



PCI-P16R16 Series Classic Driver DLL Software Manual

Version 1.2, Jun. 2015

SUPPORT

This manual relates to the following boards: PCI-P8R8, PCI-P8R8U, PCI-P16R16U, PCI-P16R16, PCI-P16C16, PCI-P16POR16, PCI-P16POR16U, PEX-P8POR8i and PEX-P16POR16i.

WARRANTY

All products manufactured by ICP DAS are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

WARNING

ICP DAS assumes no liability for damages consequent to the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, nor for any infringements of patents or other rights of third parties resulting from its use.

COPYRIGHT

Copyright © 2015 by ICP DAS. All rights are reserved.

TRADEMARK

Names are used for identification only and may be registered trademarks of their respective companies.

CONTACT US

If you have any question, please feel to contact us at:

service@icpdas.com; service.icpdas@gmail.com

We will give you quick response within 2 workdays.



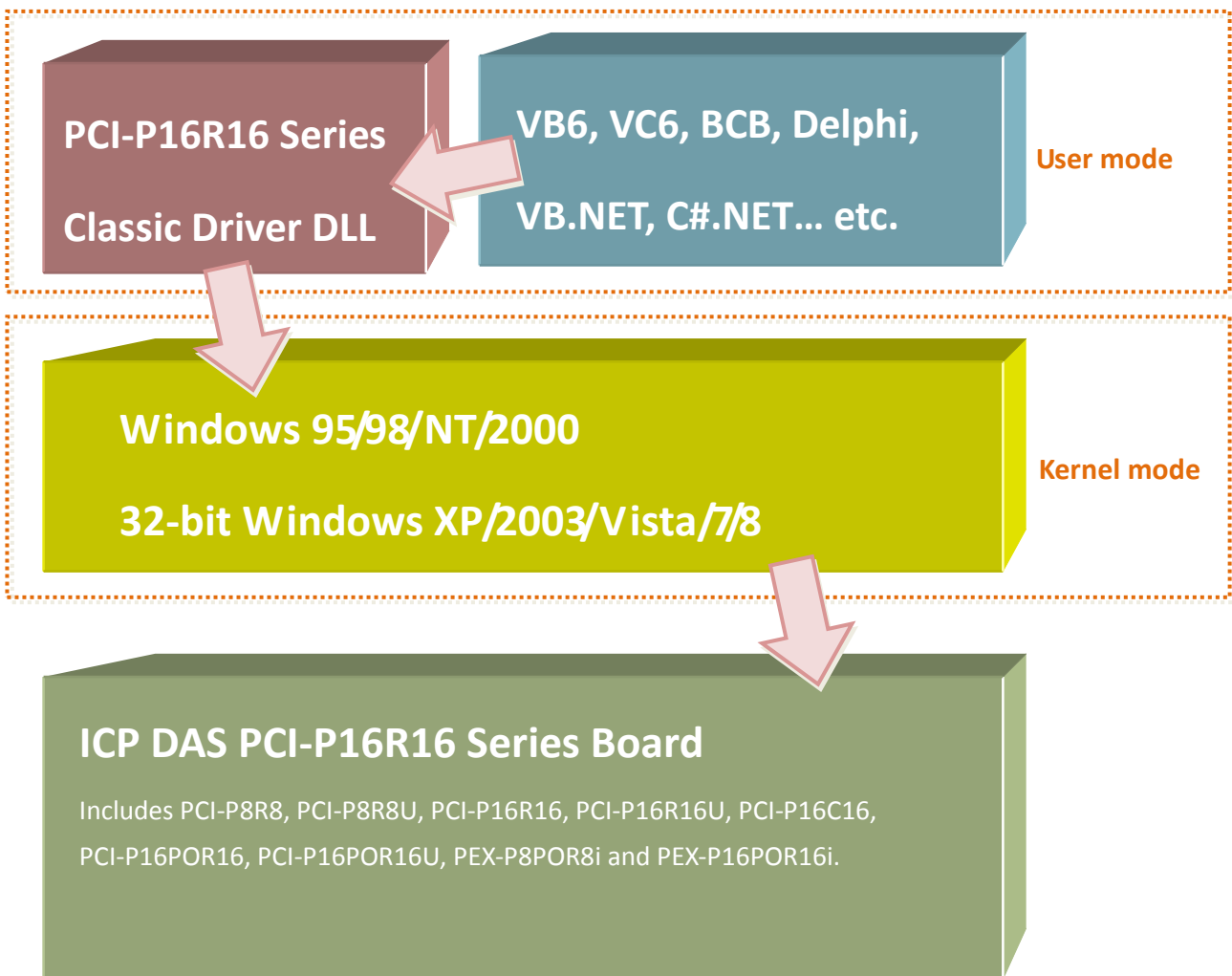
TABLE OF CONTENTS

1.	INTRODUCTION	2
1.1	REFERENCE.....	3
2.	SOFTWARE INSTALLATION	4
2.1	OBTAINING THE DRIVER INSTALLER PACKAGE	4
2.2	DRIVER INSTALLING PROCEDURE	5
2.3	PLUG AND PLAY DRIVER INSTALLATION	7
2.4	UNINSTALLING THE PCI-P16R16 SERIES CLASSIC DRIVER.....	9
3.	DLL FUNCTION DESCRIPTIONS	11
3.1	ERROR CODE TABLE.....	13
3.2	TEST FUNCTIONS	14
	<i>PCI_FloatSub2</i>	14
	<i>PCI_ShortSub2</i>	15
	<i>PCI_GetDllVersion</i>	16
3.3	DRIVER INITIALIZATION FUNCTIONS	17
	<i>PCI_DriverInit</i>	17
	<i>PCI_DriverClose</i>	19
	<i>PCI_GetDriverVersion</i>	19
	<i>PCI_GetConfigAddressSpace</i>	20
3.4	DIGITAL I/O FUNCTIONS FOR PCI-P16R16 SERIES	22
	<i>P16R16_DO</i>	22
	<i>P16R16_DI</i>	23
3.5	DIGITAL I/O FUNCTIONS FOR PCI-P8R8 SERIES	27
	<i>P8R8_DO</i>	27
	<i>P8R8_DI</i>	28
4.	DEMO PROGRAMS	32
4.1	FOR MICROSOFT WINDOWS	32
4.2	FOR DOS.....	33
5.	PROBLEMS REPORT	35

1. Introduction

The software is a collection of Digital Input/Output subroutines for PCI-P16R16 Series card add-on cards for **Windows 95/98/NT, Windows 2000 and 32-bit Windows XP/2003/Vista/7/8** applications. The application structure is presented in the following diagram.

