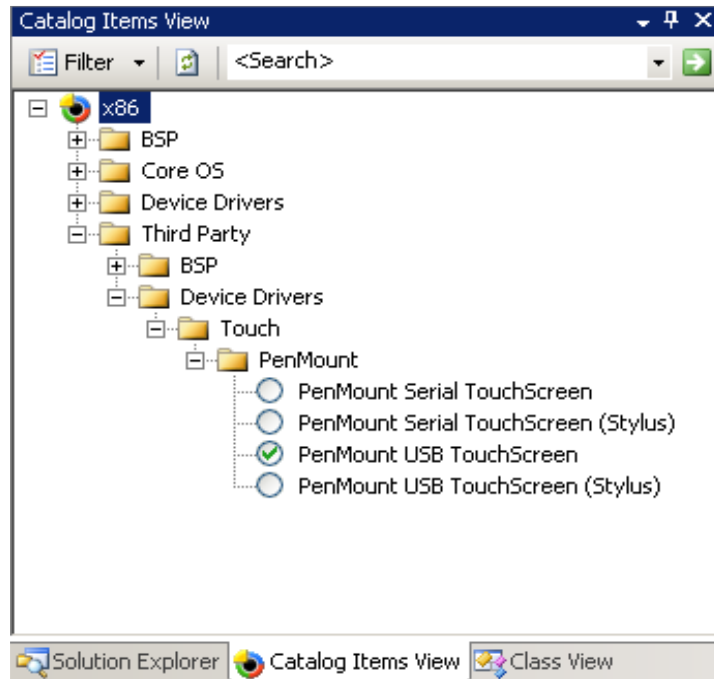
2.1 Adding PenMount Support in Platform Builder

If the PenMount device component is installed correctly, new items will be displayed in the “Catalog Items View”, in the following location:

Third Party > Device Driver > Touch > PenMount

- PenMount Windows CE Device Driver v3.4

The PenMount V3.4 device driver can run in two different modes: standard and stylus. System integrators will need to choose one of the items from the list.

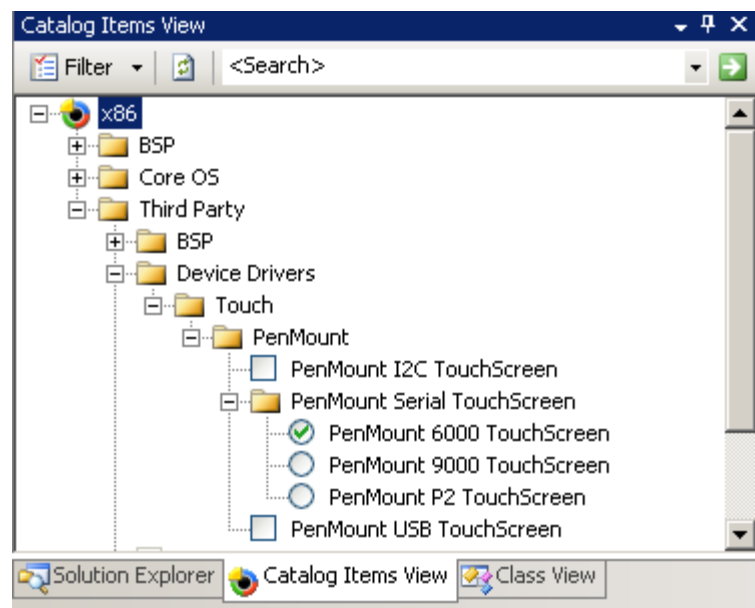


If standard mode is chosen, the touchscreen works as a standard mouse device.

If choosing stylus mode is chosen, the touchscreen can be used with the “Transcriber Handwriting Recognition Application” in Windows CE 5.0 and Windows Embedded CE 6.0.

