PISO-813

Software Manual

[For Windows 95/98/NT/2000/ME/XP]

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 damage 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, ICP DAS assumes no responsibility for its use, or for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Trademark

The names used for identification only maybe registered trademarks of their respective companies.

License

The user can use, modify and backup this software **on a single machine.** The user may not reproduce, transfer or distribute this software, or any copy, in whole or in part.

Table of Contents

1. INTRODUCTION 3

1.1 REFERENCE 3

1.2 A/D GAIN CODE TABLE 4

2. DECLARATION FILES 5

2.1 PISO813.H 6

2.2 PISO813.BAS 8

2.3 PISO813.PAS 11

3. FUNCTION DESCRIPTIONS 15

3.1 FUNCTIONS OF TEST 15

3.1.1 PISO813_GetDllVersion 15

3.1.2 PISO813_ShortSub 16

3.1.3 PISO813_FloatSub 16

3.2 FUNCTIONS OF I/O 17

3.2.1 PISO813_OutputByte 17

3.2.2 PISO813_InputByte 17

3.2.3 PISO813_OutputWord 18

3.2.4 PISO813_InputWord 18

3.3 FUNCTIONS OF DRIVER 19

3.3.1 PISO813_GetDriverVersion 19

3.3.2 PISO813_DriverInit 19

3.3.3 PISO813_DriverClose 20

3.3.4 PISO813_GetConfigAddressSpace 20

3.4 A/D FUNCTIONS 21

3.4.1 PISO813_SetChGain 21

3.4.2 PISO813_AD2F 21

3.4.3 PISO813_AD_Hex 22

3.4.4 PISO813_AD_Float 22

3.4.5 PISO813_ADs_Hex 23

3.4.6 PISO813_ADs_Float 24

4. PROGRAM ARCHITECTURE 25

5. PROBLEMS REPORT 26

1.Introduction

The software is a collection of digital I/O and Analog-Input subroutines for PISO-813 add-on cards for Windows 95/98 and Windows NT 4.0 Applications. These subroutines are written with C language and perform a variety of digital I/O and Analog-Input operations.

The subroutines in PISO813.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 VC++, VB, Delphi, and BORLAND C++ Builder easily. To speed-up your developing process, some demonstration source program are provided.

1.1Reference

Please refer to the following user manuals:

- **PnPInstall.pdf:**
  Describes how to install the PnP (Plug and Play) driver for PCI card under Windows 95/98.

- **SoftInst.pdf:**
  Describes how to install the software package under Windows 95/98/NT.

- **CallDll.pdf:**
  Describes how to call the DLL functions with VC++5, VB5, Delphi3 and Borland C++ Builder 3.

- **ResCheck.pdf:**
  Describes how to check the resources I/O Port address, IRQ number and DMA number for add-on cards under Windows 95/98/NT.

1.2A/D Gain Code table

PISO-813 Bipolar mode GAIN CONTROL CODE TABLE

- JP2 : Bipolar

GAIN	**JP1 : 10V Input Range**	**JP1 : 20V Input Range**	GAIN2	GAIN1	GAIN0	**Gain-Code**
1	**± 5V**	**± 10V**	0	0	0	**0x0**
2	**± 2.5V**	**± 5V**	0	0	1	**0x1**
4	**± 1.25V**	**± 2.5V**	0	1	0	**0x2**
8	**± 0.625V**	**± 1.25V**	0	1	1	**0x3**
16	Not use	**± 0.625**	1	0	0	**0x4**

PISO-813 Unipolar mode GAIN CONTROL CODE TABLE

- JP2 : Unipolar

GAIN	**JP1 : 10V Input Range**	JP1 : 20V Input Range	GAIN2	GAIN1	GAIN0	**Gain-Code**
1	**0~10V**	Not use	0	0	0	**0**
2	**0~5V**	Not use	0	0	1	**1**
4	**0~2.5V**	Not use	0	1	0	**2**
8	**0~1.25V**	Not use	0	1	1	**3**
16	**0~0.625V**	Not use	1	0	0	**4**

