PCI-P16R16 Series

Software Manual

[ver. 1.0, Apr. 2011]

Supports

Board includes PCI-P8R8, PCI-P16R16, PCI-P8POR8, PCI-P16POR16, PCI-P16C16, PEX-P8POR8i and PEX-P16POR16i. These cards support P16R16.DLL driver.

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

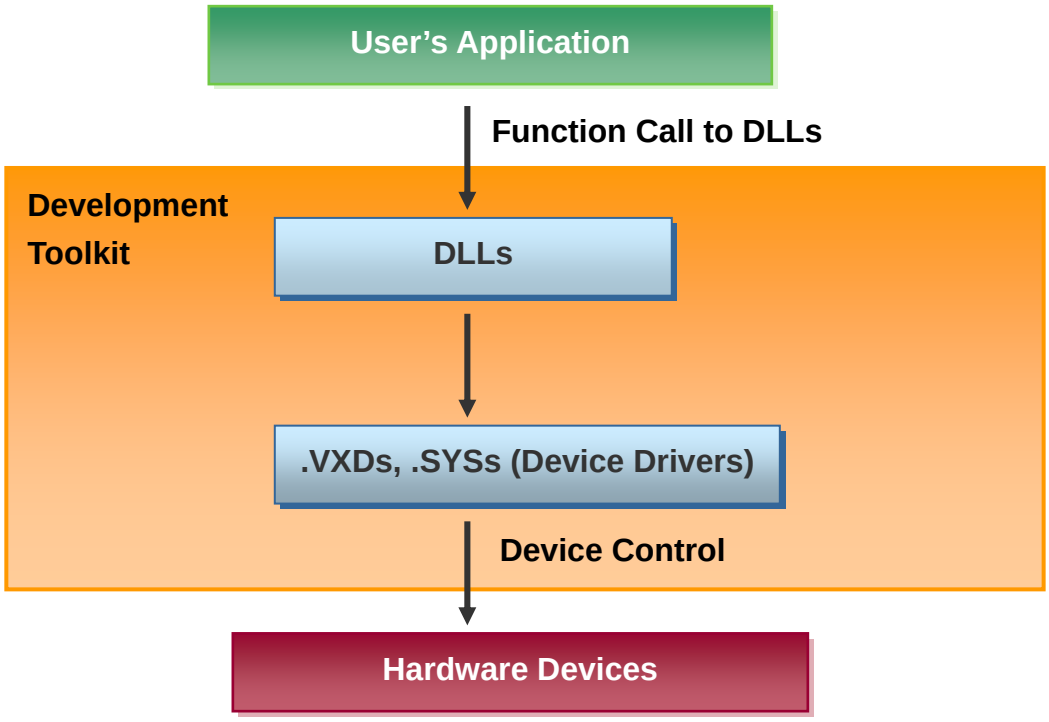
Trademark

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

Table of Contents

1. DLL Function Description

The PCI-P16R16.DLL driver is the API function library for the PCI-P16R16 series card that works the Windows 95/98/ME/NT/2000 and 32-bit Windows XP/2003/Vista/7 systems. The application structure is illustrated in the figure below. User's application programs that calls P16R16.DLL library in user mode can be developed using various tools, such as VB, VC, Delphi, Borland C++ Builder, VB.NET, VC.NET and C#. The DLL driver then calls the P16R16.sys file in order to access the hardware.



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/manual/

- **Software Installation Guide.pdf:**
  Describes how to install the software package under Windows 95/98/ 2000/XP/2003/Vista/2008/7(32-bit).
  http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/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/

- **PCI-P8R8_P16R16_Series_Hardware_Manual.pdf:**
  PCI-P16R16 series card hardware manual for PCI-P8R8/P16R16, PCI-P8POR8/P16POR16, PCI-P16C16 and PEX-P8POR8i/P16POR16i.
  http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pci-p16r16/manual/

1.2 Declaration Files

After the drivers are installed, the relevant demo programs, development libraries and declaration header files for the different development environments will be available in the following locations.

For detailed PCI-P16R16 series Windows driver installed information, please refer to Quick Start Guide (CD:\NAPDOS\PCI\PCI-P16R16\Manual\QuickStart\).
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pci-p16r16/manual/quickstart/

The demo program for Windows is contained in:
CD:\NAPDOS\PCI\PCI-P16R16\DLL_OCX\Demo\
http://ftp.icpdas.com/pub/cd/iocard/pci/napdos/pci/pci-p16r16/dll_ocx/demo/

- BCB 3 → for Borland C++ Builder 4
  P16R16.H → Header files
  P16R16.LIB → Linkage library for BCB only
- Delphi3 → for Delphi 3
  P16R16.PAS → Declaration files
- VB6 → for Visual Basic 6
  P16R16.BAS → Declaration files
- VC6 → for Visual C++ 6
  P16R16.H → Header files
  P16R16.LIB → Linkage library for VC6 only
- VB.NET2005 → for VB.NET2005
  P16R16.vb → Visual Basic Source files
- CSharp2005 → for C#.NET2005
  P16R16.cs → Visual C# Source files

2. Function Descriptions

In order to simplify and clarify the description, the attribute of the input and output parameters of the function is indicated as [input] and [output], respectively, as shown in the following table.