The subroutines in **P16R16.DLL** are easy understanding as its name standing for. It provides powerful, easy-to-use subroutine for developing your data acquisition application. Your program can call these DLL functions by **VB, VC, Delphi, BCB, VB.NET 2005 and C#.NET 2005, etc.** easily. To speed-up your developing process, some demonstration source program are provided.



1.1 Reference

Please refer to the following user manuals:

- **PCI-P16R16_PnP_Driver_Installation.pdf:**
Describes how to install the PnP (Plug and Play) driver for PCI-P16R16 series card under Windows 95/98.
<http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pci-p16r16/manual/>

- **PCI-P16R16_DLL_OCX_Installation.pdf:**
Describes how to install the software package under Windows 95/98 and Windows NT.
<http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pci-p16r16/manual/>

- **PCI-P8R8_P16R16_Series_Hardware_Manual.pdf:**
PCI-P16R16 series card hardware manual for PCI-P8R8/P16R16, PCI-P16POR16/P16POR16U, PCI-P16C16 and PEX-P8POR8i/P16POR16i.
<http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pci-p16r16/manual/>

- **Resource Checking .pdf:**
Describes how to check the resources I/O Port address, IRQ number and DMA number for add-on cards under Windows 95/98/2000/XP/2003/ Vista/2008/7(32-bit).
<http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/manual/>

- **Calling DLL Functions.pdf:**
Describes how to call the DLL functions with VC++6, VB6, Delphi4 and Borland C++ Builder 4.
<http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/manual/>


2. Software Installation

This chapter describes the process for installing the PCI-P16R16 Series Classic Driver for the PCI-P16R16 Series card.

2.1 Obtaining the Driver Installer Package

PCI-P16R16 series card can be used on Linux and Windows 95/98/NT/2000 and 32-bit XP/2003/Vista/7/8 based systems, and the drivers are fully Plug & Play (PnP) compliant for easy installation.

The driver installer package for the PCI-P16R16 series can be found on the supplied CD-ROM, or can be obtained from the ICP DAS FTP web site. The location and addresses are indicated in the table below:

	CD:\\ NAPDOS\\PCI\\PCI-P16R16\\DLL_OCX\\
	ftp://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pci-p16r16/dll_ocx/
	http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pci-p16r16/dll_ocx/

Install the appropriate driver for your operating system, as follows:

Name	Folder	Operating System
Pci-p16r16_Winxxxx_ xxx.exe	Win98	For Windows 95 and Windows 98
	WinNT	For Windows NT
	Win2K_XP_7	For Windows 2000, 32-bit Windows XP, 32-bit Windows 2003, 32-bit Windows Vista , 32-bit Windows 7 and 32-bit Windows 8 .

2.2 Driver Installing Procedure

Before the driver installation, you must complete the hardware installation. For detailed information about the hardware installation, please refer to appropriate hardware user manual for your PCI-P16R16 series card.

The hardware user manual is contained in:



CD:\NAPDOS\PCI\PCI-P16R16 \Manual\



<http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pci-p16r16/manual/>

To install the PCI-P16R16 series classic drivers under Windows XP, follow the procedure described below.

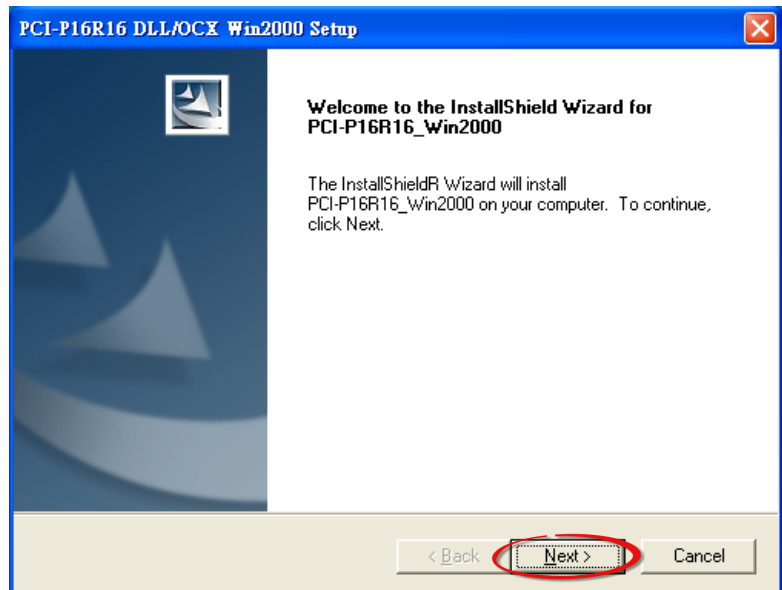


Note that if operating system is the Windows 95/98 and Windows NT, refer to “PCI-P16R16_DLL_OCX_Installation.pdf” for more detailed information.