2. Declaration Files

```
|--\Driver                  ← some device driver
    |--\BCB3                ← for Borland C++ Builder 3
    |    |--\PISO813.H      ← Header file
    |   +--\PISO813.LIB     ← Linkage library for BCB3 only
    |
    |--\Delphi3             ← for Delphi 3
    |   +--\PISO813.PAS     ← Declaration file
    |
    |--\VB5                 ← for Visual Basic 5
    |   +--\PISO813.BAS     ← Declaration file
    |
    +--\VC5                 ← for Visual C++ 5
        |--\PISO813.H       ← Header file
        +--\PISO813.LIB     ← Linkage library for VC5 only
```

2.1 PISO813.H

```
#ifdef __cplusplus
    #define EXPORTS extern "C" __declspec (dllimport)
#else
    #define EXPORTS
#endif
/************ define PISO813 relative address **********************/
#define PISO813_AD_LO        0xd0    // Analog to Digital, Low Byte
#define PISO813_AD_HI        0xd4    // Analog to Digital, High Byte
#define PISO813_SET_CH       0xE0    // channel selecting
#define PISO813_SET_GAIN     0xE4    // PGA gain code
#define PISO813_SOFT_TRIG    0xF0    // A/D trigger control register
/****** define the gain mode ********/
#define PISO813_BI_1         0x00
#define PISO813_BI_2         0x01
#define PISO813_BI_4         0x02
#define PISO813_BI_8         0x03
#define PISO813_BI_16        0x04

#define PISO813_UNI_1        0x00
#define PISO813_UNI_2        0x01
#define PISO813_UNI_4        0x02
#define PISO813_UNI_8        0x03
#define PISO813_UNI_16       0x04
/*************    return code    *************/
#define PISO813_NoError                     0
#define PISO813_DriverOpenError             1
#define PISO813_DriverNoOpen                2
#define PISO813_GetDriverVersionError       3
#define PISO813_CallDriverError             4
#define PISO813_FindBoardError              5
#define PISO813_ExceedBoardNumber           6
#define PISO813_TimeOutError                0xffff
#define PISO813_AdError2                    -100.0
// ID
#define PISO_813                            0x800A00
```

```
// Test functions
EXPORTS float       CALLBACK PISO813_FloatSub(float fA, float fB);
EXPORTS short       CALLBACK PISO813_ShortSub(short nA, short nB);
EXPORTS WORD        CALLBACK PISO813_GetDllVersion(void);

// Driver functions
EXPORTS WORD        CALLBACK PISO813_DriverInit(void);
EXPORTS void        CALLBACK PISO813_DriverClose(void);
EXPORTS WORD        CALLBACK PISO813_SearchCard(WORD *wBoards,
    DWORD dwPIOCardID);
EXPORTS WORD        CALLBACK PISO813_GetDriverVersion(
    WORD *wDriverVersion);
EXPORTS WORD        CALLBACK PISO813_GetConfigAddressSpace(
    WORD  wBoardNo, DWORD *wAddrBase, WORD *wIrqNo,
    WORD *wSubVendor, WORD *wSubDevice, WORD *wSubAux,
    WORD *wSlotBus,   WORD *wSlotDevice);
// DIO functions
EXPORTS void        CALLBACK PISO813_OutputWord(
    DWORD wPortAddress, DWORD wOutData);
EXPORTS void        CALLBACK PISO813_OutputByte(DWORD wPortAddr,
    WORD bOutputValue);
EXPORTS DWORD   CALLBACK PISO813_InputWord(
    DWORD wPortAddress);
EXPORTS WORD        CALLBACK PISO813_InputByte(DWORD wPortAddr);

// AD functions
EXPORTS WORD        CALLBACK PISO813_SetChGain(
    DWORD wAddrBase, WORD wChannel , WORD wGainCode);
EXPORTS WORD        CALLBACK PISO813_AD_Hex(DWORD wAddrBase);
EXPORTS WORD        CALLBACK PISO813_ADs_Hex(DWORD wAddrBase,
    WORD  *wBuffer, DWORD dwDataNo);
EXPORTS float       CALLBACK PISO813_AD_Float(DWORD wAddrBase,
    WORD wJump20v, WORD wBipolar);
EXPORTS float       CALLBACK PISO813_ADs_Float(DWORD wAddrBase,
    WORD wJump20v, WORD wBipolar, float *fBuffer, DWORD dwDataNo);
EXPORTS float       CALLBACK PISO813_AD2F(WORD whex,
    WORD wGainCode,  WORD wJump20v,  WORD wBipolar);
```

