ICP DAS UniDAQ SDK User Manual

English Version

Warranty

All products manufactured by ICP DAS are under warranty regarding defective materials for a period of one year, starting from the date of delivery to the original purchaser.

Disclaimer

ICP DAS assumes no liability for damages resulting from 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. Changes to the specifications and features in this manual may be made by ICP DAS without prior notice. No part of this manual may be reproduced, copied, translated, transmitted, or published in any form or by any means without ICP DAS's prior written permission

Copyright

Trademark

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

TABLE OF CONTENTS

1 Introduction

ICP DAS PC-based I/O board Development Kits provide a UniDAQ SDK Dynamic Link Library together with a number of simple samples. Refering to these samples; users can learn how to implement I/O card applications quickly and easily, without the need to consider the programming steps required for the hardware register.

This document is an introduction to the DLL functions contained in the UniDAQ SDK. Each function is described in detail with examples of how it can be used to program an application, including several special founctions, and how it can be used to retrieve function argurments, return values and configuration code information.

The DLL includes the Driver Functions, Advance Driver Functions, Digital Input/Output Functions, Interrupt Functions, Analog Input Functions, Timer/Counter Functions and Memory Input/Output Functions.

Section 1.1 is a simple listing of the function categories. Section 1.2 contains the function return value code definition table. Section 1.3 contains the analog input and analog output configuration code table that be used to cans set the analog input and output range **(Note:** only analog input or output function board**)**. Section 1.4 contains board model serial number table. Chapter 2 is an introduction to the DLL functions introduction. After reading this chapter, the user will have a greates understanding of how to implement and operate these functions.

1.1 Function Definitions

The UniDAQ SDK provides a function library that divided into several categories “Driver functions”, “Advance driver functions”, “Digital I/O functions”, “Interrupt functions”, “Analog input functions”, “Analog output functions“, “Timer/Counter functions“ and “Memory I/O functions“.

- Driver Functions

Function Name	Parameters
Ixud_GetDllVersion	DWORD *wDLLVer
Ixud_DriverInit	WORD *wTotalBoards
Ixud_DriverClose	void
Ixud_SearchCard	WORD *wTotalBoards, DWORD dwModelNo
Ixud_GetBoardNoByCardID	WORD *wBoardNo, DWORD dwModelNumber, WORD wCardID

- Advance Driver Functions

Function Name	Parameters
Ixud_GetCardInfo	WORD wBoardNo, PIXUD_DEVICE_INFO sDevInfo,PIXUD_CARD_INFO sCardInfo, char * szModelName
Ixud_ReadPort	DWORD dwAddress, WORD wSize, DWORD* dwVal
Ixud_WritePort	DWORD dwAddress, WORD wSize, DWORD dwVal
Ixud_ReadPort32	DWORD dwAddress, DWORD* dwLow, DWORD* dwHigh
Ixud_WritePort32	DWORD dwAddress, DWORD dwLow, DWORD dwHigh

- Digital I/O Functions

Function Name	Parameters
Ixud_SetDIOModes32	WORD wBoardNo, DWORD dwDioMode
Ixud_SetDIOMode	WORD wBoardNo, WORD wPortNo, WORD wDioMode
Ixud_ReadDI	WORD wBoardNo, WORD wPortN, DWORD *dwDIVal
Ixud_WriteDO	WORD wBoardNo, WORD wPortNo, DWORD dwDOVal
Ixud_ReadDI32	WORD wBoardNo, WORD wPortNo, DWORD* dwLow, DWORD* dwHigh
Ixud_WriteDO32	WORD wBoardNo, WORD wPortNo, DWORD dwLow, DWORD dwHigh
Ixud_SoftwareReadbackDO	WORD wBoardNo, WORD wPortNo, DWORD *dwDOVal

