Fujitsu Touch Panel (Serial)
Device driver setting manual

For Windows95/98/Me/NT4.0/2000/XP
V1.0L27

General

This manual explains how to make setting of touch panel driver. Please refer readme.txt of each driver set for installation.

1. Setting of driver

1.1 Start of setting screen

After touch panel driver is installed, "Touch panel" icon of the figure shown below is registered in the control panel. A setting tool starts when this icon on the control panel is double-clicked.

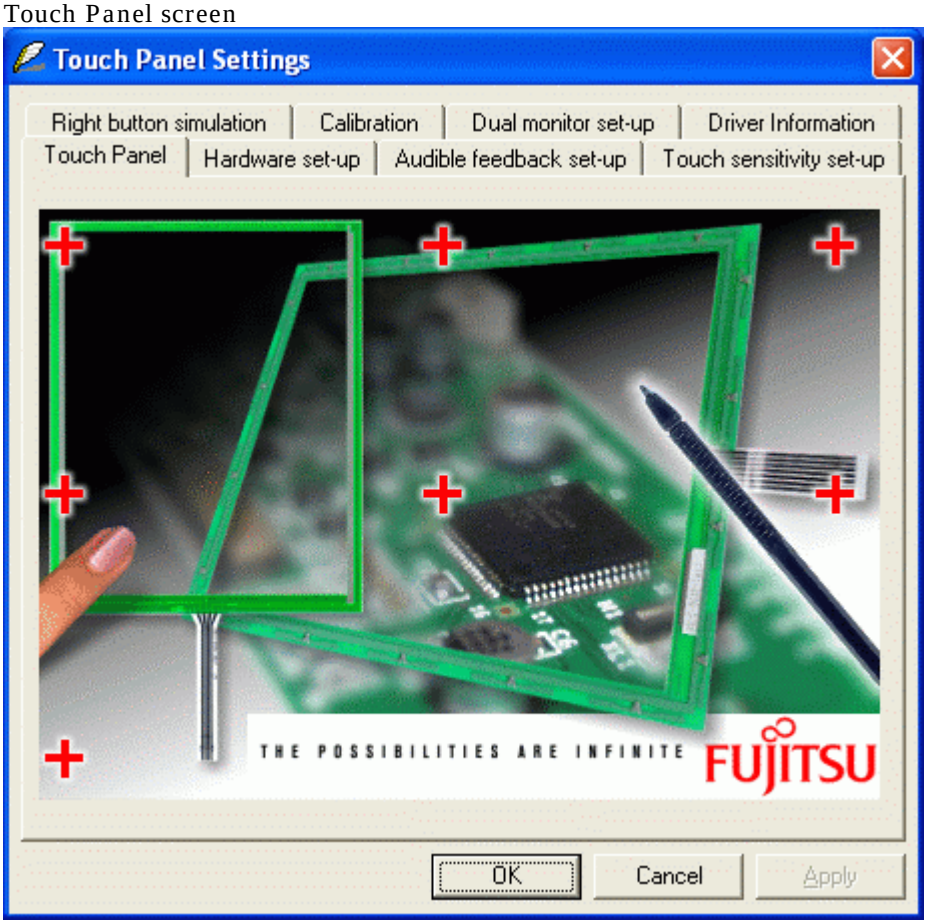
Start icon of touch panel setting tool

1.2 Setting screen

1.2.1 Start screen

When a setting tool is started, the dialog box in the figure below is displayed. This setting tool has eight pages in case of the NonPNP touch panel for Windows95/98/Me/NT4.0/2000/XP. And, This setting tool has seven pages because of there is no “Dual monitor set-up” page in case of the PNP touch panel for Windows95/98/Me/2000/XP

Touch Panel screen



*1) The same driver is used for Non PNP Touch and PNP Touch Panel in case of WindowsNT4.0.

1.2.2 Hardware set-up screen

When Windows95/98/Me PNP driver and Windows95/98/Me/NT4.0 NonPNP device driver are being installed. Figure 1 is indicated.

When Windows2000/XP PNP driver and WindowsWindows200/XP NonPNP driver and Widndows98/Me/NT4.0 NonPNP User mode driver are being installed. Figure 2 is indicated.

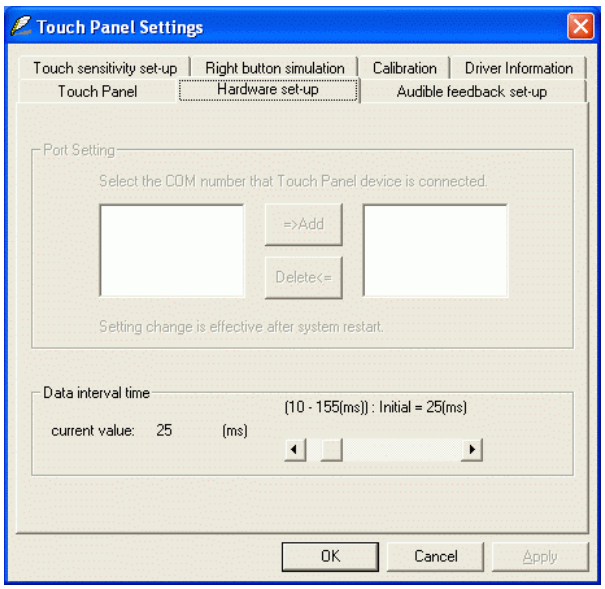


Figure 1

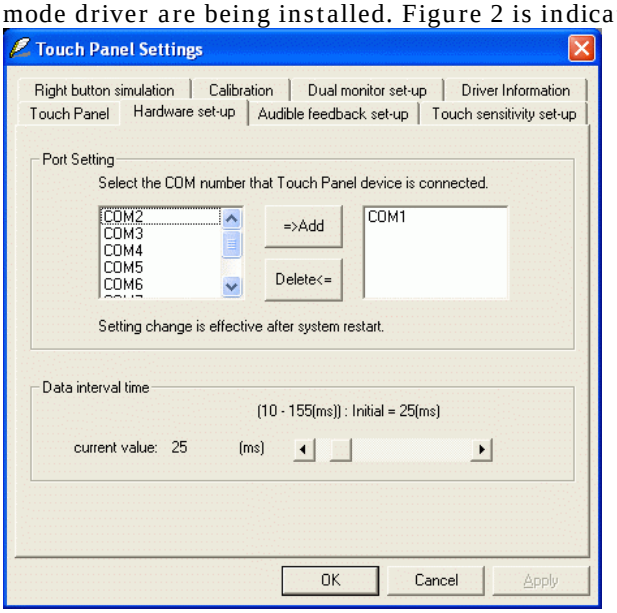


Figure 2

This page is concerning basic setting of the touch panel.
(Port setting and Data interval time).

<Port Setting>

It sets the port to where the touch panel is connected. You can select from COM1 to COM9.