2.2 PISO813.BAS

```
Attribute VB_Name = "PISO813"

'/************ define PISO813 relative address **********************/
Global Const PISO813_AD_LO          = &HD0 'Analog to Digital, Low Byte
Global Const PISO813_AD_HI          = &HD4 'Analog to Digital, High Byte
Global Const PISO813_SET_CH         = &HE0 'channel selecting
Global Const PISO813_SET_GAIN       = &HE4 'PGA gain code
Global Const PISO813_SOFT_TRIG      = &HF0 'A/D trigger control register
'/****** define the gain mode ********/
Global Const PISO813_BI_1           = &H0
Global Const PISO813_BI_2           = &H1
Global Const PISO813_BI_4           = &H2
Global Const PISO813_BI_8           = &H3
Global Const PISO813_BI_16          = &H4

Global Const PISO813_UNI_1          = &H0
Global Const PISO813_UNI_2          = &H1
Global Const PISO813_UNI_4          = &H2
Global Const PISO813_UNI_8          = &H3
Global Const PISO813_UNI_16         = &H4
'/*************    return code    *************/
Global Const PISO813_NoError                    = 0
Global Const PISO813_DriverOpenError            = 1
Global Const PISO813_DriverNoOpen               = 2
Global Const PISO813_GetDriverVersionError      = 3
Global Const PISO813_CallDriverError            = 4
Global Const PISO813_FindBoardError             = 5
Global Const PISO813_ExceedBoardNumber          = 6
Global Const PISO813_TimeOutError               = &HFFFF
Global Const PISO813_AdError2                   = -100#

' ID
Global Const PISO_813                           = &H800A00
```

```
' The Test functions
Declare Function PISO813_ShortSub Lib "PISO813.dll" (ByVal a As Integer,
      ByVal b As Integer) As Integer
Declare Function PISO813_FloatSub Lib "PISO813.dll" (ByVal a As Single,
      ByVal b As Single) As Single
Declare Function PISO813_GetDllVersion Lib "PISO813.dll" () As Integer


' The Driver functions
Declare Function PISO813_DriverInit Lib "PISO813.dll" () As Integer
Declare Sub PISO813_DriverClose Lib "PISO813.dll" ()
Declare Function PISO813_SearchCard Lib "PISO813.dll" ( _
      wBoards As Integer, ByVal dwPIOCardID As Long) As Integer
Declare Function PISO813_GetDriverVersion Lib "PISO813.dll"  _
      (wDriverVersion As Integer) As Integer
Declare Function PISO813_GetConfigAddressSpace Lib "PISO813.dll" ( _
      ByVal wBoardNo As Integer, wAddrBase As Long, wIrqNo As Integer, _
      wSubVendor As Integer, wSubDevice As Integer, wSubAux As Integer, _
      wSlotBus As Integer, wSlotDevice As Integer) As Integer

Declare Function PISO813_ActiveBoard Lib "PISO813.dll" ( _
      ByVal wBoardNo As Integer) As Integer
Declare Function PISO813_WhichBoardActive Lib "PISO813.dll" () As Integer


' DIO functions
Declare Sub PISO813_OutputByte Lib "PISO813.dll" ( _
      ByVal address As Long, ByVal dataout As Integer)
Declare Sub PISO813_OutputWord Lib "PISO813.dll" ( _
      ByVal address As Long, ByVal dataout As Long)
Declare Function PISO813_InputByte Lib "PISO813.dll" ( _
      ByVal address As Long) As Integer
Declare Function PISO813_InputWord Lib "PISO813.dll" ( _
      ByVal address As Long) As Long

' AD functions
Declare Function PISO813_SetChGain Lib "PISO813.dll" ( _
```

```
    ByVal wAddrBase As Long, ByVal wChannel As Integer, _
    ByVal wGainCode As Integer) As Integer
Declare Function PISO813_AD_Hex Lib "PISO813.dll" ( _
    ByVal wAddrBase As Long) As Integer
Declare Function PISO813_ADs_Hex Lib "PISO813.dll" ( _
    ByVal wAddrBase As Long, wBuffer As Integer, _
    ByVal dwDataNo As Long) As Integer
Declare Function PISO813_AD_Float Lib "PISO813.dll" ( _
    ByVal wAddrBase As Long, ByVal wJump10v As Integer, _
    ByVal wBipolar As Integer) As Single
Declare Function PISO813_ADs_Float Lib "PISO813.dll" ( _
    ByVal wAddrBase As Long, ByVal wJump10v As Integer, _
    ByVal wBipolar As Integer, fBuffer As Single, _
    ByVal dwDataNo As Long) As Single
Declare Function PISO813_AD2F Lib "PISO813.dll" (ByVal whex as integer, _
    ByVal wGainCode as integer, ByVal wJump20v as integer, _
    ByVal wBipolar as integer) as Single
```