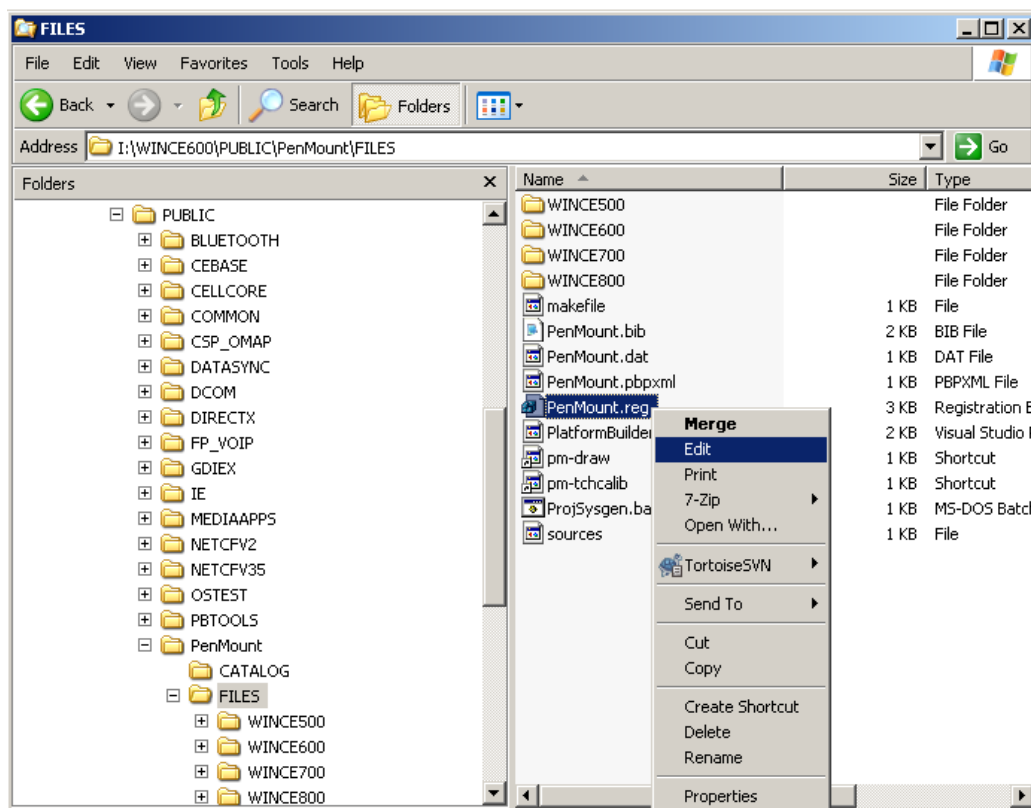
- PenMount Windows CE Device Driver v4

The PenMount V4 device driver runs in stylus mode only. System integrators can choose one more desired communication interface.



2.2 Configurations

Most device driver configurations can be adjusted by editing the [PenMount.reg](#) registry file in the following location.



2.2.1 Stylus Settings

The stylus settings are under the following registry key:

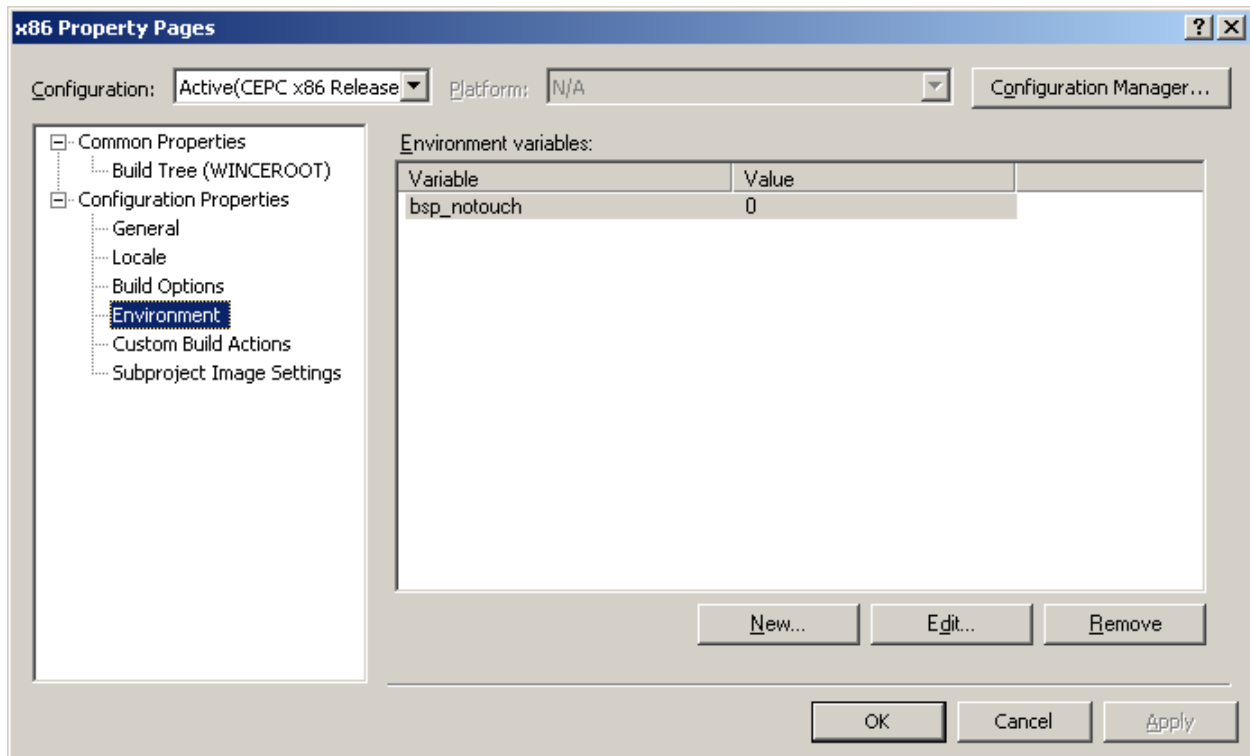
[HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\TOUCH]