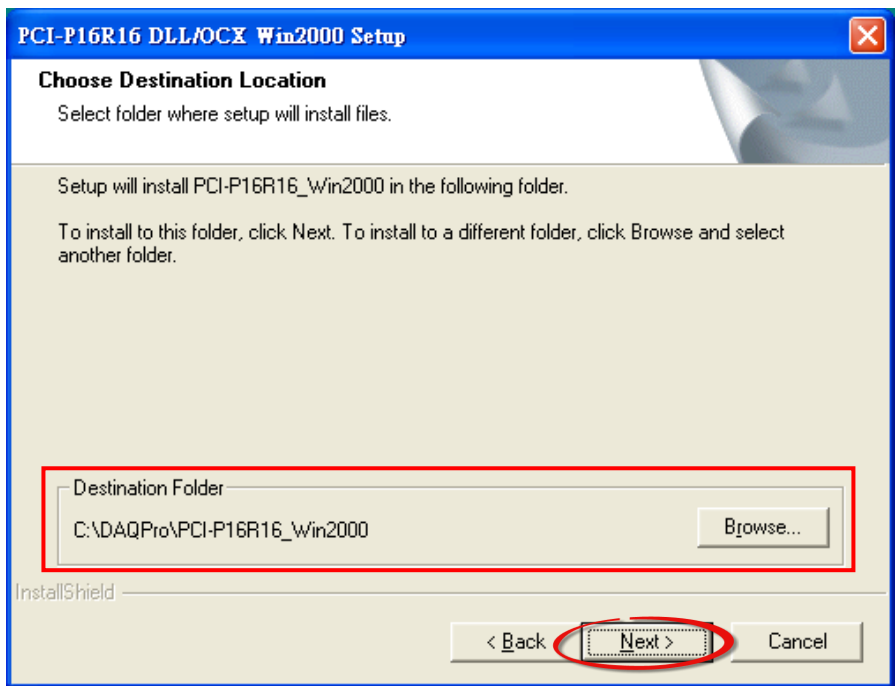


Step 1: Double-Click “PCI-P16R16_Win2k_xxxx.exe” to install driver.

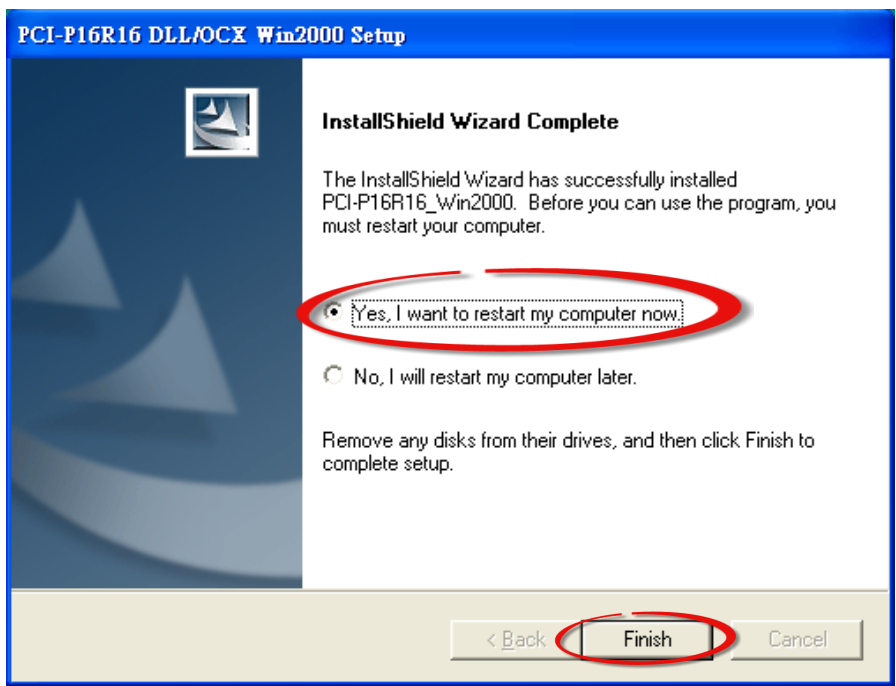
Step 2: Click the “**Next>**” button to start the installation on the “PCI-P16R16 DLL/OCX Win2000 Setup” window.



Step 3: Click the “**Next>**” button to install the driver into the **default** folder.



Step 4: Selection “**Yes, restart computer now**” and then click the “**Finish**” button.



2.3 Plug and Play Driver Installation



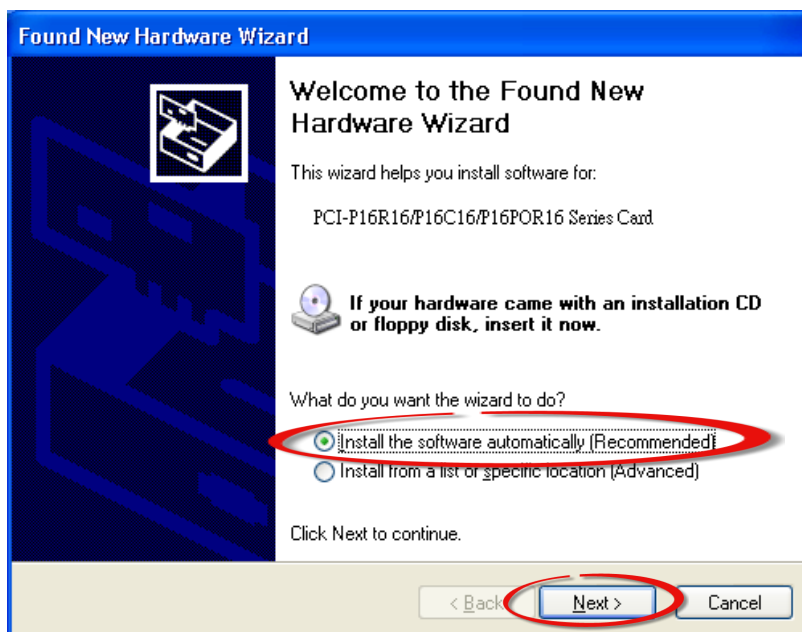
Note that if operating system is the Windows 95 and Windows 98, refer to “PCI-P16R16_PnP_Driver_Installation.pdf” for more detailed information.

Step 1: The system should find the new card and then continue to finish the Plug and Play steps.