2.3 PISO813.PAS

```
unit  PISO813;        { PISO813.dll interface unit }

interface

type PSingle=^Single;
type PWord=^Word;
type DWORD=Cardinal;

const

//************ define PISO813 relative address **********************/
  PISO813_AD_LO          =$d0;    // Analog to Digital, Low Byte
  PISO813_AD_HI          =$d4;    // Analog to Digital, High Byte
  PISO813_SET_CH         =$E0;    // channel selecting
  PISO813_SET_GAIN       =$E4;    // PGA gain code
  PISO813_SOFT_TRIG      =$F0;    // A/D trigger control register

//****** define the gain mode ********/
  PISO813_BI_1           =$00;
  PISO813_BI_2           =$01;
  PISO813_BI_4           =$02;
  PISO813_BI_8           =$03;
  PISO813_BI_16          =$04;

  PISO813_UNI_1          =$00;
  PISO813_UNI_2          =$01;
  PISO813_UNI_4          =$02;
  PISO813_UNI_8          =$03;
  PISO813_UNI_16         =$04;

//*************    return code    *************/
  PISO813_NoError                =0;
  PISO813_DriverOpenError        =1;
  PISO813_DriverNoOpen           =2;
```

```
  PISO813_GetDriverVersionError   =3;
  PISO813_CallDriverError         =4;
  PISO813_FindBoardError          =5;
  PISO813_ExceedBoardNumber       =6;
  PISO813_TimeOutError            =$ffff;
  PISO813_AdError2                =-100.0;


  // ID
  PISO_813                        = $800A00;



// Test functions
function  PISO813_ShortSub(nA : smallint; nB : smallint)   : smallint; StdCall;
function  PISO813_FloatSub(fA :   single; fB : single)     : single; StdCall;
function  PISO813_GetDllVersion                            :  word; StdCall;

// Driver functions
function  PISO813_DriverInit                               : word; StdCall;
procedure PISO813_DriverClose                                    ; StdCall;
function  PISO813_SearchCard(var wBoards:WORD;
      dwPIOCardID:LongInt):WORD; StdCall;
function  PISO813_GetDriverVersion(var wDriverVer: word)  : word; StdCall;
function  PISO813_GetConfigAddressSpace(wBoardNo:word;
      var wAddrBase:LongInt;  var wIrqNo:word; var wSubVerdor:word;
      var wSubDevice:word;    var wSubAux:word; var wSlotBus:word;
      var wSlotDevice:word ): word; StdCall;

// DIO functions
procedure PISO813_OutputByte(wPortAddr : LongInt; bOutputVal : Word)
      ; StdCall;
procedure PISO813_OutputWord(wPortAddr : LongInt; wOutputVal : LongInt)
      ; StdCall;
function  PISO813_InputByte(wPortAddr  : LongInt )         : word; StdCall;
function  PISO813_InputWord(wPortAddr  : LongInt )         : LongInt; StdCall;

// AD functions
```

```
function  PISO813_SetChGain(wAddrBase : LongInt; wChannel:Integer;
      wGainCode:Integer):Word; StdCall;
function  PISO813_AD_Hex(   wAddrBase : LongInt):Word; StdCall;
function  PISO813_ADs_Hex(  wAddrBase : LongInt; wBuffer:PWord;
      dwDataNo:LongInt):Word; StdCall;
function  PISO813_AD_Float( wAddrBase : LongInt; wJump20v:Integer;
      wBipolar:Integer):Single; StdCall;
function  PISO813_ADs_Float(wAddrBase : LongInt; wJump20v:Integer;
      wBipolar:Integer; fBuffer:PSingle; dwDataNo:LongInt):Single; StdCall;
function  PISO813_AD2F(whex:Word; wGainCode:Word; wJump20v:Word;
      wBipolar:Word):Single; StdCall;

implementation

// Test functions
function  PISO813_ShortSub;                external 'PISO813.DLL' name
      'PISO813_ShortSub';
function  PISO813_FloatSub;                external 'PISO813.DLL' name
      'PISO813_FloatSub';
function  PISO813_GetDllVersion;           external 'PISO813.DLL' name
      'PISO813_GetDllVersion';

// Driver functions
function  PISO813_DriverInit;              external 'PISO813.DLL' name
      'PISO813_DriverInit';
procedure PISO813_DriverClose;             external 'PISO813.DLL' name
      'PISO813_DriverClose';
function  PISO813_SearchCard;              external 'PISO813.DLL' name
      'PISO813_SearchCard';
function  PISO813_GetDriverVersion;        external 'PISO813.DLL' name
      'PISO813_GetDriverVersion';
function  PISO813_GetConfigAddressSpace;external 'PISO813.DLL' name
      'PISO813_GetConfigAddressSpace';

//function  PISO813_ActiveBoard;           external 'PISO813.DLL' name
      'PISO813_ActiveBoard';
//function  PISO813_WhichBoardActive;      external 'PISO813.DLL' name
      'PISO813_WhichBoardActive';
```

```
// DIO functions
procedure PISO813_OutputByte;          external 'PISO813.DLL' name
      'PISO813_OutputByte';
procedure PISO813_OutputWord;          external 'PISO813.DLL' name
      'PISO813_OutputWord';
function  PISO813_InputByte;           external 'PISO813.DLL' name
      'PISO813_InputByte';
function  PISO813_InputWord;           external 'PISO813.DLL' name
      'PISO813_InputWord';

// AD functions
function  PISO813_SetChGain;           external 'PISO813.DLL' name
      'PISO813_SetChGain';
function  PISO813_AD_Hex;              external 'PISO813.DLL' name
      'PISO813_AD_Hex';
function  PISO813_ADs_Hex;             external 'PISO813.DLL' name
      'PISO813_ADs_Hex';
function  PISO813_AD_Float;            external 'PISO813.DLL' name
      'PISO813_AD_Float';
function  PISO813_ADs_Float;           external 'PISO813.DLL' name
      'PISO813_ADs_Float';
function  PISO813_AD2F;                external 'PISO813.DLL' name
      'PISO813_AD2F';


end.
```