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.

- The return codes of DLLs are defined as follows:

// return code

Error Code	Error ID
0	NoError
1	DriverHandleError
2	DriverCallError
3	NotFoundBoard
4	FindBoardError
5	ExceedBoardNumber

■ The defined DLL are given as follows:

Reference	Function Definition
Sec. 2.2	**Functions of Test**
Sec. 2.2.1	float PCI_FloatSub2(float fA, float fB);
Sec. 2.2.2	short PCI_ShortSub2(short nA, short nB);
Sec. 2.2.3	DWORD PCI_GetDllVersion(void);
Sec. 2.3	**Driver Initialization Functions**
Sec. 2.3.1	WORD PCI_DriverInit(WORD *wBoards);
Sec. 2.3.2	WORD PCI_DriverClose(void);
Sec. 2.3.3	WORD PCI_GetDriverVersion(WORD *wDriverVersion);
Sec. 2.3.4	WORD PCI_GetConfigAddressSpace(WORD wBoardNo, WORD *TypeID, WORD *wAddress0, WORD *Waddress1, WORD *wAddress2, WORD *wAddress3, WORD *wAddress4, WORD *wAddress5);
Sec. 2.4	**Digital I/O Functions for PCI-P16R16 Series**
Sec. 2.4.1	void CALLBACK P16R16_DO(WORD BoardAddr, WORD OutData);
Sec. 2.4.2	WORD CALLBACK P16R16_DI(WORD BoardAddr);
Sec. 2.5	**Digital I/O Functions for PCI-P8R8 Series**
Sec. 2.5.1	void CALLBACK P8R8_DO(WORD BoardAddr, WORD OutData);
Sec. 2.5.2	BYTE CALLBACK P8R8_DI(WORD BoardAddr);

2.1 P16R16.H

```
#define EXPORTS extern "C" __declspec (dllimport)

// return code
#define NoError                 0
#define DriverHandleError       1
#define DriverCallError         2
#define NotFoundBoard           3
#define FindBoardError          4
#define ExceedBoardNumber       5

// define Type Name ID
#define TYPE_P16R16             0
#define TYPE_P8R8               1
#define TYPE_TMC12              2
#define TYPE_DA16               3
#define TYPE_DA8                4

EXPORTS float  CALLBACK PCI_FloatSub2(float fA, float fB);
EXPORTS short  CALLBACK PCI_ShortSub2(short nA, short nB);
EXPORTS WORD   CALLBACK PCI_GetDllVersion(void);

EXPORTS WORD   CALLBACK PCI_DriverInit(WORD *wTotalBoards);
EXPORTS void   CALLBACK PCI_DriverClose(void);
EXPORTS WORD   CALLBACK PCI_GetDriverVersion(WORD *wVxdVersion);

EXPORTS WORD   CALLBACK PCI_GetConfigAddressSpace
           (WORD wBoardNo, WORD *TypeID,
            WORD *wAddress0, WORD *wAddress1, WORD *wAddress2,
           WORD *wAddress3, WORD *wAddress4, WORD *wAddress5);

EXPORTS WORD   CALLBACK PCI_WhichBoardActive(void);
EXPORTS void   CALLBACK P16R16_DO(WORD BaseAddr, WORD OutData);
EXPORTS WORD   CALLBACK P16R16_DI(WORD BaseAddr);
EXPORTS void   CALLBACK P8R8_DO(WORD BaseAddr, WORD OutData);
EXPORTS BYTE   CALLBACK P8R8_DI(WORD BaseAddr);
```

2.2 Functions of Test

2.2.1 PCI_FloatSub2

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

- **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

2.2.2 PCI_ShortSub2

- **Description:**
  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

2.2.3 PCI_GetDllVersion

- **Description:**
  This functions is used to retrieve the version number of the P16R16.DLL driver

- **Syntax:**
  WORD **PCI_GetDllVersion**(void)

- **Parameters:**
  void

- **Returns:**
  201 (hex) for version 2.01

2.3 Driver Initialization Functions

2.3.1 PCI_DriverInit

- **Description:**
  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_Driverlinit** (WORD ***wTolalBoard**)

- **Parameters:**

***wTotalBoard**	[Input]	Address of wTotalBoard. When wTotalBoard = 1: either P16R16 or P8R8 in PC. When wTotalBoard = 2: possibility of combination → * One P16R16 and one P8R8 in PC. * Two P16R16 boards in PC. * Two P8R8 boards in PC.