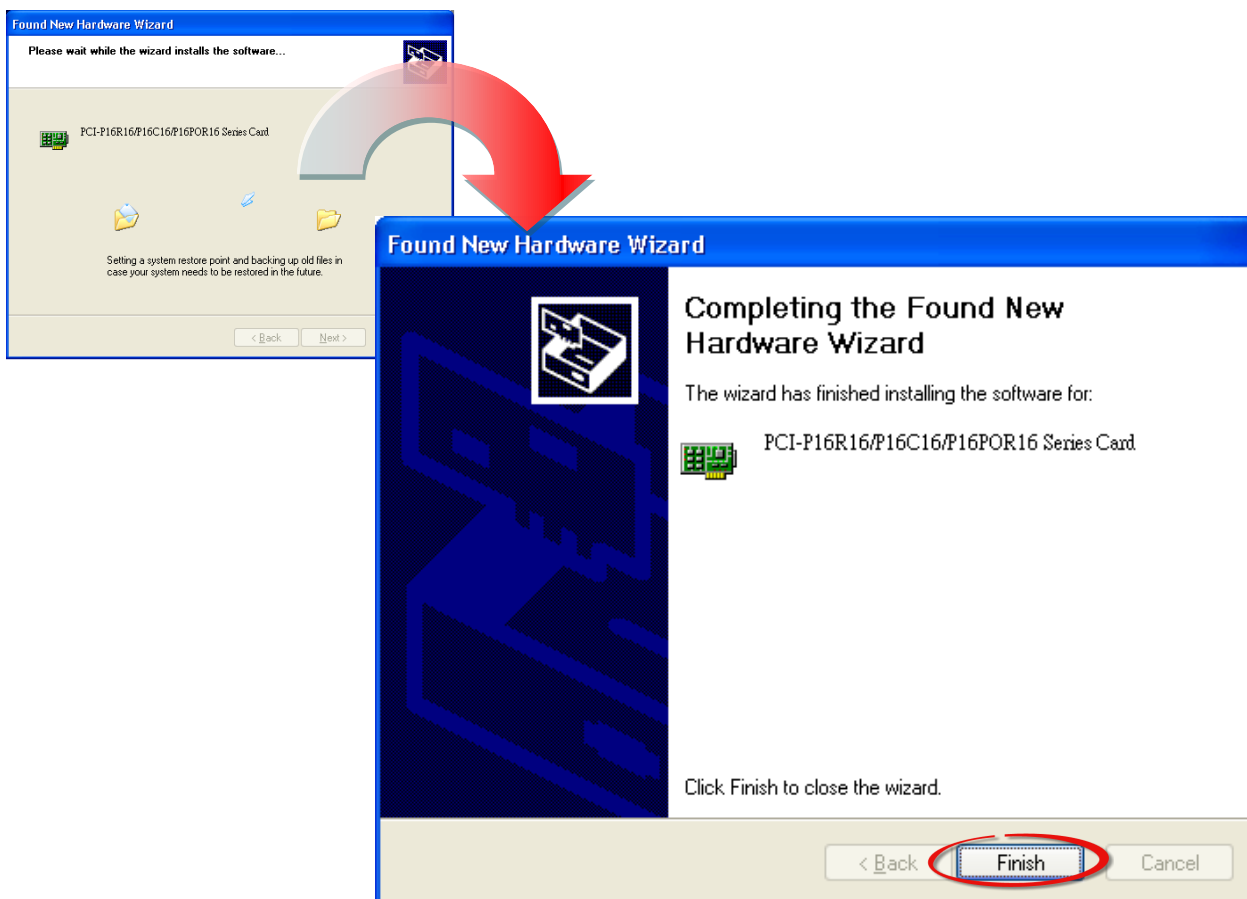
Note: Some operating system (such as Windows Vista/7) will find the new card and make it work automatically, so the Step2 to Step4 will be skipped.



Step 2: Select “Install the software automatically [Recommended]” and click the “Next>” button.



Step 3: Click the **“Finish”** button.



Step 4: Windows pops up **“Found New Hardware”** dialog box again.



2.4 Uninstalling the PCI-P16R16 Series Classic Driver

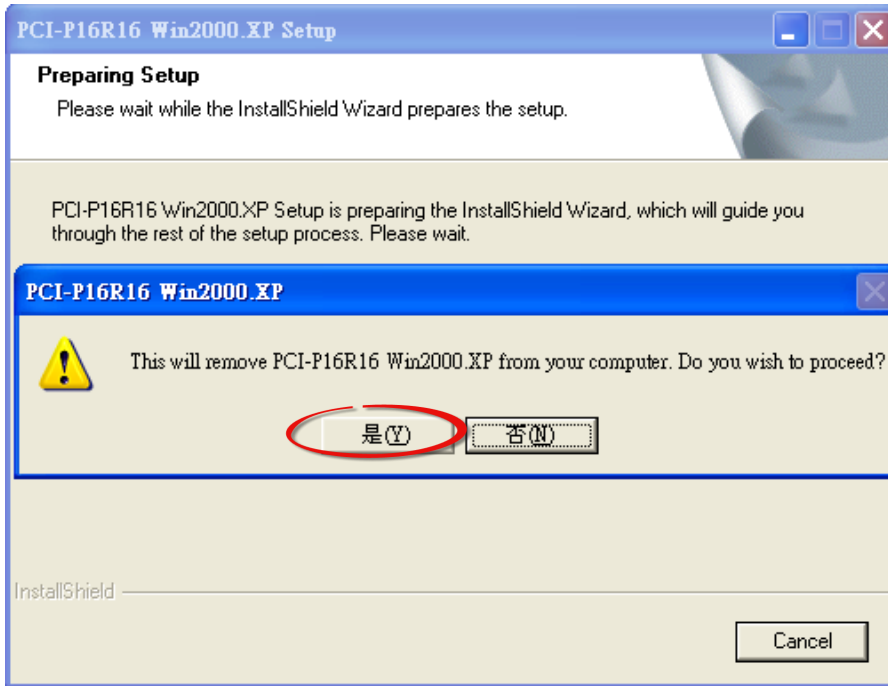
The ICP DAS PCI-P16R16 Series Classic Driver includes an uninstallation utility that allows you remove the software from your computer. To uninstall the software, follow the procedure described below:

Step 1: Select **Settings >> Control Panel >> Add/Remove Programs** from the Windows **Start** menu.