3. Function Descriptions

In this chapter, we use some keywords to indicate the attribute of Parameters.

Keyword	Setting parameter by user before calling this function ?	Get the data/value from this parameter after calling this function ?
[Input]	Yes	No
[Output]	No	Yes
[Input, Output]	Yes	Yes

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

3.1 FUNCTIONS OF TEST

3.1.1 PISO813_GetDllVersion

- **Description:**

  To get the version number of PISO813.DLL

- **Syntax:**

  WORD PISO813_GetDllVersion(void)

- **Parameter:**

  None

- **Return:**

  200(hex) for version 2.00

3.1.2 PISO813_ShortSub

- **Description:**

  To perform the subtraction as nA - nB in short data type. This function is provided for testing DLL linkage purpose.

- **Syntax:**

  short PISO813_ShortSub(short nA, short nB)

- **Parameter:**

  nA :[Input] 2 bytes short data type value
  nB :[Input] 2 bytes short data type value

- **Return:**

  The value of nA - nB

3.1.3 PISO813_FloatSub

- **Description:**

  To perform the subtraction as fA - fB in float data type. This function is provided for testing DLL linkage purpose.

- **Syntax:**

  float PISO813_FloatSub(float fA, float fB)

- **Parameter:**

  fA : [Input] 4 bytes floating point value
  fB : [Input] 4 bytes floating point value

- **Return:**

  The value of fA - fB

3.2 FUNCTIONS OF I/O

3.2.1 PISO813_OutputByte

- **Description :**

  This subroutine will send the 8 bits data to the desired I/O port.

- **Syntax :**

  void PISO813_OutputByte(DWORD wPortAddr, WORD bOutputVal);

- **Parameter :**

  wPortAddr : [Input] I/O port addresses, please refer to function PISO813_GetConfigAddressSpace.
  Only the low WORD is valid.

  bOutputVal : [Input] 8 bit data send to I/O port.
  Only the low BYTE is valid.