Item	Description
DriverName	The touchscreen device driver name. Please do not change the value.
MaxCalError	The allowed offset between the calibrated touch coordinate and the expected location. In other words, touch calibration may not pass if the outcome exceeds this value. When using touch screens with lesser linearity, please consider adjusting this to a larger value.
CalibOffset	Controls the position of the calibration points, which is the offset from the screen edge. For example, when set to 5, the calibration points will be displayed at the 5%*screen width position. When set to 0, the calibration points will be displayed on the screen edge.
CalibMode	Valid calibration modes are 4 and 9.

CalibrationData

The raw data for each calibration point, which will be used to calculate the calibrated touch position. Please consider using the touch calibration utility to change this value.

When using a stylus, please also check if the OSDesign has “BSP_NOTOUCH” cleared. Stylus will not work when this value is set. System Integrators can manually clear this value in the OSDesign Property Page.



2.2.2 RS-232 Interface Settings

The RS-232 settings are slightly different when using different versions of the device driver.

- PenMount Windows CE Device Driver v3.4

Settings can be found under the following registry key:

[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\PMSEr]

Item	Description
Dll	The RS-232 device driver name, which is PMSEr.dll. Please do not change the value.
Prefix	The device name prefix. Please do not change the value.
Index	The device name index. Please do not change the value.
Order	The load order of the device driver.

	Please properly set this value so that it will be loaded only after the COM port device driver is loaded.								
Port	<p>The COM port number. For example, if PenMount device is attached to "COM1:", the value is 1.</p> <p>Please note that on some systems, one of the COM ports will be used for debugging, and the port number in Windows CE might not match its original number in BIOS.</p> <p>For example, "COM1:" might be the "COM2" in BIOS.</p>								
IClass	The device class. Please do not change the value.								
Protocol	<p>The predefined index of the PenMount device protocol. The device driver will not detect the device model dynamically, so system integrators need to make sure that this is set to the correct value.</p> <table> <tr> <th>Value</th><th>Protocol</th></tr> <tr> <td>1</td><td>PenMount 9000 Controller</td></tr> <tr> <td>3</td><td>PenMount 6000 Controller</td></tr> <tr> <td>5</td><td>PenMount PCI Controller</td></tr> </table>	Value	Protocol	1	PenMount 9000 Controller	3	PenMount 6000 Controller	5	PenMount PCI Controller
Value	Protocol								
1	PenMount 9000 Controller								
3	PenMount 6000 Controller								
5	PenMount PCI Controller								
Baudrate	<p>The baud rate used for communication. Please set to one of the following values.</p> <table> <tr> <th>Value</th><th>Protocol</th></tr> <tr> <td>2580</td><td>9600 bps</td></tr> <tr> <td>4800</td><td>19200 bps</td></tr> <tr> <td>9600</td><td>38400 bps</td></tr> </table>	Value	Protocol	2580	9600 bps	4800	19200 bps	9600	38400 bps
Value	Protocol								
2580	9600 bps								
4800	19200 bps								
9600	38400 bps								
HideCursor	(For standard mode only) Hide the mouse cursor								
DisableEEPROM	Disable loading or saving calibration data to the non-volatile memory on device								