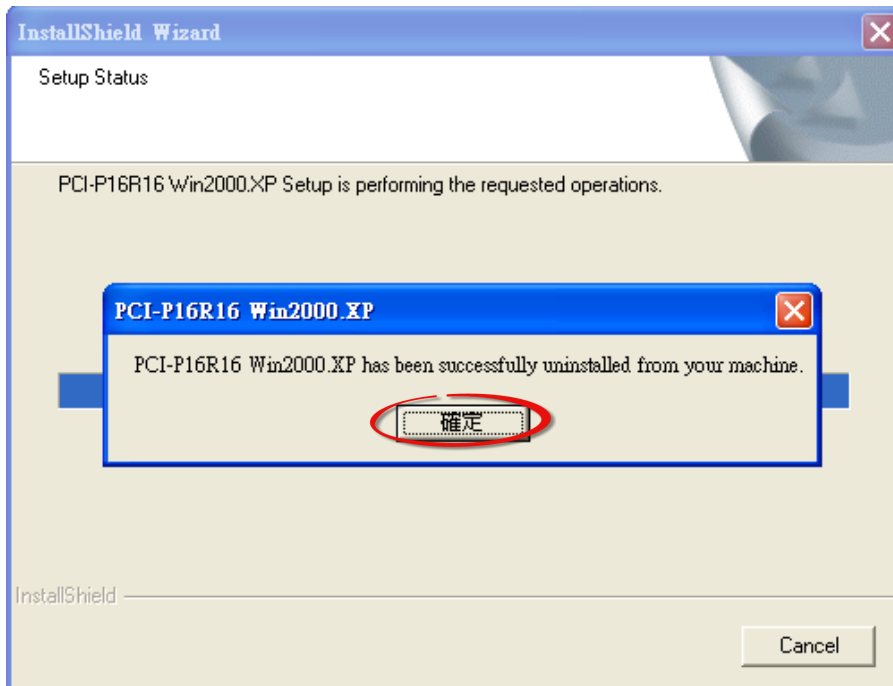
Step 2: Click the **Install/Uninstall** tab and highlight the item **PCI-P16R16 DLL/OCX Win2000** and then click the remove button.



Step 3: When the message box loads, click the **Yes(Y)** button to uninstall the software.



Step 4: After the uninstallation process is complete, a dialog box will be displayed to you that the driver was successfully removed. Click the **“OK”** button to finish the uninstallation process.



3. DLL Function Descriptions

All of the functions provided for PCI-P16R16 series card are listed below in Tables 3-2 to 3-5. This list of functions is expanded on in the text that follows. However, in order to make a clear and simplified description of the functions, the attributes of the input and output parameters for every function is indicated as [input] and [output] respectively, as shown in following Table 3-1. Furthermore, the error code of all functions supported by PCI-P16R16 series card is also listed in [Section 3.1](#).

➤ Table 3-1:

Keyword	Parameter must be set by the user before calling the function	Data/value from this parameter is retrieved after calling the function
[Input]	Yes	No
[Output]	No	Yes
[Input, Output]	Yes	Yes

Note: All of the parameters need to be allocated spaces by the user.

A function in P16R16.DLL (DLL for Windows OS) will be exactly the same prototype as P16R16H.LIB (huge mode library for DOS) and P16R16L.LIB (large mode library for DOS). It is convenient to develop applications under different platforms.

➤ Table 3-2: Test Functions Table

Section	Function Definition
3.2	Test Functions
	float PCI_FloatSub2 (float fA, float fB);
	short PCI_ShortSub2 (short nA, short nB);
	DWORD PCI_GetDllVersion (void);

➤ Table 3-3: Driver Initialization Functions Table

Section	Function Definition
3.3	Driver Initialization Functions
	WORD PCI_DriverInit (WORD *wTotalBoard);
	WORD PCI_DriverClose (void);
	WORD PCI_GetDriverVersion (WORD *wDriverVersion);
	WORD PCI_GetConfigAddressSpace (WORD wBoardNo, WORD *TypeID, WORD *wAddress0, WORD *wAddress1, WORD *wAddress2, WORD *wAddress3, WORD *wAddress4, WORD *wAddress5);

➤ Table 3-4: Digital I/O Functions for PCI-P16R16 Series Table

Section	Function Definition
3.4	Digital I/O Functions for PCI-P16R16 Series
	void CALLBACK P16R16_DO (WORD BoardAddr, WORD OutData);
	WORD CALLBACK P16R16_DI (WORD BoardAddr);

➤ Table 3-5: Digital I/O Functions for PCI-P8R8 Series Table

Section	Function Definition
3.5	Digital I/O Functions for PCI-P8R8 Series
	void CALLBACK P8R8_DO (WORD BoardAddr, WORD OutData);
	BYTE CALLBACK P8R8_DI (WORD BoardAddr);

3.1 Error Code Table

For the most errors, it is recommended to check:

1. Does the device driver installs successful?
2. Does the card have plugged?
3. Does the card conflicts with other device?
4. Close other applications to free the system resources.
5. Try to use another slot to plug the card.
6. Restart your system to try again.

➤ The return codes of DLLs are defined as follows:

Error Code	Error ID	Error String
0	NoError	OK (No Error)
1	DriverHandleError	Return handle is wrong when open device driver.
2	DriverCallError	Can't open the NAPPCI.VXD or NAPWNT.SYS in Windows
3	NotFoundBoard	Can't detect the board on the system
4	FindBoardError	Can't find the board on the system
5	ExceedBoardNumber	Invalidate board number (Valid range: 0 to TotalBoard -1)

3.2 Test Functions

PCI_FloatSub2

This function is used to perform the subtraction (as **fA** - **fB** in float data type), and is provided for testing DLL linkage purpose.

➤ **Syntax:**

float **PCI_FloatSub2**(float **fA**, float **fB**);

➤ **Parameters:**

fA

[Input] 4 bytes floating point value

fB

[Input] 4 bytes floating point value

➤ **Returns:**

The value of $fA - fB$

PCI_ShortSub2

This function is used to perform the subtraction (as **nA - nB** in short data type), and is provided for testing DLL linkage purposes.

➤ **Syntax:**

short PCI_ShortSub2(short **nA**, short **nB**);

➤ **Parameters:**

nA

[Input] 2 bytes short data type value

nB

[Input] 2 bytes short data type value

➤ **Returns:**

The value of $nA - nB$

PCI_GetDllVersion

This function is used to retrieve the version number of the P16R16.DLL.

➤ **Syntax:**

WORD **PCI_GetDllVersion**(void);

➤ **Parameters:**

None

➤ **Returns:**

DLL version information.

For example: If 200(hex) value is return, it means driver version is 2.00.

3.3 Driver Initialization Functions

PCI_DriverInit

This function is used to initialize the kernel driver (napwnt.sys for Windows NT/2K/XP/Vista/7, nappci.vxd for Windows 95/98). It is necessary to call on the function the first time you use this program.