- **Return:**

  None

3.2.2 PISO813_InputByte

- **Description :**

  This subroutine will input the 8 bit data from the desired I/O port.

- **Syntax :**

  WORD PISO813_InputByte(DWORD wPortAddr);

- **Parameter :**

  wPortAddr : [Input] I/O port addresses, please refer to function PISO813_GetConfigAddressSpace().
  Only the low WORD is valid.

- **Return:**

  16 bits data with the leading 8 bits are all 0.
  (Only the low BYTE is valid.)

3.2.3 PISO813_OutputWord

- **Description :**

  This subroutine will send the 16 bits data to the desired I/O port.

- **Syntax :**

  void PISO813_OutputWord(DWORD wPortAddr, DWORD wOutputVal);

- **Parameter :**

  wPortAddr : [Input] I/O port addresses, please refer to function PISO813_GetConfigAddressSpace(). Only the low WORD is valid.

  wOutputVal : [Input] 16 bit data send to I/O port. Only the low WORD is valid.

- **Return:**

  None

3.2.4 PISO813_InputWord

- **Description :**

  This subroutine will input the 16 bit data from the desired I/O port.

- **Syntax :**

  DWORD PISO813_InputWord(DWORD wPortAddr);

- **Parameter :**

  wPortAddr : [Input] I/O port addresses, please refer to function PISO813_GetConfigAddressSpace(). Only the low WORD is valid.

- **Return:**

  16 bit data. Only the low WORD is valid.

3.3 FUNCTIONS OF DRIVER

3.3.1 PISO813_GetDriverVersion

- **Description :**

  This subroutine will read the version number of PISO-813 driver.

- **Syntax :**

  WORD PISO813_GetDriverVersion(WORD *wDriverVersion);

- **Parameter :**

  wDriverVersion : [Output] address of wDriverVersion

- **Return:**

  PISO813_NoError : OK
  PISO813_DriverNoOpen : The PISO-813 driver no open
  PISO813_GetDriverVersionError : Read driver version error

3.3.2 PISO813_DriverInit

- **Description :**

  This subroutine will open the PISO-813 driver and allocate the resource for the device. This function must be called once before calling other PISO-813 functions.

- **Syntax :**

  WORD PISO813_DriverInit();

- **Parameter :**

  None

- **Return:**

  PISO813_NoError : OK
  PISO813_DriverOpenError : open PISO-813 Driver error

3.3.3 PISO813_DriverClose

- **Description :**

  This subroutine will close the PISO-813 Driver and release the resource from the device. This function must be called once before exit the user's application.

- **Syntax :** void PISO813_DriverClose();

- **Parameter :** None

- **Return:** None

3.3.4 PISO813_GetConfigAddressSpace

- **Description :**

  Get the I/O address of PISO-813 board n.

- **Syntax :**

```
WORD PISO813_GetConfigAddressSpace
        ( WORD wBoardNo,   DWORD *wAddrBase,     WORD *wIrqNo,
        WORD *wSubVendor, WORD *wSubDevice,     WORD *wSubAux,
        WORD *wSlotBus,    WORD *wSlotDevice);
```

- **Parameter :**

wBoardNo	: [Input] PISO-813 board number
wAddrBase	: [Output] The base address of PISO-813 board. Only the low WORD is valid.
wIrqNo	: [Output] The IRQ number that the PISO-813 board using.
wSubVendor	: [Output] Sub Vendor ID.
wSubDevice	: [Output] Sub Device ID.
wSubAux	: [Output] Sub Aux ID.
wSlotBus	: [Output] Slot Bus number.
wSlotDevice	: [Output] Slot Device ID.

- **Return:**

PISO813_NoError	: OK
PISO813_FindBoardError	: handshake check error
PISO813_ExceedBoardError	: wBoardNo is invalidated

3.4A/D Functions

3.4.1 PISO813_SetChGain

- **Description:**

  This subroutine wills setting the cannel number and Gain-Code (Refer to Section 1.2) for the AD converter.

- **Syntax:**

  WORD PISO813_SetChGain(DWORD wBase, WORD wChannel,
  WORD wGainCode);

- **Parameter:**

  wBase : [Input] I/O port base addresses, please refer the PISO813_GetConfigAddressSpace().
  wChannel : [Input] A/D channel number, 0 to 31.
  wGainCode : [Input] The value is 0 to 4, refer to Sec. 1.2.