- Interrupt Functions

Function Name	Parameters
Ixud_SetEventCallback	WORD wBoardNo, WORD wEventType, WORD wInterruptSource, HANDLE *hEvent, PVOID CallbackFun, DWORD dwCallBackParameter
Ixud_RemoveEventCallback	WORD wBoardNo, WORD wInterruptSource
Ixud_InstallIrq	WORD wBoardNo, DWORD dwInterruptMask
Ixud_RemoveIrq	WORD wBoardNo

■ Analog Input Functions

Function Name	Parameters
Ixud_ConfigAI	WORD wBoardNo, WORD wFIFOSizeKB, DWORD BufferSizeCount, WORD wCardType, WORD wDelaySettlingTime
Ixud_ClearAIBuffer	WORD wBoardNo
Ixud_GetBufferStatus	WORD wBoardNo, WORD *wBufferStatus, DWORD *dwDataCount
Ixud_ReadAI	WORD wBoardNo, WORD wChannel, WORD wConfig, float *fValue
Ixud_ReadAIH	WORD wBoardNo, WORD wChannel, WORD wConfig, DWORD *dwValue
Ixud_PollingAI	WORD wBoardNo, WORD wChannel, WORD wConfig, DWORD dwDataCount, float fValue[]
Ixud_PollingAIH	WORD wBoardNo, WORD wChannel, WORD wConfig, DWORD dwDataCount, DWORD dwValue[]
Ixud_PollingAIScan	WORD wBoardNo, WORD wChannels, WORD wChannelList[], WORD wConfigList[], DWORD dwDataCountPerChannel, float fValue[]
Ixud_PollingAIScanH	WORD wBoardNo, WORD wChannels, WORD wChannelList[], WORD wConfigList[], DWORD dwDataCountPerChannel, DWORD dwValue[]
Ixud_StartAI	WORD wBoardNo, WORD wChannel, WORD wConfig, float fSamplingRate, DWORD dwDataCount
Ixud_StartAIScan	WORD wBoardNo, WORD wChannels, WORD wChannelList[], WORD wConfigList[], float fSamplingRate, DWORD dwDataCountPerChannel
Ixud_GetAIBuffer	WORD wBoardNo, DWORD dwDataCount, float fValue[]
Ixud_GetAIBufferH	WORD wBoardNo, DWORD dwDataCount, DWORD hValue[]
Ixud_StopAI	WORD wBoardNo

■ Analog Output Functions

Function Name	Parameters
Ixud_ConfigAO	WORD wBoardNo, WORD wChannel, WORD wCfgCode
Ixud_WriteAOVoltage	WORD wBoardNo, WORD wChannel, float fValue
Ixud_WriteAOVoltageH	WORD wBoardNo, WORD wChannel, DWORD hValue
Ixud_WriteAOCurrent	WORD wBoardNo, WORD wChannel, float fValue
Ixud_WriteAOCurrentH	WORD wBoardNo, WORD wChannel, DWORD hValue

■ Timer/Counter Functions

Function Name	Parameters
Ixud_ReadCounter	WORD wBoardNo, WORD wChannel, DWORD *dwValue
Ixud_SetCounter	WORD wBoardNo, WORD wChannel, WORD wMode, DWORD dwValue
Ixud_DisableCounter	WORD wBoardNo, WORD wChannel
Ixud_SetFCChannelMode	WORD wBoardNo, WORD wChannel, WORD wMode, WORD wDelayMs
Ixud_ReadFrequency	WORD wBoardNo, WORD wChannel, float *fFrequency, DWORD dwTimeOutMs, WORD *wStatus

■ Memory I/O Functions

Function Name	Parameters
Ixud_ReadMemory	WORD wBoardNo, DWORD dwOffset, WORD Size, DWORD *dwValue
Ixud_WriteMemory	WORD wBoardNo, DWORD dwOffset, WORD Size, DWORD *dwValue
Ixud_ReadMemory32	WORD wBoardNo, DWORD dwOffset, DWORD *dwLow, DWORD *dwHigh
Ixud_WriteMemory32	WORD wBoardNo, DWORD dwOffset, DWORD dwLow, DWORD dwHigh

1.2 Function Return Value Definitions

When calling a function from the UniDAQ SDK that will return a value, this value indicates the operating status of the function. Refer to the following table for the definition of each return value.