➤ **Syntax:**

```
WORD PCI_DriverInit(WORD * wTotalBoard);
```

➤ **Parameters:**

**wTotalBoard*

[Input] Address of wTotalBoard.

When wTotalBoard = 1 → there is only one P16R16 or P8R8 card installed.

When wTotalBoard = 2 → there are two P16R16 or P8R8 cards installed

The possibility of combination as follows:

- One P16R16 and one P8R8 in PC
- Two P16R16 boards in PC
- Two P8R8 boards in PC

➤ **Returns:**

Refer to "[Section 3.1 Error Code Table](#)"

➤ **Example:**

The following is a Visual C++ source code sample:

```
LRESULT CALLBACK WndProc(HWND hwnd, UINT iMsg, WPARAM wParam, LPARAM lParam)
{
    static char cBuf[80];
    HDC          hdc;
    TEXTMETRIC  tm;
    PAINTSTRUCT ps;
    int         i;
    switch (iMsg)
    {
    case WM_CREATE : // window initial

        /******
        /* NOTICE: call PCI_DriverInit() to initialize the driver. */
        /******

        // Initial the device driver, and return the board number in the PC

        wInitialCode=PCI_DriverInit(&wTotalBoard);

        if( wInitialCode!=NoError )
        {
            MessageBox(hwnd,"No PCI card in this system !!!","PCI Card Error",MB_OK);
        }
        :
        :
        :
    }
}
```

PCI_DriverClose

Terminates the device driver (napwnt.sys for window NT/2K/XP/Vista/7, nappci.vxd for Windows 95/98). In DOS version, this function is provided just for uniformity or W32 program. It can only return a NoError.

- **Syntax:**
vio **PCI_DriverClose**(void);

- **Parameters:**
None

- **Returns:**
None

PCI_GetDriverVersion

Gets the version number of device driver (nappci.vxd for windows 95/98, napwnt.sys for Windows NT/2K/XP/Vista/7)

- **Syntax:**
WORD **PCI_GetDriverVersion**(WORD *wDriverVersion);

- **Parameters:**

*wDriverVersion
[Input] Address of wDriverVersion. wDriverVersion = 200 [hex] → Version 2.00

- **Returns:**
Refer to "[Section 3.1 Error Code Table](#)"

PCI_GetConfigAddressSpace

Reads configuration space for P16R16 and P8R8 series board then gets the content of Base Address0, Base Address1, Base Address2, Base Address3, Base Address4 and Base Address5.

➤ **Syntax:**

```
WORD PCI_GetConfigAddressSpace (WORD wBoardNo,  
                                DWORD *wTypeID,  
                                WORD *wAddress0,  
                                WORD *wAddress1,  
                                WORD *wAddress2,  
                                WORD *wAddress3,  
                                WORD *wAddress4,  
                                WORD *wAddress5  
                                );
```

➤ **Parameters:**

wBoardNo

[Input] The Board number for PCI-P16R16/P8R8 board. (Start from 0)

wTypeID

[Input] Address of wType as follow:

wTypeID	Model
0	This board is PCI-P16R16 series
1	This board is PCI-P8R8 series
2	This board is PCI-TMC12 series
3	This board is PIO-DA16
4	This board is PIO-DA8

wAddress0

wAddress1

wAddress2

wAddress3

wAddress4

wAddress5

[Output] The six base address of a PCI device will be stored in these variables.

➤ **Returns:**

Refer to "[Section 3.1 Error Code Table](#)"

3.4 Digital I/O Functions for PCI-P16R16 Series



Note that PCI-P16R16 series board includes PCI-P16R16(U), PCI-P16C16, PCI-P16POR16(U) and PEX-P16POR16i.

P16R16_DO

This function is used to send 16 bits of data to the DO port for PCI-P16R16 series boards.

➤ **Syntax:**

```
void P16R16_DO(DWORD BaseAddr, WORD OutData);
```

➤ **Parameters:**

BaseAddr

[Input] DO port base addresses.

OutData

[Input] 16 bits of data sent to the DO port.

➤ **Returns:**

None

➤ **Example:**

See pages 24 to 27

P16R16_DI

This function is used to read 16 bits of data from the PCI-P16R16's DI port.

➤ **Syntax:**

WORD **P16R16_DI**(DWORD **BaseAddr**);

➤ **Parameters:**

BaseAddr

[Input] DO port base addresses.