[In case of the PNP touch panel for Windows95/98/Me/2000/XP]

The setup can't be changed, because COM is recognized automatically.
The port setup part is disabled with Windows2000/XP. The port setup part except for "Connected port" is disabled with Windows95/98/Me.
And, when you apply focus to the selected COM port shown in "Connected port" list box, resource data (I/O address and IRQ number) are displayed at the right.

[In case of the NonPNP touch panel which figure 1 is indicated in .]

You can add two maximum COM ports.
How to add a port : Input three fields of port setting below. And, it is added to the port connected by pushing an " => Add " button.*2)*3)*4)

COM : Input the COM port number that the device is connected.
I/O : Input the I/O address that the device is connected.
IRQ : Input the IRQ number that the device is connected.

How to delete a port : Select the port to delete from the list of " Connected port" . And, push " Delete <= " button. As a result, selected COM is deleted.*2)

[In case of the NonPNP touch panel which figure 2 is indicated in .]
You can add nine maximum COM ports.

How to add a port : Select the port to add from the left list of “=>Add” button.
And, push “=>Add” button. As a result, selected COM is added.* 2)

How to delete a port : Select the port to delete from the right list of " Delete <= " button.
And, push " Delete <= " button. As a result, selected COM is deleted.*2)

Notes

*2) Setting change should be effective after the system restart. And you can’t use "Calibration" tab page and "Use port" of "Dual monitor set-up" tab page until system restart.

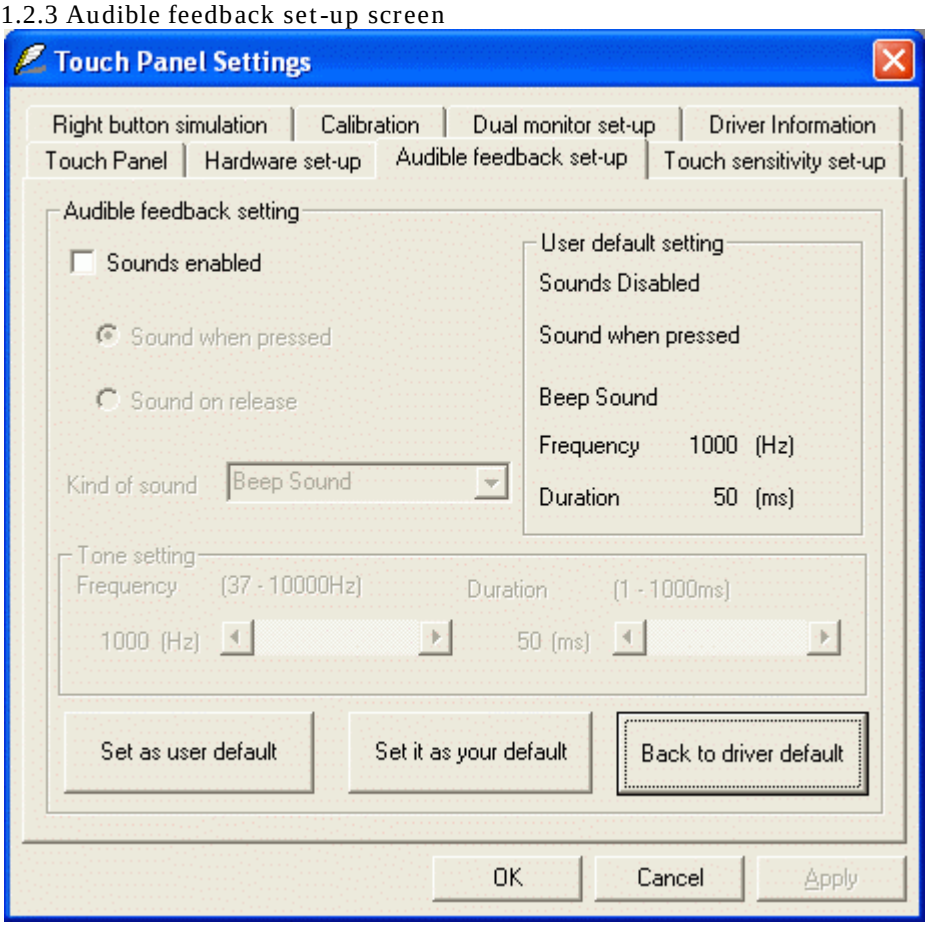
*3) There is a thing that PC causes defective operation when Port address(I/O address , IRQ number) not set by Windows is input.

*4) When ports other than COM1-9 are used, input a suitable port number of COM1-9 that is not used instead of actual COM port number. And, set the I/O address and the IRQ number as you wish to use.

<Data interval time>

Using slider in the scroll bar, you can change coordinates data interval time of touch panel controller from 10ms to 155ms by 5ms step.

1.2.3 Audible feedback set-up screen



This page is concerning a sound function of touch panel driver.

<Audible feedback setting>

Sound enabled	:Sound function is enabled when checked.
Sound when pressed	:The sound comes out when touch panel is pressed.
Sound on release	:The sound comes out when touch panel is released.
Kind of sound	:The sound can be selected from among Beep Sound, Asterisk, Exclamation, Question, Critical Stop, Default Sound, and Standard Sound .*5)
Back to driver default	:The sound setting is returned to driver default setting.(Sound disabled , Sound when pressed ,Beep Sound , Frequency:1000Hz , Beep time:50ms)
Set it as user default	: Present sound settings are preserved as standard setting. Contents preserved are displayed in the column of "User default setting".
Set as user default	:Back to user default setting according to preserved condition.