- PenMount Windows CE Device Driver v4

In PenMount Windows CE V4, there is a universal device driver for all communication interfaces and its settings can be found under the following registry key:

[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\Touch]

Item	Description
Dll	The touch device driver name. Please do not change the value.
Prefix	The device name prefix. Please do not change the value.
Index	The device name index. Please do not change the value.
Order	<p>The load order of the device driver.</p> <p>Please properly set this value so that it will be loaded only after the COM port device driver is loaded.</p>
IClass	The device class. Please do not change the value.
ReportMode	<p>0: (default) Stylus mode.</p> <p>1: Mouse report mode, single touch only. Default value when using PenMount 9000.</p>
EnableTX	0: RS-232 TX is disabled, no command will be sent with RS-232 interface. Please

use this if EEPROM support is not needed.

1: (default) RS-232 TX is disabled, command will be sent with RS-232 interface.

If the PenMount device is attached to “COM1:”, please change settings in the following registry key. If the device is attached to other COM ports, please change “COM1” to a proper value.

[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\Touch\COM1]

Item	Description								
Model	The PenMount device model. The device driver will not detect the device model dynamically, so system integrators need to make sure that this is set to the correct value.								
	<table><tr><th>Value</th><th>Protocol</th></tr><tr><td>3500</td><td>PenMount PCI Controller</td></tr><tr><td>6000</td><td>PenMount 6000 Controller</td></tr><tr><td>9000</td><td>PenMount 9000 Controller</td></tr></table>	Value	Protocol	3500	PenMount PCI Controller	6000	PenMount 6000 Controller	9000	PenMount 9000 Controller
	Value	Protocol							
	3500	PenMount PCI Controller							
	6000	PenMount 6000 Controller							
9000	PenMount 9000 Controller								
Baudrate	The baud rate used for communication. Please set to one of the following values.								
	<table><tr><th>Value</th><th>Protocol</th></tr><tr><td>2580</td><td>9600 bps</td></tr><tr><td>4800</td><td>19200 bps</td></tr><tr><td>9600</td><td>38400 bps</td></tr></table>	Value	Protocol	2580	9600 bps	4800	19200 bps	9600	38400 bps
	Value	Protocol							
	2580	9600 bps							
4800	19200 bps								
9600	38400 bps								

2.2.3 I2C Interface Settings (V4 Only)

If the PenMount device is attached to I2C2, please change settings in the following registry key. If the device is attached to other I2C buses, please change “I2C2” to a proper value.

[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\Touch\I2C2]

Item	Description				
Model	The PenMount device model. The device driver will not detect the device model dynamically, so system integrators need to make sure that this is set to the correct value.				
	<table><tr><th>Value</th><th>Protocol</th></tr><tr><td>3500</td><td>PenMount PCI Controller</td></tr></table>	Value	Protocol	3500	PenMount PCI Controller
	Value	Protocol			
3500	PenMount PCI Controller				
SlaveAddr	The slave address of the PenMount device. By default, the value is 0x38.				

- The pm-bsp File

I²C interface communication is sometimes hardware dependent and the PenMount device driver keeps these codes in the “pm-bsp.dll” file which can be found in the following directory:

PenMount\FILES\\$_(_WINCEOSVER)\\$_(_TGTCPU)