➤ **Returns:**

The 16-bit value read from DI port

➤ **Example:**

The following is a source code sample:

```
/* *****  
/* This program is developed by Turbo C 2.0  
/* *****  
/* Demo 1: One P16R16 card demo.  
/* *****  
  
#include "P16R16.H"  
int main()  
{  
    int    i,j;  
    WORD   nVal;  
    float  fVal;  
    WORD   wBoards,wRetVal,wVal;  
    WORD   wInData;  
    WORD   wTypeID;  
    WORD   wAddress0, wAddress1, wAddress2;  
    WORD   wAddress3, wAddress4, wAddress5;  
    WORD   P16R16_BaseAddress, P8R8_BaseAddress;  
    WORD   wP16R16No, wP8R8No;  
  
    clrscr();
```



```
/* initiating PCI-P16R16 card and detect how many P16R16/P8R8 card in PC */
wRetVal=PCI_DriverInit(&wBoards);
printf("Threr are %d P16R16 Cards in this PC\n",wBoards);

if( wBoards==0 )
{
    putch(0x07); putch(0x07); putch(0x07);
    printf("There are no P16R16/P8R8 card in this PC !!!\n");
    exit(0);
}

/* dump every P16R16/P8R8 card's configuration address space */
for(i=0; i<wBoards; i++)
{
    wRetVal=PCI_GetConfigAddressSpace(i,&wTypeID,
        &wAddress0,&wAddress1,&wAddress2,
        &wAddress3,&wAddress4,&wAddress5);

    if( !wRetVal )
    {
        switch( wTypeID )
        {
            case 0: printf("==> %02d Board Name:PCI-P16R16\n",i);
                P16R16_BaseAddress=wAddress2;
                wP16R16No++;
                break;
            case 1: printf("==> %02d Board Name:PCI-P8R8\n",i);
                P8R8_BaseAddress=wAddress2;
                wP8R8No++;
                break;
            case 2: printf("==> %02d Board Name:PCI-TMC12\n",i);
                break;
            case 3: printf("==> %02d Board Name:PCI-DA16\n",i);
                break;
            case 4: printf("==> %02d Board Name:PCI-DA8\n",i);
                break;
        }
        printf(" --> Addr0:%04x | Addr1:%04x | Addr2:%0x\n",
            wAddress0,wAddress1,wAddress2);
        printf(" --> Addr3:%04x | Addr4:%04x | Addr5:%0x\n",
            wAddress3,wAddress4,wAddress5);
    }
}
```

```
/* getting the Driver version */
wRetVal=PCI_GetDriverVersion(&wVal);
printf("Driver Version=%x\n",wVal);

/* call a function to test if exact calling LIB */
nVal=PCI_ShortSub2(1,2);
printf("PCI_ShortSub2(1,2) = %d\n",nVal);

/* call another function to test if exact calling LIB */
fVal=PCI_FloatSub2(1.0,2.0);
printf("PCI_FloatSub2(1.0,2.0) = %f\n",fVal);

if( wP16R16No<1 )
{
    patch(0x07);
    printf("Please plug one PCI-P16R16 in PC !!!\n");
    exit(0);
}

/*****
/***** PCI-P16R16 DI/DO demo *****/
/*****

printf("The PCI-P16R16 DO/DI testing !!!\n");

P16R16_DO(P16R16_BaseAddress,0x0000); /* Digital output */
delay(500); /* Delay a little time 500ms */

wInData=P16R16_DI(P16R16_BaseAddress); /* Digital input */
printf("Digital Output -> 0000H | Digital Input -> %04xH\n",wInData);

P16R16_DO(P16R16_BaseAddress,0xFFFF); /* Digital output */
delay(500); /* Delay a little time 500ms */

wInData=P16R16_DI(P16R16_BaseAddress); /* Digital input */
printf("Digital Output -> FFFFH | Digital Input -> %04xH\n",wInData);

P16R16_DO(P16R16_BaseAddress,0x5555); /* Digital output */
delay(500); /* Delay a little time 500ms */

wInData=P16R16_DI(P16R16_BaseAddress); /* Digital input */
printf("Digital Output -> 5555H | Digital Input -> %04xH\n",wInData);
```

```
P16R16_DO(P16R16_BaseAddress,0xAAAA);    /* Digital output */  
delay(500);    /* Delay a little time 500ms */  
  
wInData=P16R16_DI(P16R16_BaseAddress); /* Digital input */  
printf("Digital Output -> AAAAH | Digital Input -> %04xH\n",wInData);  
  
PCI_DriverClose();  
return 0;
```

3.5 Digital I/O Functions for PCI-P8R8 Series



Note that PCI-P8R8 series board includes PCI-P8R8(U) and PEX-P8POR8i.

P8R8_DO

This function is used to send 8 bits of data to the DO port for PCI-P8R8 series boards.

➤ **Syntax:**

```
void P8R8_DO(DWORD BaseAddr, WORD OutData);
```

➤ **Parameters:**

BaseAddr

[Input] DO port base addresses.

OutData

[Input] 8 bits of data sent to the DO port..

➤ **Returns:**

None

➤ **Example:**

See pages 29 to 32

P8R8_DI

This function is used to read 8 bits of data from the PCI-P8R8's DI port.

➤ **Syntax:**

WORD **P8R8_DI**(DWORD **BaseAddr**);

➤ **Parameters:**

BaseAddr

[Input] DO port base addresses.

➤ **Returns:**

The 8-bit value read from DI port

➤ **Example:**

The following is a source code sample:

```
/*  
/* This program is developed by Turbo C 2.0 */  
/*  
/* Demo 2: One P8R8 card demo. */  
/*  
#include "P16R16.H"  
int main()  
{  
    int    i,j;  
    WORD   nVal;  
    float  fVal;  
    WORD   wBoards,wRetVal,wVal;  
    WORD   wInData;  
    WORD   wTypeID;  
    WORD   wAddress0,wAddress1,wAddress2;  
    WORD   wAddress3,wAddress4,wAddress5;  
    WORD   P16R16_BaseAddress,P8R8_BaseAddress;  
    WORD   wP16R16No,wP8R8No;  
  
    clrscr();
```

```
/* initiating PCI-P16R16 card and detect how many P16R16/P8R8 card in PC */
wRetVal=PCI_DriverInit(&wBoards);
printf("Threr are %d PCI-P8R8/P16R16 Cards in this PC\n",wBoards);
if( wBoards==0 )
{
    putchar(0x07); putchar(0x07); putchar(0x07);
    printf("There are no P8R8/P16R16 card in this PC !!!\n");
    exit(0);
}

/* dump every P16R16/P8R8 card's configuration address space */
for(i=0; i<wBoards; i++)
{
    wRetVal=PCI_GetConfigAddressSpace(i,&wTypeID,
        &wAddress0,&wAddress1,&wAddress2,
        &wAddress3,&wAddress4,&wAddress5);
    if( !wRetVal )
    {
        switch( wTypeID )
        {
            case 0: printf("==> %02d Board Name:PCI-P16R16\n",i);
                P16R16_BaseAddress=wAddress2;
                wP16R16No++;
                break;
            case 1: printf("==> %02d Board Name:PCI-P8R8\n",i);
                P8R8_BaseAddress=wAddress2;
                wP8R8No++;
                break;
            case 2: printf("==> %02d Board Name:PCI-TMC12\n",i);
                break;
            case 3: printf("==> %02d Board Name:PCI-DA16\n",i);
                break;
            case 4: printf("==> %02d Board Name:PCI-DA8\n",i);
                break;
        }
        printf(" --> Addr0:%04x | Addr1:%04x | Addr2:%0x\n",
            wAddress0,wAddress1,wAddress2);
        printf(" --> Addr3:%04x | Addr4:%04x | Addr5:%0x\n\n",
            wAddress3,wAddress4,wAddress5);
    }
}
```

```
/* getting the Driver version */
wRetVal=PCI_GetDriverVersion(&wVal);
printf("Driver Version=%x\n",wVal);

/* call a function to test if exact calling LIB */
nVal=PCI_ShortSub2(1,2);
printf("PCI_ShortSub2(1,2) = %d\n",nVal);

/* call another function to test if exact calling LIB */
fVal=PCI_FloatSub2(1.0,2.0);
printf("PCI_FloatSub2(1.0,2.0) = %f\n",fVal);

if( wP8R8No<1 )
{
    putchar(0x07);
    printf("Please plug one PCI-P8R8 in PC !!!\n");
    exit(0);
}

/*****
/*****  PCI-P8R8 DI/DO demo  *****/
/*****

printf("The PCI-P8R8 DO/DI testing !!!\n");
P8R8_DO(P8R8_BaseAddress,0x0000);    /* Digital output */
delay(500);                          /* Delay a little time */

wInData=P8R8_DI(P8R8_BaseAddress);    /* Digital input  */
printf("Digital Output -> 0000H   |   Digital Input -> %04xH\n",wInData);

P8R8_DO(P8R8_BaseAddress,0xFFFF);    /* Digital output */
delay(500);                          /* Delay a little time */

wInData=P8R8_DI(P8R8_BaseAddress);    /* Digital input  */
printf("Digital Output -> FFFFH   |   Digital Input -> %04xH\n",wInData);

P8R8_DO(P8R8_BaseAddress,0x5555);    /* Digital output */
delay(500);                          /* Delay a little time */
```

```
wInData=P8R8_DI(P8R8_BaseAddress);    /* Digital input */  
printf("Digital Output -> 5555H | Digital Input -> %04xH\n",wInData);  
  
P8R8_DO(P8R8_BaseAddress,0xAAAA);    /* Digital output */  
delay(500);                            /* Delay a little time */  
  
wInData=P8R8_DI(P8R8_BaseAddress);    /* Digital input */  
printf("Digital Output -> AAAAH | Digital Input -> %04xH\n",wInData);  
  
PCI_DriverClose();  
return 0;  
}
```