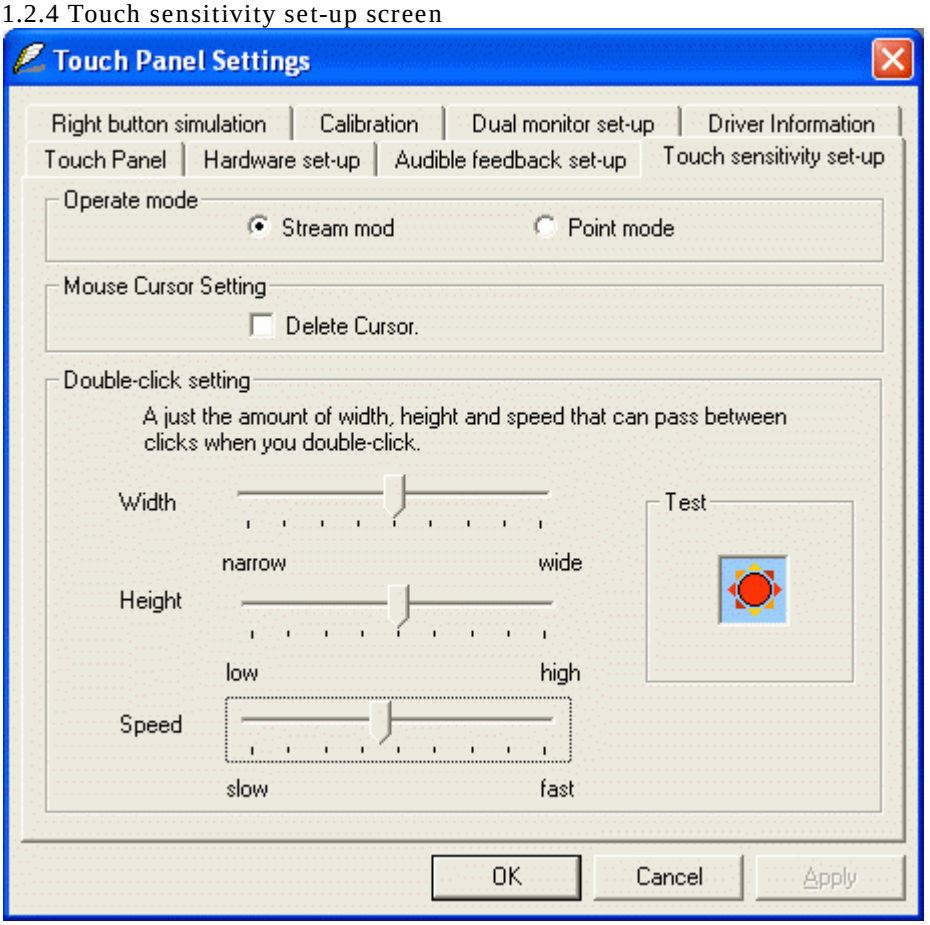
<Tone setting>

Frequency	: It sets the frequency when the kind of the sound is "Beep sound". (37~10000Hz)
Duration	: It sets the Beep time when the kind of the sound is “Beep sound”. (1~1000ms)

Notes

*5) Please use “Sounds” in the control panel to set the condition of Asterisk, Exclamation, Question, Critical Stop, Default Sound, and Standard Sound.

1.2.4 Touch sensitivity set-up screen



This page is concerning operation type and double-clicking.

<Operate mode>

Stream mode : The coordinates value is continuously output until touch panel turns to off.

Point mode : Point operation. Turning off is notified immediately after turning on is notified. Even if touch continues at the following, no data is output.

<Mouse Cursor Setting>Delete Cursor

The display of the mouse cursor is deleted . The special cursor of other application isn't deleted by this setup.

<Double-click setting>

The range and the speed to make the system recognize double-clicking can be changed. Moreover, it can be set that double-clicking is prohibited. Setting change is effective after “OK" or "Apply" button is pushed. *6)

Double-clicking prohibition : Disable the double-clicking of the touch panel when checked.

Width : The horizontal range recognized as double-clicking is set. 2 - 32 [pixel]

Height : The vertical range recognized as double-clicking is set. 2 - 32 [pixel]

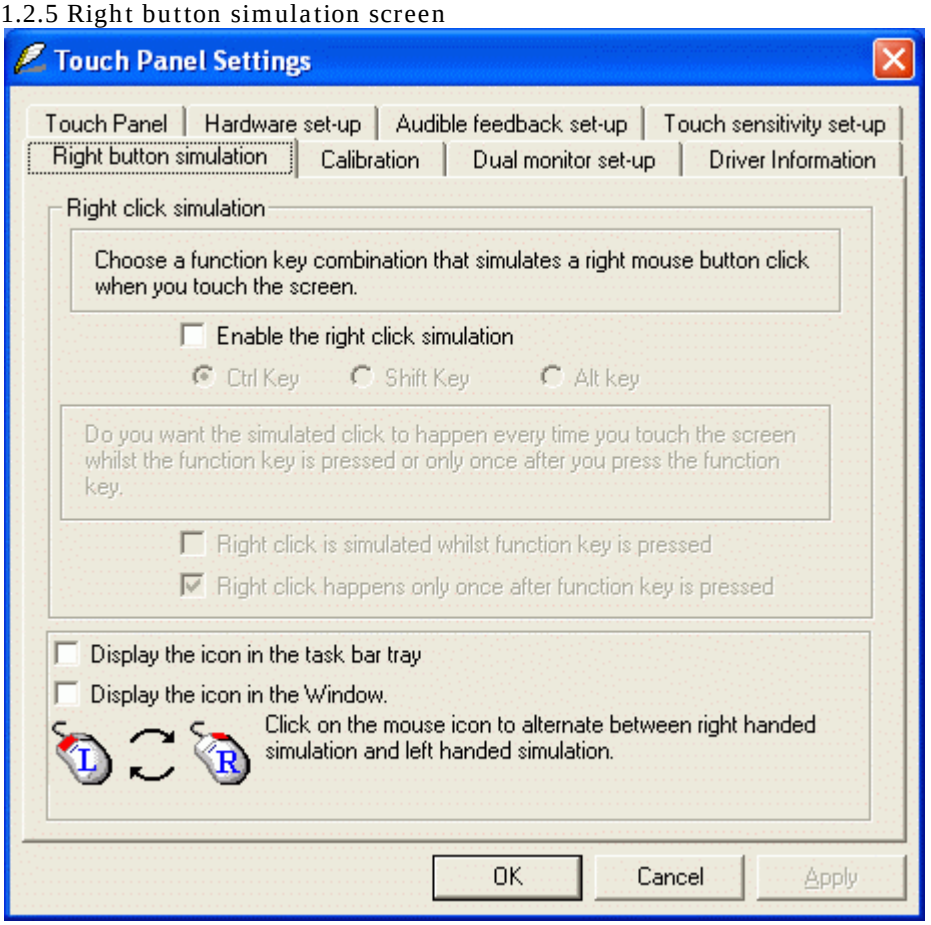
Speed : The maximum time between clicks recognized as double-clicking is set. 150 - 850[ms]

Test : The actual double-click test on displayed icon.
Icon display is changed when double-clicking is recognized

Notes

*6) Settings of the above-mentioned width, height, and the speed also affect to other mice connected with the system. But double-clicking prohibition is only for the touch panel.

1.2.5 Right button simulation screen



This page is concerning right button click simulation.

<Right click simulation>

Enable the right click simulation

: Enable function and activate following items when checked.

Ctrl key, Shift key, Alt key

: Select the cooperative key for right-clicking function.

Right click is simulated whilst function key is pressed

: Just as the guidance when checked. (The Alt key dose not work with this setting.)

Right click happens only once after function key is pressed

: Just as the guidance when checked.

Display the icon in the task bar tray

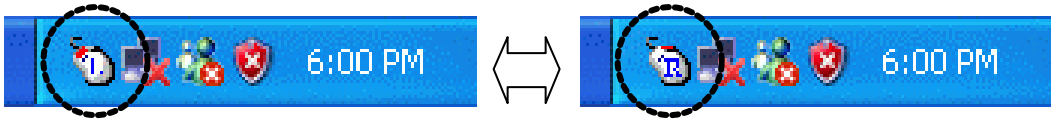
: The icon of the mouse style Figure 3 is registered in the task tray and right click simulation works without cooperative key operation when checked.