Return Value	Definition	Description
0	Ixud_NoErr	Correct
1	Ixud_OpenDriverErr	Open driver error
2	Ixud_PnPDriverErr	Plug & Play error
3	Ixud_DriverNoOpen	The driver was not open
4	Ixud_GetDriverVersionErr	Recieve driver version error
5	Ixud_ExceedBoardNumber	Board number error
6	Ixud_FindBoardErr	No board found
7	Ixud_BoardMappingErr	Board Mapping error
8	Ixud_DIOModesErr	Digital input/output mode setting error
9	Ixud_InvalidAddress	Invalid address
10	Ixud_InvalidSize	Invalid size
11	Ixud_InvalidPortNumber	Invalid port number
12	Ixud_UnSupportedModel	This board model is not supported
13	Ixud_UnSupportedFun	This function is not supported
14	Ixud_InvalidChannelNumber	Invalid channel number
15	Ixud_InvalidValue	Invalid value
16	Ixud_InvalidMode	Invalid mode
17	Ixud_GetAIStatusTimeOut	A timeout occurred while receiving the status of the analog input
18	Ixud_TimeOutErr	Timeout error.
19	Ixud_CfgCodeIndexErr	A compatible configuration code table index could not be found
20	Ixud_ADCCTLTimeoutErr	ADC controller a timeout error
21	Ixud_FindPCIIndexErr	A compatible PCI table index value could not be found
22	Ixud_InvalidSetting	Invalid setting value.
23	Ixud_AllocateMemErr	Error while allocating the memory space
24	Ixud_InstallEventErr	Error while installing the interrupt event
25	Ixud_InstallIrqErr	Error while installing the interrupt IRQ
26	Ixud_RemoveIrqErr	Error while removing the interrupt IRQ
27	Ixud_ClearIntCountErr	Error while the clear interrupt count
28	Ixud_GetSysBufferErr	Error while retrieving the system buffer
29	Ixud_CreateEventErr	Error while create the event
30	Ixud_UnSupportedResolution	Resolution not supported
31	Ixud_CreateThreadErr	Error while create the thread
32	Ixud_ThreadTimeOutErr	Thread timeout error
33	Ixud_FIFOOverFlowErr	FIFO overflow error
34	Ixud_FIFOTimeOutErr	FIFO timeout error
35	Ixud_GetIntInstStatus	Retrieves the status of the interrupt installation.
36	Ixud_GetBufStatus	Retrieves the status of the system buffer
37	Ixud_SetBufCountErr	Error while setting the buffer count
38	Ixud_SetBufInfoErr	Error while setting the buffer data
39	Ixud_FindCardIDErr	Card ID code could not be found
40	Ixud_EventThreadErr	Event Thread error
41	Ixud_AutoCreateEventErr	Error while automatically creating an event
42	Ixud_RegThreadErr	Register Thread error
43	Ixud_SearchEventErr	Search Event error
44	Ixud_FifoResetErr	Error while resetting the FIFO
45	Ixud_InvalidBlock	Invalid EEPROM block
46	Ixud_InvalidAddr	Invalid EEPROM address
47	Ixud_AcqireSpinLock	Error while acquiring spin lock
48	Ixud_ReleaseSpinLock	Error while releasing spin lock
49	Ixud_SetControlErr	Analog input setting error
50	Ixud_InvalidChannels	Invalid channel number

Return Value	Definition	Description
51	Ixud_SearchCardErr	Invalid model number
52	Ixud_SetMapAddressErr	Error while setting the mapping address
53	Ixud_ReleaseMapAddressErr	Error while releasing the mapping address
54	Ixud_InvalidOffset	Invalid memory offset
55	Ixud_ShareHandleErr	Open the share memory fail
56	Ixud_InvalidDataCount	Invalid data count
57	Ixud_WriteEEPErr	Error while writing the EEPROM

1.3 Configuration Code Define

The following configuration code values can be used to set the analog input/output range.