- **Returns:**

NoError	OK
NotFoundBoard	Can't detect the existence of P16R16/P8R8
DriverCallError	Can't open the NAPPCI.Vxd in Windows 95/98. Can't open the NAPWNT.SYS in Windows NT/2K/XP/Vista/7.
DriverHandleError	Return handle is wrong when open device driver.

- **Demo Program:**

[VC example]

```
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);
    }
        :
        :
        :
   }
```

2.3.2 PCI_DriverClose

- **Description:**
  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:**
  void **PCI_DriverClose**(void)

- **Parameters:**
  void

- **Returns:**
  void

2.3.3 PCI_GetDriverVersion

- **Description:**
  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:**

NoError	OK
NotFoundBoard	Can't detect the existence of P16R16/P8R8
DriverCallError	Can't open the NAPPCI.Vxd in Windows 95/98. Can't open the NAPWNT.SYS in Windows NT/2K/XP/Vista/7.
DriverHandleError	Return handle is wrong when open device driver.

2.3.4 PCI_GetCongfigAddressSpace

■ **Description:**

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

■ **Syntax:**

WORD **PCI_GetConfigAddressSpace** (WORD **wBoadNo**,
WORD ***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: 0: this board is PCI_P16R16 1:this board is PCI_P8R8 2:this board is PCI_TMC12 3:this board is PIO_DA16 4:this board is PIO_DA8
***wAddress0**	[Output]	The six base address of a PCI device will be stored in these variables.
***wAddress1**		
***wAddress2**		
***wAddress3**		
***wAddress4**		
***wAddress5**		

■ **Returns:**

NoError	OK
FindBoardError	Can't detect the existence of P16R16/P8R8
ExceedBoardError	Can't open the NAPPCI.Vxd in Windows 95/98

2.4 Digital I/O for PCI-P16R16 Series

PCI-P16R16 series board includes PCI-P16R16, PCI-P16C16, PCI-P16POR16 and PEX-P16POR16i.

2.4.1P16R16_DO

- **Description:**
  Sends 16-bit data to D/O port of the PCI-P16R16.

- **Syntax:**
  Void **P16R16_DO** (WORD **BaseAddr,** WORD **OutData**)

- **Parameters:**

BaseAddr	[Input]	D/O port base address
OutData	[Input]	the 16-bit data sent to D/O port

- **Returns:**
  void

- **Demo Program:**
  Please refer to page 16.

2.4.2 P16R16_DI

- **Description:**
  Reads 16-bit data from PCI-P16R16's D/I port.

- **Syntax:**
  WORD **P16R16_DI** (WORD **BaseAddr**)

- **Parameters:**

BaseAddr	[Input]	D/O port base address

- **Returns:**
  The 16-bit value read from D/I port

■ Demo Program:

```
/*********************************************************/
/* 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 PCI-P16R16/P8R8 Cards in this PC,
  tally.\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\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 )
{
   putch(0x07);
   printf("Please plug one PCI-P16R16 in PC !!!\n");
   exit(0);
}

/*************************************/
/*****  PCI-P16R16 DO/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;
    }
```

2.5 Digital I/O for PCI-P8R8 Series

PCI-P8R8 series board includes PCI-P8R8, PCI-P8POR8 and PEX-P8POR8i.

2.5.1 P8R8_DO

- **Description:**
  Sends 8-bit data to D/O port of the PCI-P8R8.

- **Syntax:**
  Void **P8R8_DO** (WORD **BaseAddr,** WORD **OutData**)

- **Parameters:**

BaseAddr	[Input]	D/O port base address
OutData	[Input]	the 8-bit data sent to D/O port

- **Returns:**
  void

- **Demo Program:**
  Please refer to page 19.

2.5.2 P8R8_DI

- **Description:**
  Reads 8-bit data from D/I port of the PCI-P8R8.

- **Syntax:**
  Void **P8R8_DO** (WORD **BaseAddr,** WORD **OutData**)

- **Parameters:**

BaseAddr	[Input]	D/O port base address

- **Returns:**
  void

- **Demo Program:**

```
/***********************************************/
/*   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();

/* initiaing 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 )
  {
    putch(0x07);  putch(0x07);  putch(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 )
  {
     putch(0x07);
     printf("Please plug one PCI-P8R8 in PC !!!\n");
     exit(0);
  }

  /***********************************/
  /*****  PCI-P8R8 DO/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;
}
```

2.6 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/2008/7.
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 test 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.

E-mail: Service@icpads.com
Service.icpdas@ gmail.com

Web Site: http://www.icpdas.com
http://www.icpdas.com.tw