Display the icon in the Window

: The icon of the mouse style Figure 4 is registered in the desktop and right click simulation works without cooperative key operation when checked.

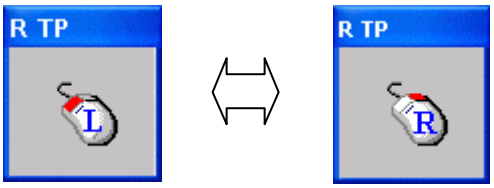
Operation of icon

: The letter of “L” and “R” of the icon shows a present state. At the “R” state, touch works as right-click once and state turns to “L”. State is changed alternatively by touching icon.



Display of “L” Display of “R”

Figure 3



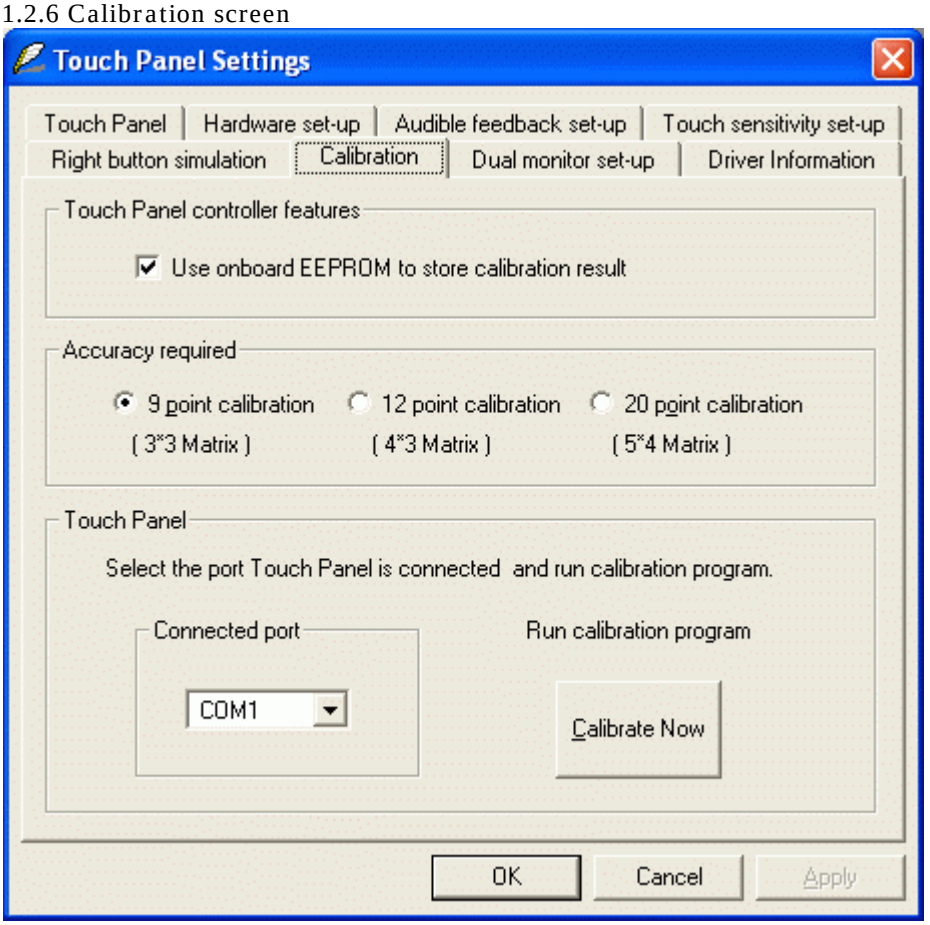
Display of “L” Display of “R”

Figure 4

Notes

*7) When the mouse is set for left-handed, the display of L and R becomes opposite.

1.2.6 Calibration screen



This page is concerning the correction of the touch panel.

<Touch Panel controller features>

Use onboard EEPROM to store calibration results : Do not check here when EEPROM is not used.

<Accuracy required>

Select the calibration point number.

<Calibration>*8)

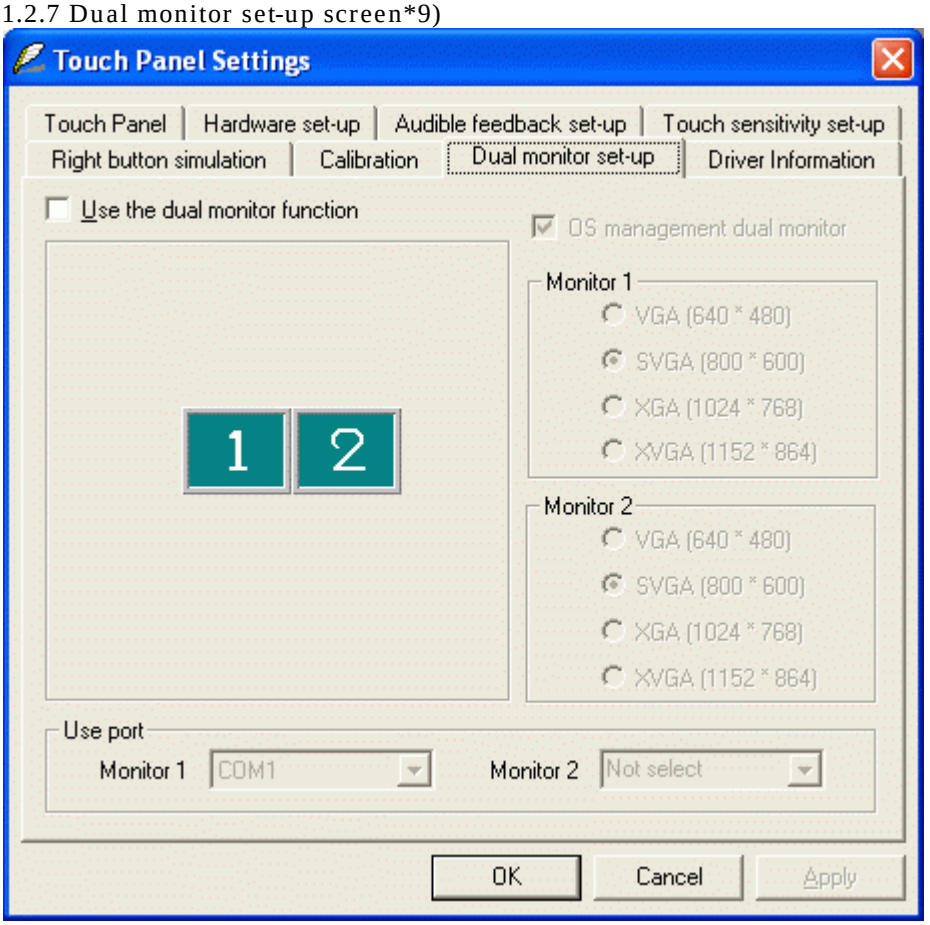
Connected port : Select target touch panel to do calibration when you are using two panels. (In case of a PNP touch panel of WindowsNT95/98/Me/2000/XP , COM is recognized automatically. Because of that, it doesn't need to choose it.)