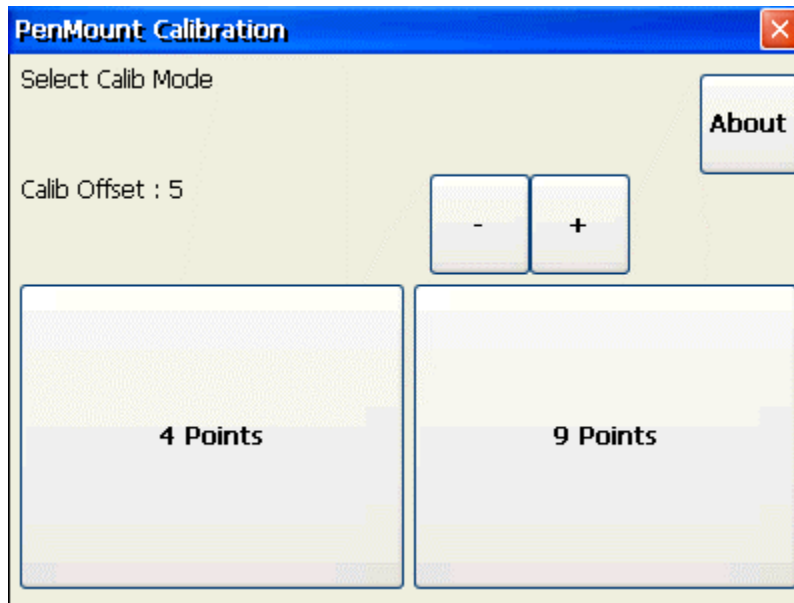
The default “pm-bsp.dll” in the device driver package does not provide any function, so system integrators must implement their own “pm-bsp.dll” to support I²C communications on their hardware platform. Please reference the “BSP Framework” chapter for more information.

3. Utilities

PenMount provides several utilities for adjusting and testing touch functionality. Some of these utilities, such as calibration and draw, are included in the PenMount device driver package. For other utilities, please contact PenMount for availability.

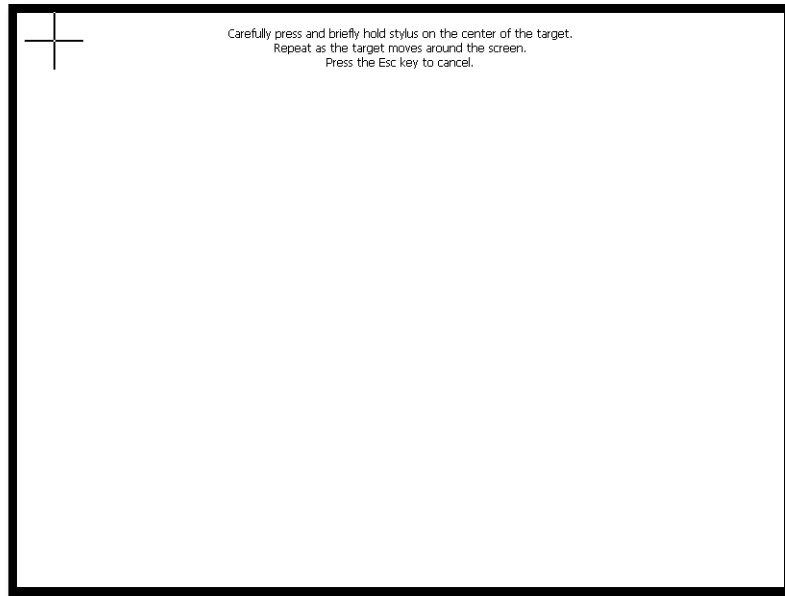
3.1 Touch Calibration

The PenMount Touch calibration utility is designed to calibrate resistive touchscreens and can also be used to check version information.



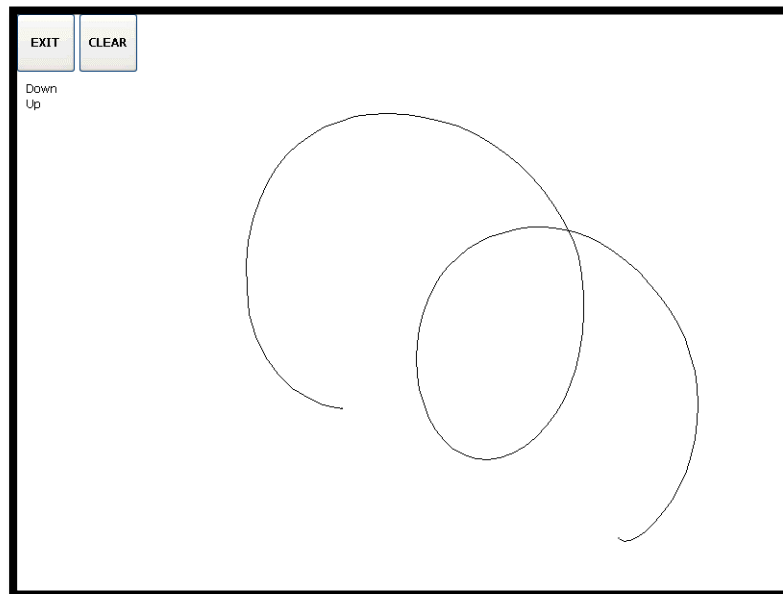
- **About**
The About button shows the device driver and controller versions. If no device is connected, the device version will not be shown.
- **Calib Offset**
Please click on the “-” or “+” button to dynamically change the calibration offset value. The minimum value is 0, and the maximum value is 15.
- **Calib Mode**