1.3.1 Analog Input Configuration Codes

Configuration Code	Definition	Range
0	IXUD_BI_10V	Bipolar +/- 10V
1	IXUD_BI_5V	Bipolar +/- 5V
2	IXUD_BI_2V5	Bipolar +/- 2.5V
3	IXUD_BI_1V25	Bipolar +/- 1.25V
4	IXUD_BI_0V625	Bipolar +/- 0.625V
5	IXUD_BI_0V3125	Bipolar +/- 0.3125V
6	IXUD_BI_0V5	Bipolar +/- 0.5V
7	IXUD_BI_0V05	Bipolar +/- 0.05V
8	IXUD_BI_0V005	Bipolar +/- 0.005
9	IXUD_BI_1V	Bipolar +/- 1V
10	IXUD_BI_0V1	Bipolar +/- 0.1V
11	IXUD_BI_0V01	Bipolar +/- 0.01V
12	IXUD_BI_0V001	Bipolar +/- 0.001V
13	IXUD_UNI_20V	Unipolar 0 ~ 20V
14	IXUD_UNI_10V	Unipolar 0 ~ 10V
15	IXUD_UNI_5V	Unipolar 0 ~ 5V
16	IXUD_UNI_2V5	Unipolar 0 ~ 2.5V
17	IXUD_UNI_1V25	Unipolar 0 ~ 1.25V
18	IXUD_UNI_0V625	Unipolar 0 ~ 0.625V
19	IXUD_UNI_1V	Unipolar 0 ~ 1V
20	IXUD_UNI_0V1	Unipolar 0 ~ 0.1V
21	IXUD_UNI_0V01	Unipolar 0 ~ 0.01V
22	IXUD_UNI_0V001	Unipolar 0 ~ 0.001V

1.3.2 Analog Output Configuration Code

Configuration Code	Definition	Range
0	IXUD_AO_UNI_5V	0 ~ 5V
1	IXUD_AO_BI_5V	+/- 5V
2	IXUD_AO_UNI_10V	0 ~ 10V
3	IXUD_AO_BI_10V	+/- 10V
4	IXUD_AO_UNI_20V	0 ~ 20V
5	IXUD_AO_BI_20V	+/- 20V

1.4 Board Model Numbers

Definition	Model No. (HEX)	Supported model numbers
PIOD56	800140	PIO-D24/D56/D24U/D56U
PIOD48	800130	PIO-D48/D48U
PIOD64	800120	PIO-D64/D64U
PIOD96	800110	PIO-D96
PIOD144	800100	PIO-D144
PIOD168	800150	PIO-D168
PIODA	800400	PIO-DA4/DA8/DA16/DA4U/DA8U/DA16U/PISO-DA4U/DA8U/DA16U
PIO821	800310	PIO-821 L/H
PIO827	FF0000	PIO-827 LU
PISOP16R16	1800FF	PISO-P16R16
PISOC64	800800	PISO-C64
PISOP64	800810	PISO-P64
PISOA64	800850	PISO-A64
PISOP32C32	800820	PISO-P32C32/P32C32U/P32S32WU
PISOP32A32	800870	PISO-P32A32
PISOP8R8	800830	PISO-P8R8/PISO-P8R8AC/PISO-P8R8DC, PISO-P16R16 (U/E)
PISO730	800840	PISO-730
PISO730A	800880	PISO-730A
PISO725	8008FF	PISO-725
PISODA2	800B00	PISO-DA2
PISO813	800A00	PISO-813
PCITMC12	DF2962	PCI-TMC12/PCI-TMC12A
PCIM512	DE9562	PCI-M512
PCIM256	DE92A6	PCI-M256
PCIM128	DE9178	PCI-M128
PCIFC16	B13017	PCI-FC16U
PCID64	DE3513	PCI-D64
PCI822	DE3823	PCI-822 LU
PCI826	DE3827	PCI-826 LU
PCI827	DE3828	PCI-827 LU
PCI100X	341002	PCI-1002 LU/HU
PCI1202	345672	PCI-1202 L/H , PCI-1202U L/H
PCI1602	345676	PCI-1602/1602U, PCI-1602F
PCI180X	345678	PCI-1800 L/H, PCI-1802 L/H
PCIP8R8	D6102B	PCI-P8R8
PCIP16R16	D61E39	PCI-P16R16/P16C16/P16POR16