Calibration Now :This button executes the calibration program. When “Touch Panel controller features” and “Accuracy required” are changed. Calibration can't be carried out until “OK” and “Apply” button are pushed and a setup is renewed. (Detail operation is described in section 2.)

Notes

*8) When you change the port setting in the “Hardware set-up” tab, the calibration program can't be carried out till system restarts.

1.2.7 Dual monitor set-up screen*9)



This page is concerning the dual monitor usage.

<Use the dual monitor function>
Check here when you wish to use dual monitor function.

<OS management dual monitor>
Select whether it is OS management dual monitor or video card management dual monitor.
It is disable in case of WindowsNT4.0.

<Monitor position setting>
Drag the picture of monitor 1 and 2, and set the position same as actual monitor arrangement.

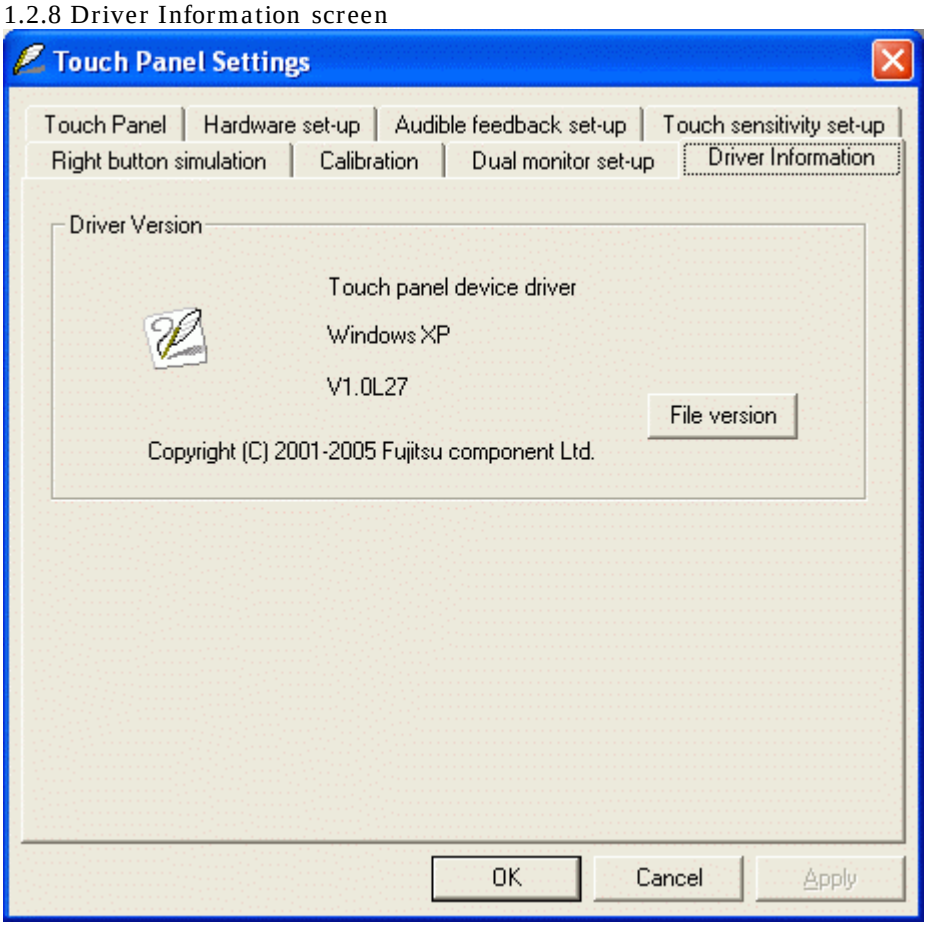
<Monitor 1>
Select the resolution of monitor1.

<Monitor 2>
Select the resolution of monitor2.

<Use port> *10)
Monitor 1 : Select the port of the touch panel set up on monitor1.
Monitor 2 : Select the port of the touch panel set up on monitor2.

Notes
*9) Dual monitor set-up screen isn't shown with a PNP touch panel for Widnows95/98/Me/2000/XP.
*10) When you change the port setting in the “Hardware set-up” tab, “Use port” can't be changed till system restarts.

1.2.8 Driver Information screen



Information on the driver is displayed.

<Driver Version>

Driver name , applied operating system and driver version are indicated.

Clicking “File version” button, you can see information of individual file included in this touch panel driver set.

2. Calibration program

The calibration is done so that pressed position of touch panel device matches to the display position. When the touch panel driver setup or the touch panel device is exchanged, it is necessary to calibrate.

2.1 Start of calibration program

Double-click the "Touch Panel" icon on the control panel and then click "Calibration" tab. Select the port in “Connected port”, and click “Calibration Now”.

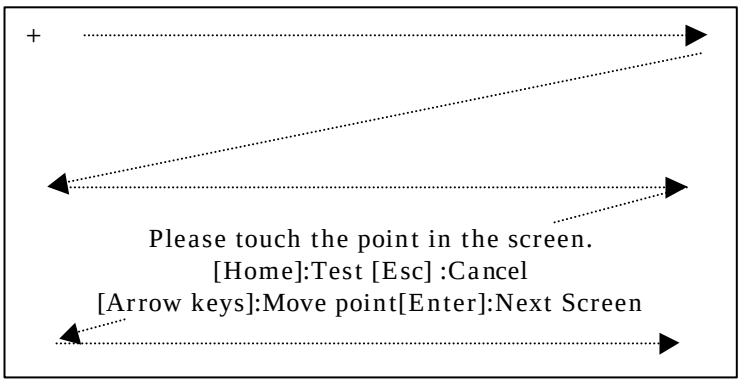
2.2 Composition of calibration program

The calibration program is composed of “Calibration screen” where point input operation is done and “Test screen” to confirm calibration result.

2.3 Calibration screen

1) Execute the calibration program.
2) The calibration screen is displayed (one “ + “ mark is displayed on the left up corner).
3) Push “ + “ mark. When a push point was effective. The following " + " mark is displayed.
4) After all “ + “ marks are pressed (9 or 12 or 20 pieces), screen is changed to “Test screen” automatically.

< Calibration screen>



Key input at calibration screen

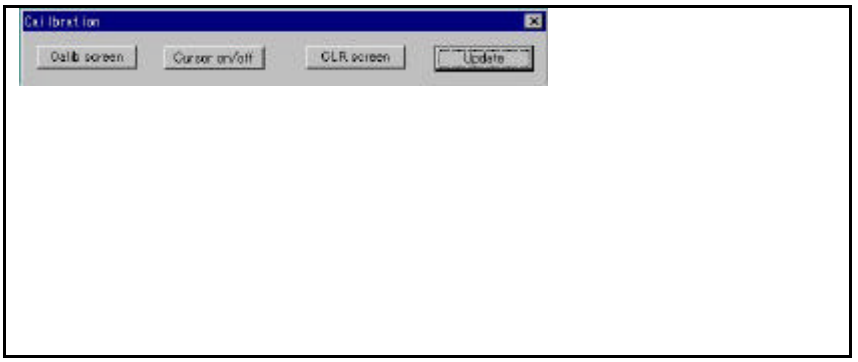
[Home]	: Cancel calibration data and skip to test screen.
[Esc]	: Cancel calibration data and close calibration program.
[Arrow keys]	: Move the calibration point mark.
[Enter]	: Revise the calibration data and skip to test screen.