Please click on the “4 Points” or “9 Points” button to select the calibration mode. This will also launch the calibration UI. Please press and hold the cross until it moves to the next position.



3.2 Draw

The PenMount draw utility can be used for touch function testing.



4. SDK

The SDK provides API for programmers to gain touch control.

4.1 Summary

The SDK comes with the following files used during compile time and runtime.

1. Include

Please include the following file in your code.

[inc/pm-tchsdk.h](#)

2. Library

Please link to the following file during linking.

[lib/%_TGTCPU%/pm-tchsdk.lib](#)

3. RunTime

Please put the following file in the same location as your program.

[lib/%_TGTCPU%/pm-tchsdk.dll](#)

4. Sample

● Disable Sample

This sample demonstrates how to use API to disable touch report, and enable touch report. The code can be found in the following location:

[sample\Disable](#)

● GetVersion Sample

This sample demonstrates how to use API to get firmware version. The code can be found in the following location:

[sample\GetVersion](#)

● Rotation Sample

This sample demonstrates how to use API to adjust panel orientation. The code can be found in the following location:

[sample\Rotation](#)

● Sensitivity Sample

This sample demonstrates how to use API to adjust touch sensitivity. The code can be found in the following location:

[sample\Sensitivity](#)

- **UpdateCalib Sample**

This sample demonstrates how to use API to update calibration data dynamically.

[Sample\UpdateCalib](#)

4.2 Get SDK Revision

Syntax

```
int pmTchSdk_Revision ( int * major_version,  
                        int * minor_version,  
                        int * build_version )
```

Description

This function retrieves the revision of the SDK.

Parameters

major_version: Pointer to the memory that will store the SDK major version.

minor_version: Pointer to the memory that will store the SDK minor version.

build_version: Pointer to the memory that will store the SDK build version.

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Requirements

Device	Driver V3.3	Driver V3.4	Driver V4
PenMount 6000 USB	O	O	O
PenMount P2 USB	O	O	O
PenMount 5000	O	O	O
PenMount 9000	O	O	O
PenMount 6000 RS-232	O	O	O
PenMount P2 RS-232	O	O	O

4.3 Disable Touch Report

Syntax

```
int pmTchSdk_DisableTouch ( void )
```

Description

This function disables touch reports of the PenMount controller.

Parameters

none

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Requirements

Device	Driver V3.3	Driver V3.4	Driver V4
PenMount 6000 USB	O	O	O
PenMount P2 USB		O	O
PenMount 5000			
PenMount 9000			O
PenMount 6000 RS-232			O
PenMount P2 RS-232			O

4.4 Enable Touch Report

Syntax

```
int pmTchSdk_EnableTouch ( void )
```

Description

This function enables touch reports of the PenMount controller. When device resets, the touch report is enabled by default.

Parameters

none

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Requirements

Device	Driver V3.3	Driver V3.4	Driver V4
PenMount 6000 USB	O	O	O
PenMount P2 USB		O	O
PenMount 5000			
PenMount 9000			
PenMount 6000 RS-232			O
PenMount P2 RS-232			O

4.5 Get Firmware Version

Syntax

```
int pmTchSdk_GetVersion (int Version[5])
```

Description

This function gets the firmware version of the attached PenMount device.

Parameters

Version: Pointer to an int type of array buffer that will store the firmware information after function succeeds.