- **Return:**

  PISO813_NoError : OK

3.4.2 PISO813_AD2F

- **Description:**

  This subroutine will convert the Hex value to floating value depending on GainCode , Bipolar/Unipolar and 10v/20v.

- **Syntax:**

  float PISO813_AD2F(WORD wHexValue, WORD wGainCode,
  WORD wJump20v , WORD wBipolar);

- **Parameter:**

  wHexValue : [Input] Hex Value 0 ~ 0x0FFF
  wGainCode : [Input] The value is 0 to 4.
  Refer to Sec. 1.2 for detail information.
  wJump20v : [Input] 1:20v(HW default) 0:10v
  wBipolar : [Input] 1:Bipolar(HW default) 0:Unipolar

- **Return:**

  PISO813_AdError2 : A/D converter error (return –100.0)
  Other value : The **floating-point value** of A/D conversion (-10 to 10)

3.4.3 PISO813_AD_Hex

- **Description:**

  This subroutine will perform an A/D conversion by polling. The A/D converter is 12 bits for PISO-813. Refer to PISO813_SetChGain().

- **Syntax:**

  WORD PISO813_AD_Hex(DWORD wBase);

- **Parameter:**

  wBase : [Input] I/O port base addresses, please refer to PISO813_GetConfigAddressSpace().

- **Return:**

  PISO813_TimeOutError : A/D converter error (return 0xffff)

  Other value : The **Hex value** of A/D conversion (0 ~ 0x0fff)

3.4.4 PISO813_AD_Float

- **Description:**

  This subroutine will perform an A/D conversion by polling. The A/D converter is 12 bits for PISO-813. This subroutine will compute the result according to the **configuration code** (Section 1.2). Refer to PISO813_SetChGain().

- **Syntax:**

  float PISO813_AD_Float(DWORD wBase, WORD wJump20v,
  WORD wBipolar);

- **Parameter:**

  wBase : [Input] I/O port base addresses, please refer to PISO813_GetConfigAddressSpace().

  wJump20v : [Input] 1:20v(HW default) 0:10v

  wBipolar : [Input] 1:Bipolar(HW default) 0:Unipolar

- **Return:**

  PISO813_AdError2 : A/D converter error (return –100.0)

  Other value : The **floating-point value** of A/D conversion (-10 to 10)

3.4.5 PISO813_ADs_Hex

- **Description:**

  This subroutine will perform a number of A/D conversions by polling. This subroutine is very similar to PISO813_AD_Hex except that this subroutine will perform wCount of conversions instead of just one conversion. After A/D conversing, the A/D data are stored in a buffer in Hex format. The **wBuf** is the starting address of this data buffer. Refer to PISO813_SetChGain().

- **Syntax:**

  WORD PISO813_ADs_Hex(DWORD wBase, WORD wBuf[],
  DWORD wCount);

- **Parameter:**

  wBase : [Input] I/O port base addresses, please refer to PISO813_GetConfigAddressSpace().

  wBuf : [Output] Starting address of the data buffer
  The user must allocate spaces for this buffer and send the address into the function. This function will fill the data into this buffer. The user cans analyze these data from the buffer after calling this function.

  wCount : [Input] Number of A/D conversions will be performed

- **Return:**

  PISO813_TimeOutError : A/D converter error (0xffff)

  PISO813_NoError : Operation is OK

3.4.6PISO813_ADs_Float

- **Description:**

  This subroutine will perform a number of A/D conversions by polling. This subroutine is very similar to PISO813_AD_Float except that this subroutine will perform wCount of conversions instead of just one conversion. Then the A/D data are stored in a data buffer in Float format. The **fBuf** is the starting address of this data buffer. Refer to PISO813_SetChGain().