Notes

*11) When you use the onboard EEPROM, check on “Use onboard EEPROM to store calibration results”. First time you do this, it is necessary to input all the calibration points. But a partial input is possible since second time or later calibration. Calibration data and calibration point is kept in registry. And, you must input all calibration points every time.

2.4 Test screen

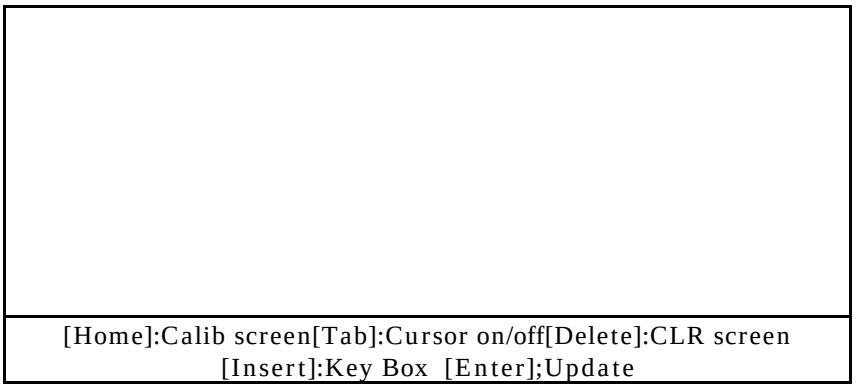
You can do free drawing in this screen so that the result of the calibration can be judged by watching pen input position and displayed position.



<Drawing test screen where key box is used >

Key box input of this drawing test screen

Calib screen	: Back to calibration screen.
Cursor on/off	: On and off the cursor display on the screen.
CLR screen	: Clear the screen.
Update	: Close the calibration program.
Close button	: Close the key box window, and screen changes to the style below.



< Drawing test screen where key box is not used >

Key input of this drawing test screen

[Home]	: Back to calibration screen.
[Tab]	: On and off the cursor display on the screen.
[Delete]	: Clear the screen and key guidance part moves up and down.
[Insert]	: The key box is displayed and screen changes into the first style.
[Enter]	: Close the calibration program.

3. The setup of coordinate rotate.

3.1 Registry change method

The setup of coordinate rotate is made to take a coordinate starting point in rotation by changing “SpinTop” of the registry. That process show in the following.

(1) It clicks on the "Start" button, and open "Run..." of the system.
(2) It is input to the column of “Open” as “regedit”.
(3) Click “OK” button.
(4) Select “HKEY_LOCAL_MACHINE¥Software¥Fujitsu Takamisawa¥Serial”.*10)
(5) The value of “SpinTop” is changed as follows.

-The setup value coordinate rotation

The setup value of the screen 1

0x00000000: There is no coordinates rotation.
0x00000001: Coordinates rotate right 90 degrees.
0x00000002: Coordinates rotate right 180 degrees.
0x00000003: Coordinates rotate right 270 degrees.

The setup value of the screen 2

0x00000000: There is no coordinates rotation.
0x00000010: Coordinates rotate right 90 degrees.
0x00000020: Coordinates rotate right 180 degrees.
0x00000030: Coordinates rotate right 270 degrees.

* In case of a PNP touch panel driver. The screen 2 can't be set up.

(6) Restart the system.

*12) “SpinTop” of the following registry is changed by the version of Windows.

[Touch panel for WindowsNT.]
“HKEY_LOCAL_MACHINE¥SYSTEM¥CurrentControlSet¥Services¥FidmousP¥Parameters”
“HKEY_LOCAL_MACHINE¥SYSTEM¥CurrentControlSet¥Services¥FidmousS¥Parameters”
Both "SpinTop"' are changed to the same value.

[Non PNP touch panel for Windows95/98/Me.]
“HKEY_LOCAL_MACHINE¥Software¥Fujitsu Takamisawa¥Serial”

[PNP touch panel for Windows95/98/Me.]
“HKEY_LOCAL_MACHINE¥Software¥Fujitsu Takamisawa¥SerialPnP”

[Non PNP touch panel for Windows2000/XP.]
“HKEY_LOCAL_MACHINE¥Software¥Fujitsu Takamisawa¥FIDTSERV”

[PNP touch panel for Windows2000/XP.]
“HKEY_LOCAL_MACHINE¥ System¥CurrentControlSet¥Services¥FIDMOUS¥Parameters”

3.2 The method which rotation a coordinate starting point from the outside application .
The outside application can rotate a coordinate starting point like the setup of a registry.
This setup is cleared after the power supply re-injection. "How to make the outside application" is shown in the following.

3.2.1 Device driver for Windows95/98/Me NonPNP , Driver for Windows95/98/Me PNP and Driver for Windows2000/XP PNP

(1) The handle of the driver is acquired by using the “CreateFile” function. Symbolic name of each driver is defined as follows.

-Symbolic name

Driver for Windows95/98/Me PNP	:”FIDMOUR”
Driver for Windows2000/XP PNP	:”FIDMOUS”
Device driver for Windows95/98/Me NonPNP (Primary)	:”FIDMOUA”
Device driver for Windows95/98/Me NonPNP (Secondary)	:”FIDMOUSR”

(2) The setup of the coordinate rotation and acquisition of present coordinate rotation information are done by using the "DeviceIoControl" function. Each control code and setup value is defined as follows. "winioctl.h" is necessary to use this function.

- Control code

[Driver for Windows95/98/Me PNP, Device driver for Windows95/98/Me NonPNP]

The setup of the coordinate rotation
0x7000000c
Acquisition of present coordinate rotation information
0x7000000f

[Driver for Windows2000/XP PNP]
The setup of the coordinate rotation
CTL_CODE(FILE_DEVICE_UNKNOWN, 0x0841, METHOD_BUFFERED, FILE_ANY_ACCESS)
Acquisition of present coordinate rotation information
CTL_CODE(FILE_DEVICE_UNKNOWN, 0x0840, METHOD_BUFFERED, FILE_ANY_ACCESS)

-The setup value coordinate rotation
The setup value of the screen 1
0x00000000: There is no coordinates rotation.
0x00000001: Coordinates rotate right 90 degrees.
0x00000002: Coordinates rotate right 180 degrees.
0x00000003: Coordinates rotate right 270 degrees.
The setup value of the screen 2
0x00000000: There is no coordinates rotation.
0x00000010: Coordinates rotate right 90 degrees.
0x00000020: Coordinates rotate right 180 degrees.
0x00000030: Coordinates rotate right 270 degrees.