Index	Description	Example: 1410.D01.2.1.3
Version[0]	Product Number	1410
Version[1]	Major Version	2
Version[2]	Minor Version	1
Version[3]	Build Version	3
Version[4]	ODM Version	1

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Requirements

Device	Driver V3.3	Driver V3.4	Driver V4
PenMount 6000 USB		O	O
PenMount P2 USB		O	O
PenMount 5000			
PenMount 9000			O
PenMount 6000 RS-232			O

4.6 Adjust Panel Orientation

Syntax

```
int pmTchSdk_SetRotation ( int Degree )
```

Description

This function changes touch reports.

Parameters

Degree: The orientation of the touch panel.

0: The orientation is the natural orientation of the display device; it should be used as the default.

90: The orientation is rotated 90 degrees (measured counterclockwise) from 0.

180: The orientation is rotated 180 degrees (measured counterclockwise).

270: The orientation is rotated 270 degrees (measured counterclockwise).

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Requirements

Device	Driver V3.3	Driver V3.4	Driver V4
PenMount 6000 USB			
PenMount P2 USB		O	O
PenMount 5000			
PenMount 9000			
PenMount 6000 RS-232			
PenMount P2 RS-232			O

4.7 Get Edge Offset

Syntax

```
int pmTchSdk_GetEdgeOffset ( RECT * pPanelRect )
```

Description

This function retrieves the current edge offset settings that PenMount controller is using. The value range is from 0 to 7.

Parameters

pPanelRect: A pointer to a RECT structure that contains the current offset settings of each edge.

pPanelRect->left : The offset of left edge.

pPanelRect->right : The offset of right edge.

pPanelRect->top : The offset of top edge.

pPanelRect->bottom : The offset of bottom edge.

Please note that the edge settings are in 0 degrees.

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Requirements

Device	Driver V3.3	Driver V3.4	Driver V4
PenMount 6000 USB			
PenMount P2 USB		O	O
PenMount 5000			
PenMount 9000			
PenMount 6000 RS-232			
PenMount P2 RS-232			O

4.8 Set Edge Offset

Syntax

```
int pmTchSdk_SetEdgeOffset ( RECT * pPanelRect )
```

Description

This function sets the current edge offset settings that PenMount controller uses. The valid value range is from 0 to 7.

Larger offset values make detecting touch on panel edges easier, but it also slightly decreases touch accuracy.

Parameters

pPanelRect: A pointer to a RECT structure of the new offset settings of each edge.

pPanelRect->left : The offset of left edge.

pPanelRect->right : The offset of right edge.

pPanelRect->top : The offset of top edge.

pPanelRect->bottom : The offset of bottom edge.

Please notice that the edge settings are in 0 degrees.

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Driver Requirements

Requirements

Device	Driver V3.3	Driver V3.4	Driver V4
PenMount 6000 USB			
PenMount P2 USB		O	O
PenMount 5000			
PenMount 9000			
PenMount 6000 RS-232			
PenMount P2 RS-232			O

4.9 Reset Device

Syntax

```
int pmTchSdk_ResetDevice ( void )
```

Description

This function enables manual reset of the PenMount controller.

Parameters

none

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Requirements

Device	Driver V3.3	Driver V3.4	Driver V4
PenMount 6000 USB			
PenMount P2 USB		O	O
PenMount 5000			
PenMount 9000			
PenMount 6000 RS-232			
PenMount P2 RS-232			O

4.10 Get Touch Sensitivity

Syntax

```
int pmTchSdk_GetSensitivity ( int * pSensitivity )
```

Description

This function gets the touch sensitivity of the PenMount controller.

Parameters

pSensitivity: A pointer to an int value of the new touch sensitivity.

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Requirements

Device	Driver V3.3	Driver V3.4	Driver V4
PenMount 6000 USB			
PenMount P2 USB		O	O
PenMount 5000			

PenMount 9000	
PenMount 6000 RS-232	
PenMount P2 RS-232	0

4.11 Set Touch Sensitivity

Syntax

```
int pmTchSdk_SetSensitivity ( int sensitivity )
```

Description

This function enables manual reset of the PenMount controller.

To ensure that touch works properly, please manually call the `pmTchSdk_ResetDevice()` function after setting the sensitivity.