4. Demo Programs

4.1 For Microsoft Windows

During the installation process for the DLL driver, the correct kernel driver will be registered in the operation system and the DLL driver and demo programs will be copied to the correct location based on the driver software package that was selected (Win95/98/NT/2K or 32-/64-bit Win XP/2003/Vista/7/8). After installing the driver, the related demo programs development library and declaration header files for the various development environments will be available in the folders indicated below. **Note that none of the demo programs will function correctly if the DLL driver has not been properly installed.**

The demo programs for PCI-P16R16 Series Classic Driver can be found in the following locations:



CD:\NAPDOS\PCI\PCI-P16R16\DLL_OCX\Demo\



http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pci-p16r16/dll_ocx/demo/

⊕ BCB3 → for Borland C++ Builder 3
P16R16.H → Header files
P16R16.LIB → Linkage library for BCB only

⊕ Delphi3 → for Delphi 3
P16R16.PAS → Declaration files

⊕ VC6 → for Visual C++ 6
P16R16.H → Header files
P16R16.LIB → Linkage library for VC only

⊕ VB6 → for Visual Basic 6
P16R16.BAS → Declaration files

⊕ VB.NET2005 → for VB.NET2005
P16R16.vb → Visual Basic Source files

⊕ CSharp2005 → for C#.NET2005
P16R16.cs → Visual C# Source files

The available demo programs available are as follows:

The list of demo programs

DIO Digital Input and Digital Output demo

4.2 For DOS

ICP DAS provides a range of demo programs that can be used in a DOS environment, together with the source code for the library and are designed for a variety of development environments.

The relevant demo programs can be found in the following locations:



CD:\NAPDOS\PCI\PCI-P16R16\DOS\PP16R16\



<http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pci-p16r16/dos/pp16r16/>

- Demo code and Library for Borland C++
 - ...\P16R16\BC\HUGE\DEMO → huge mode demo programs
 - ...\P16R16\BC\HUGE\LIB → huge mode library, P16R16H.LIB
 - ...\P16R16\BC\LARGE\DEMO → large mode demo programs
 - ...\P16R16\BC\LARGE\LIB → large mode library, P16R16L.LIB

- Demo code and Library for MSC
 - ...\P16R16\MSC\HUGE\DEMO → huge mode demo programs
 - ...\P16R16\MSC\HUGE\LIB → huge mode library, P16R16H.LIB
 - ...\P16R16\MSC\LARGE\DEMO → large mode demo programs
 - ...\P16R16\MSC\LARGE\LIB → large mode library, P16R16L.LIB

- Demo code and Library for TC
 - ...\P16R16\TC\HUGE\DEMO → huge mode demo programs
 - ...\P16R16\TC\HUGE\LIB → huge mode library, P16R16H.LIB
 - ...\P16R16\TC\LARGE\DEMO → large mode demo programs
 - ...\P16R16\TC\LARGE\LIB → large mode library, P16R16H.LIB

The available demo programs available are as follows:

The list of demo programs

Demo1	DI/DO demo for one PCI-P16R16 series card
Demo2	DI/DO demo for one PCI-P8R8 series card
Demo3	DI/DO demo for one PCI-P16R16 and one PCI-P8R8 series cards
Demo4	DI/DO demo for two PCI-P16R16 series cards

For further information regarding the demo programs and library files, see the readme.txt file that can be found in each of the DEMO and LIB sub-directories.

5. Problems Report

Technical support is available at no charge as described below. The best way to report problems is to send electronic mail to Service@icpdas.com or Service.icpdas@gmail.com on the Internet.

When reporting problems, please include the following information:

1. Is the problem reproducible? If so, how?
2. What kind and version of **platform** that you using? For example, Windows 98, Windows 2000 or 32-bit Windows XP/2003/Vista/7/8.
3. What kinds of our **products** that you using? Please see the product's manual.
4. If a dialog box with an **error message** was displayed, please include the full text of the dialog box, including the text in the title bar.
5. If the problem involves **other programs** or **hardware devices**, what devices or version of the failing programs that you using?
6. **Other comments** relative to this problem or **any suggestions** will be welcomed.

After we had received your comments, we will take about two business days to test the problems that you said. And then reply as soon as possible to you. Please check that if we had received you comments? And please keeps contact with us.