* In case of a PNP touch panel driver. The screen 2 can't be set up.

(3) The handle of the driver is closed by using the “CloseHandale” function.

3.2.2 Device driver for WindowsNT4.0

(1) The handle of the driver is acquired by using the “CreateFile” function. Symbolic name of each driver is defined as follows.

-Symbolic name
“NTMOU”

(2) Communication with the driver is done by using the “DeviceIoControl” function, and the unit ID of the driver is acquired by distinguishing a driver ID. Control code and driver ID of Primary and Secondary is defined as follows. "winioctl.h" is necessary to use this function.

- Control code
CTL_CODE(FILE_DEVICE_SERIAL_MOUSE_PORT,0x0800,METHOD_BUFFERED,FILE_ANY_ACCESS)

-Driver ID
Primary ID :7580
Secondary ID :6800

(3) The setup of the coordinate rotation and acquisition of present coordinate rotation information are done by using the "DeviceIoControl" function. Each control code and setup value is defined as follows.

- Control code
The setup of the coordinate rotation
CTL_CODE(FILE_DEVICE_SERIAL_MOUSE_PORT,0x080c,METHOD_BUFFERED, FILE_ANY_ACCESS)
Acquisition of present coordinate rotation information
CTL_CODE(FILE_DEVICE_SERIAL_MOUSE_PORT,0x080f,METHOD_BUFFERED, ILE_ANY_ACCESS)

-The setup value coordinate rotation
The setup value of the screen 1
0x00000000: There is no coordinates rotation.
0x00000001: Coordinates rotate right 90 degrees.
0x00000002: Coordinates rotate right 180 degrees.
0x00000003: Coordinates rotate right 270 degrees.
The setup value of the screen 2
0x00000000: There is no coordinates rotation.
0x00000010: Coordinates rotate right 90 degrees.
0x00000020: Coordinates rotate right 180 degrees.
0x00000030: Coordinates rotate right 270 degrees.

(4) The handle of the driver is closed by using the “CloseHandale” function.

3.2.3 User mode driver for Windows98/Me/NT4.0 NonPNP, Driver for Windows2000/XP NonPNP

(1) The handle of DLL (SpinDll.dll) for the coordinate rotation is acquired by using" LoadLibrary" function.
(2) The address of the function to set up coordinate rotation (FPHOOK_SetSPINTOP) and the function to acquire present coordinate rotation information (FPHOOK_GetSPINTOP) which DLL for the coordinate rotation is acquired by using" GetProcAddress" function.
(3) The setup of the coordinate rotation and information on the present coordinate rotation is done by using the address of the acquired function. Setup value is defined as follows.

-The setup value coordinate rotation
The setup value of the screen 1
0x00000000: There is no coordinates rotation.
0x00000001: Coordinates rotate right 90 degrees.
0x00000002: Coordinates rotate right 180 degrees.
0x00000003: Coordinates rotate right 270 degrees.
The setup value of the screen 2
0x00000000: There is no coordinates rotation.
0x00000010: Coordinates rotate right 90 degrees.
0x00000020: Coordinates rotate right 180 degrees.
0x00000030: Coordinates rotate right 270 degrees.

(4) The handle of DLL for the coordinate rotation is invalidated by using the "FreeLibrary" function.

3.3 Program example .

3.3.1 PNP touch panel driver for Windows95/98/Me

```
//A necessary header file
#include <winioctl.h>

//Control code
#define             FTIOCTL_9X_SetSpinTop             0x7000000c
#define             FTIOCTL_9X_GetSpinTop             0x7000000f

//Variable
HANDLE hKeD;                        //The handle of a touch panel driver to acquire
ULONG   cbReturned;
typedef struct      tagDANGLE_98
{
 ULONG   Degree;                    //The variable of the coordinate rotation to transmit and receive it
}DANGLE_98, *PDANGLE_98;
DANGLE_98                 DAngle_98;

// The handle acquisition of the driver
  hKeD= CreateFile("¥¥¥¥.¥¥FIDMOUR",0,0,NULL,0,0,NULL);

//The setup of the coordinate rotation
DAngle_98.Degree = 0x1;                          //0: 0 degrees  1: 90 degrees  2: 180 degrees  3: 270 degrees
(DeviceIoControl(hKeD  ,FTIOCTL_9X_SetSpinTop
          ,&DAngle_98  ,sizeof(DANGLE_98)
          ,NULL ,0 ,&cbReturned,   NULL));

//The acquisition of the present coordinate rotation information .
(DeviceIoControl(hKeD,   FTIOCTL_9X_GetSpinTop
                    ,NULL   ,0          ,&DAngle_98
           ,sizeof(DANGLE_98)
          ,&cbReturned,NULL));

//Close the handle of the driver
CloseHandle (hKeD);
```

3.3.2 PNP touch panel driver for Windows2000/XP

```
//A necessary header file
#include <winioctl.h>

//Control code
#define FTIOCTL_2K_GetSpinTop
CTL_CODE(FILE_DEVICE_UNKNOWN, 0x0840, METHOD_BUFFERED, FILE_ANY_ACCESS)

#define FTIOCTL_2K_SetSpinTop
CTL_CODE(FILE_DEVICE_UNKNOWN, 0x0841, METHOD_BUFFERED, FILE_ANY_ACCESS)

//Variable
HANDLE hKeD;                            //The handle of a touch panel driver to acquire
char completeDeviceName[] = "¥¥¥¥.¥¥FIDMOUS";
ULONG       cbReturned;
ULONG       SpinTop ;                   //The variable of the coordinate rotation to transmit it
ULONG       SpinData;                   //The variable of the coordinate rotation to receive it

// The handle acquisition of the driver
hKeD = CreateFile(completeDeviceName,
                  GENERIC_READ | GENERIC_WRITE,
                  FILE_SHARE_READ | FILE_SHARE_WRITE,
                  NULL,
                  OPEN_EXISTING, 0, NULL);

//The setup of the coordinate rotation
SpinTop  =  0x1;                        //0: 0 degrees  1: 90 degrees  2: 180 degrees  3: 270 degrees
DeviceIoControl(hKeD, FTIOCTL_2K_SetSpinTop,
                &SpinTop, sizeof(ULONG),
                NULL, 0,
                &cbReturned, NULL);

//The acquisition of the present coordinate rotation information
DeviceIoControl(hKeD, FTIOCTL_2K_GetSpinTop,
                NULL, 0,
                &SpinData, sizeof(ULONG),
                &cbReturned, NULL);

//Close the handle of the driver
CloseHandle(hKeD);
```