Parameters

sensitivity: The new touch sensitivity. The valid value range is 1 to 15.

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Requirements

Device	Driver V3.3	Driver V3.4	Driver V4
PenMount 6000 USB			
PenMount P2 USB		0	0
PenMount 5000			
PenMount 9000			
PenMount 6000 RS-232			
PenMount P2 RS-232			0

4.12 Update touch calibration data

Syntax

```
int pmTchSdk_UpdateCalibData ( void )
```

Description

This function reloads the calibration data from registry and updates the touch driver.

The calibration data is read from the following registry.

[HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\TOUCH]
"CalibrationData" = "2048,2048 205,205 3891,205 3891,3891 205,3891"

Parameters

none

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Requirements

Device	Driver V3.3	Driver V3.4	Driver V4
PenMount 6000 USB			O
PenMount P2 USB			O
PenMount 5000			O
PenMount 9000			O
PenMount 6000 RS-232			O
PenMount P2 RS-232			O

5. BSP Support

PenMount Windows CE Device Driver V4 supports I2C interface. However, since I2C configurations are different on each hardware platform, developers need manually configure the I2C pins and build the “pm-bsp.dll” file, which will be loaded by device driver at run time.

The codes can be found in the following location.

```
PenMount\SDK\SAMPLES\pm-bsp
```

5.1 Initialize I2C Settings

Syntax

```
HANDLE pmBSP_I2C_InitDevice (TCHAR * szDevName)
```

Description

This function opens the I²C device for latter I/O operations. Other initializations should also be performed here.

Parameters

szDevName

The I²C device name, specified by the I²C registry key. For example,

[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\Touch\I2C2] uses I2C2 as szDevName.

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Driver Requirements

PenMount Windows CE Driver V4.5

5.2 Initialize Interrupt

Syntax

```
DWORD pmBSP_I2C_InitIntr ( void )
```

Description

(Optional) This function is implemented to initialize a GPIO to interrupt mode and can be a falling edge trigger. Using a low level trigger might cause the system to hang since the INT line is always low when data is available and could make the interrupt service routine busy.

If not using interrupt pin, please return SYSINTR_UNDEFINED.

Parameters

None

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Driver Requirements

PenMount Windows CE Driver V4.5

5.3 Send Data to Device

Syntax

```
int pmBSP_I2C_SendData ( HANDLE hDevice,  
    unsigned char SlaveAddr,  
    unsigned char * pBuffer ,  
    unsigned long dwLength )
```

Description

This function sends command to I²C device. Please implement this function with the API provided by BSP.

Parameters

hDevice

The handle returned by pmBSP_I2C_InitDevice()

SlaveAddr

The slave address used by device, also set in registry

[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\Touch\I2C2]

"SlaveAddr" = dword:38

The default slave address for PenMount is 0x38.

pBuffer

A preallocated buffer for storing data to be sent to device.

dwLength

The size of data to be sent to device. For PenMount P2, the size is 6 bytes.

Return Value

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero.

Driver Requirements

PenMount Windows CE Driver V4.5

5.4 Wait for Incoming Data

Syntax

```
void pmBSP_I2C_WaitData ( HANDLE hWaitEvent )
```

Description

(Optional) This function is implemented to await data from device. If interrupt is not used, this function will not be called.

Parameters

None

Return Value

None

Driver Requirements

PenMount Windows CE Driver V4.5

5.5 Get Data from Device

Syntax

```
int pmBSP_I2C_GetData ( HANDLE hDevice,  
    unsigned char SlaveAddr,  
    unsigned char * pBuffer ,  
    unsigned long dwLength )
```

Description

This function reads data from I²C device. Please implement this function with the API provided by BSP.

Parameters

hDevice

The handle returned by pmBSP_I2C_InitDevice()

SlaveAddr

The slave address used by device, also set in registry

[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\Touch\I2C2]

"SlaveAddr" = dword:38

The default slave address for PenMount is 0x38.

pBuffer

A preallocated buffer for storing data read from device.

dwLength

The data size to be read from device. For PenMount P2, the size is 6 bytes.

Return Value

If the function succeeds, the return value is the actual data bytes read.

If the function fails, the return value is zero.

Driver Requirements

PenMount Windows CE Driver V4.5