- **Syntax:**

  WORD PISO813_ADs_Float(DWORD wBase, WORD wJump20v,
  WORD wBipolar, float fBuf[], DWORD wCount);

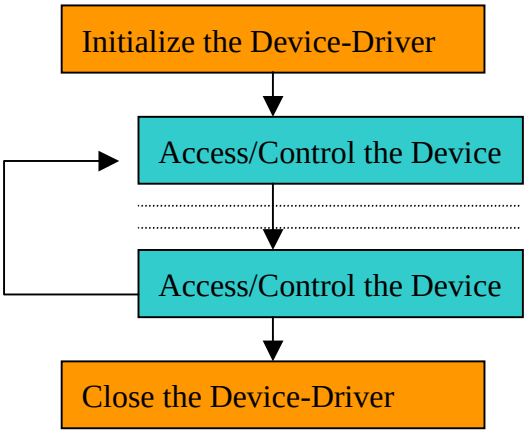
- **Parameter:**

  wBase : [Input] I/O port base addresses, refer to PISO813_GetConfigAddressSpace().
  wJump20v : [Input] 1:20v(HW default) 0:10v
  wBipolar : [Input] 1:Bipolar(HW default) 0:Unipolar
  fBuf : [Output] Starting address of the data buffer **(in float format)**
  The user must allocate spaces for this buffer and send the address into the function. This function will fill the data into this buffer. The user cans analyze these data from the buffer after calling this function.
  wCount : [Input] Number of A/D conversions will be performed

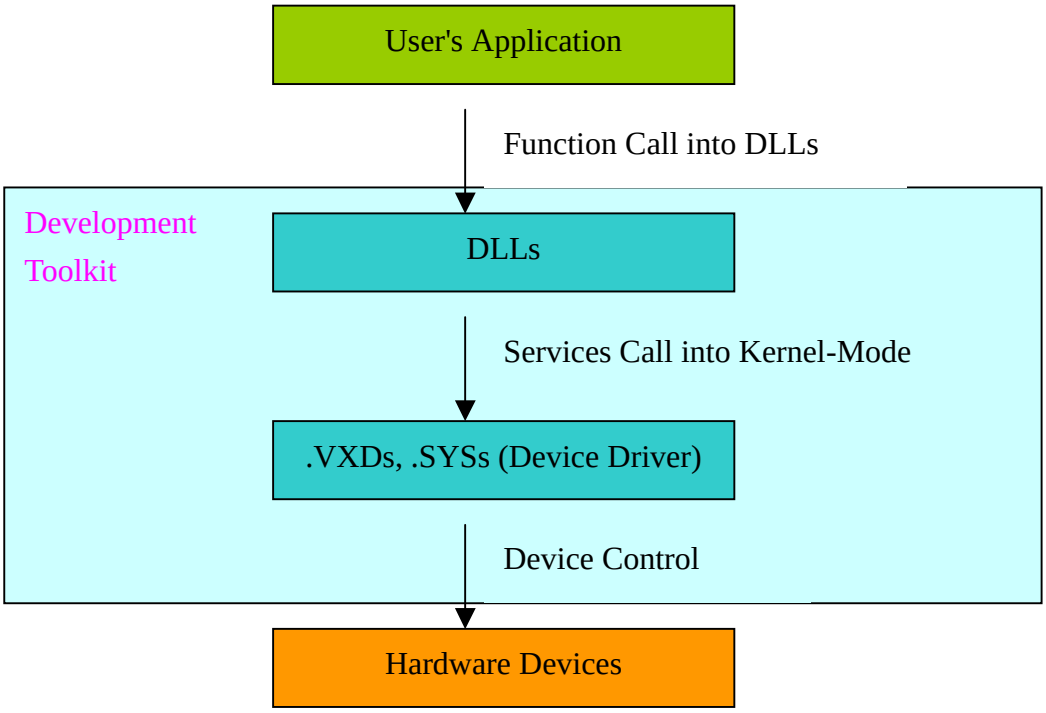
- **Return:**

  PISO813_TimeOutError : A/D converter error (0xffff)
  PISO813_NoError : Operation is OK

4. Program Architecture




```
PISO813_DriverInit()
  // Enable All DI/DO
….
     PISO813_InputByte( … )
     ……..
     ……..
     PISO813_OutputByte(…)
…..
```



5. Problems Report

Technical support is available at no charge as described below. The best way to report problems is send electronic mail to

icpdas@ms8.hinet.net
Service@icpdas.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 3.1, Windows for Workgroups, Windows NT 4.0, etc.
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 your comments? And please keeps contact with us.

ICP DAS
E-mail: icpdas@ms8.hinet.net
Service@icpdas.com
Web Site: http://www.icpdas.com
http://www.icpdas.com.tw