3.3.3 NonPNP touch panel driver for Windows95/98/Me (Device driver)

```
//A necessary header file
#include <winioctl.h>

//Control code
#define         FTIOCTL_9X_SetSpinTop           0x7000000c
#define         FTIOCTL_9X_GetSpinTop           0x7000000f
//Variable
HANDLE hKeD;                    //The handle of a touch panel driver to acquire
ULONG cbReturned;

typedef struct      tagDANGLE_98
{
        ULONG  Degree;          //The variable of the coordinate rotation to transmit and receive it
}DANGLE_98, *PDANGLE_98;

DANGLE_98       DAngle_98;

// The handle acquisition of the driver
  hKeD= CreateFile("¥¥¥¥.¥¥FIDMOUA",0,0,NULL,0,0,NULL);

//The setup of the coordinate rotation
DAngle_98.Degree = 0x01;            //The setup value that 0byte is a screen 1
                                    //The setup value that 1byte is a screen 2
                                    //0: 0 degrees  1: 90 degrees  2: 180 degrees  3: 270 degrees

(DeviceIoControl(hKeD  ,FTIOCTL_9X_SetSpinTop
                ,&DAngle_98  ,sizeof(DANGLE_98)
                ,NULL ,0 ,&cbReturned,  NULL));

//The acquisition of the present coordinate rotation information
(DeviceIoControl(hKeD,  FTIOCTL_9X_GetSpinTop
                ,NULL  ,0           ,&DAngle_98
                 ,sizeof(DANGLE_98)
                ,&cbReturned,NULL));

//Close the handle of the driver
CloseHandle (hKeD);
```

*Carry out a CreateFile function with Secondary driver as follows because symbolic name is different.

```
        hKeD= CreateFile("¥¥¥¥.¥¥FIDMOUSR",0,0,NULL,0,0,NULL);
```

3.3.3 NonPNP touch panel driver for WindowsNT4.0 (Device driver)

```
//A necessary header file
#include <winioctl.h>

//Control code
#define   FTIOCTL_NT_GETID
CTL_CODE(FILE_DEVICE_SERIAL_MOUSE_PORT, 0x0800, METHOD_BUFFERED, FILE_ANY_ACCESS)

#define   FTIOCTL_NT_ANGLECHANGE
CTL_CODE(FILE_DEVICE_SERIAL_MOUSE_PORT, 0x080c, METHOD_BUFFERED, FILE_ANY_ACCESS)

#define   FTIOCTL_NT_GETSPINTOP
CTL_CODE(FILE_DEVICE_SERIAL_MOUSE_PORT, 0x080f, METHOD_BUFFERED, FILE_ANY_ACCESS)

//NT drver ID
#define PrimaryID           7580                    //Primary driver ID
#define SecondaryID         6800                    //Secondary driver ID

//Variable
HANDLE hKeD;                                        //The handle of a touch panel driver to acquire
ULONG cbReturned;

typedef struct  tagMOUSE_PRIVATE_IN {
        USHORT unitId;
        UCHAR  code[20];
} MOUSE_PRIVATE_IN, *PMOUSE_PRIVATE_IN;

typedef struct  tagMOUSE_PRIVATE_OUT {
        USHORT unitId;
        UCHAR  code[20];
} MOUSE_PRIVATE_OUT, *PMOUSE_PRIVATE_OUT;

typedef struct      tagDANGLE{
        USHORT unitId;
        ULONG  Degree;                  //The variable of the coordinate rotation to transmit and receive it
}DANGLE, *PDANGLE;

DANGLE                  DAngle;
MOUSE_PRIVATE_IN        privatein;
MOUSE_PRIVATE_OUT       privateout;

USHORT          Primary_unitId;
USHORT          Secondary_unitId;
char            completeDeviceNameNT[] = "¥¥¥¥.¥¥NTMOU";
// The handle acquisition of the driver
hKeD = CreateFile ( completeDeviceNameNT,
        GENERIC_READ | GENERIC_WRITE,
                0, NULL,
                OPEN_EXISTING ,0, NULL);

//The acquisition of the unit ID
for (privatein.unitId=0; privatein.unitId <=3; ++(privatein.unitId))
{
        if (DeviceIoControl (hKeD, (DWORD)FTIOCTL_NT_GETID,
                        &privatein,         sizeof(MOUSE_PRIVATE_IN),
                        &privateout, sizeof(MOUSE_PRIVATE_OUT),
                        &cbReturned, NULL))
        {
                if(privateout.unitId == PrimaryID)
                        Primary_unitId = privatein.unitId;
                else if (privateout.unitId == SecondaryID)
                        Secondary_unitId = privatein.unitId;
        }
}

//The setup of the coordinate rotation .
```

```
DAngle.Degree = 0 x01;                          //The setup value that 0byte is a screen 1
                                                //The setup value that 1byte is a screen 2
                                                //0: 0 degrees  1: 90 degrees  2: 180 degrees  3: 270 degrees
DAngle.unitId = Primary_unitId;
(DeviceIoControl (hKeD, (DWORD)FTIOCTL_NT_ANGLECHANGE,
                        &DAngle, sizeof(DANGLE),
                        NULL, 0,
                        &cbReturned, NULL));

//The acquisition of the present coordinate rotation information
 (DeviceIoControl (hKeD, (DWORD)FTIOCTL_NT_GETSPINTOP,
                        NULL,0,
                        &DAngle, sizeof(DANGLE),
                        &cbReturned, NULL));

//Close the handle of the driver
CloseHandle (hKeD);
```

*Take the unit ID which transmits Secondary_unitId with Secondary driver.

3.3.4 NonPNP touch panel driver for Windows98/Me/NT4.0/200/XP (User mode driver)

```
// Variable
HINSTANCE    hFPSpin;
ULONG        SpinTop ;                  // The variable of the coordinate rotation to transmit it
ULONG        SpinData;                  // The variable of the coordinate rotation to receive it

typedef void (*pFPSPIN_SetSPINTOP)(DWORD);
typedef DWORD (*pFPSPIN_GetSPINTOP)();

pFPSPIN_SetSPINTOP    FPSPIN_SetSPINTOP;
pFPSPIN_GetSPINTOP    FPSPIN_GetSPINTOP;

//Load SpinDll.DLL
hFPSpin= LoadLibrary("SPINDLL.DLL");

// The address of the function inside SpinDll.DLL is acquired
FPSPIN_SetSPINTOP = (pFPSPIN_SetSPINTOP)GetProcAddress (hFPSpin, "FPHOOK_SetSPINTOP");
FPSPIN_GetSPINTOP = (pFPSPIN_GetSPINTOP)GetProcAddress (hFPSpin, "FPHOOK_GetSPINTOP");

// The setup of the coordinate rotation
SpinTop  =  0x01;                       //The setup value that 0byte is a screen 1
                                        //The setup value that 1byte is a screen 2
                                        //0: 0 degrees  1: 90 degrees  2: 180 degrees  3: 270 degrees
(*FPSPIN_SetSPINTOP)(SpinTop);

// The acquisition of the present coordinate rotation information
SpinData = ((*FPSPIN_GetSPINTOP)());

//Close SpinDll.DLL
FreeLibrary(hFPSpin);
```