2 Introduction to DLL Functions

The use of before attention below to keyword could convenient you reading.

Keyword	Set parameter by user before calling this function?	Get parameter by user after calling this function?
[Input]	Yes	No
[Output]	No	Yes

2.1 Driver Functions

2.1.1 Ixud_GetDllVersion Function

This function retrieves the version number of the UniDAQ SDK library.

- **Syntax**

```
WORD Ixud_GetDllVersion(
        DWORD *dwDLLVer
);
```

- **Parameters**

**dwDLLVer*
[Output] retrieves the version number of the UniDAQ SDK library.

- **Return value**

Refer to section 1.2.

2.1.2 Ixud_DriverInit Function

This function is used to request the system to allocate resources.This function will then for all boards supported by UniDAQ and initialize each board. Finally, it will retrieve the total number of boards. It is necessity on program start point, and must be used before calling any other function.

- **Syntax**

```
WORD Ixud_DriverInit(
        WORD *wTotalBoards
);
```

- **Parameters**

**wTotalBoards*
[Output] Retrieves the total number of UniDAQ boards in PC.

- **Return value**

Refer to section 1.2.

2.1.3 Ixud_DriverClose Function

Calling this function will release resources to the system. It is necessity on program end point, and must be used after calling all other function.

- **Syntax**

```
WORD Ixud_DriverClose(
        void
);
```

- **Parameters**

None

- **Return value**

Refer to section 1.2.

2.1.4 Ixud_GetBoardNoByCardID Function

This function provide to user who get to the board number from the module sequence number or ID number parameters.

- **Syntax**

```
WORD Ixud_GetBoardNoByCardID(
        WORD *wBoardNo,
        DWORD dwModelNumber,
        WORD wCardID
);
```

- **Parameters**

**wBoardNo*
[Output] Retrieves the board number.

dwModelNumber
[Input] User setting the board module sequence number that could refer to section 1.4 .

wCardID
[Input] User setting the board ID number.

- **Return value**

Refer to section 1.2.

2.2 Advanced Driver Functions

2.2.1 Ixud_GetCardInfo Function

This function retrieves the hardware and software information and the model name of the board.

- **Syntax**

```
WORD Ixud_GetCardInfo(
        WORD wBoardNo,
        PIXUD_DEVICE_INFO sDevInfo,
        PIXUD_CARD_INFO sCardInfo,
        char *szModelName
);
```

- **Parameters**

wBoardNo

[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

sDevInfo

[Output] Get to board information from System. Variable type is PIXUD_DEVICE_INFO structure. Structure define as follow：

```
typedef struct _IXUD_DEVICE_INFO_
{
        DWORD dwSize;                   // Structure size
        WORD wVendorID;                 //Vendor ID
        WORD wDeviceID;                 //Device ID
        WORD wSubVendorID;              //Sub Vendor ID
        WORD wSubDeviceID;              //Sub Device ID
        DWORD dwBAR[6];                 //PCI Bar 0 ~ 5
        UCHAR BusNo;                    //PCI Bus No.
        UCHAR DevNo;                    //Slot No. in PCI bus
        UCHAR IRQ;                      //IRQ No.
        UCHAR Aux;                      //Aux No.
        DWORD dwReserved1[6];           // Hold
}IXUD_DEVICE_INFO,*PIXUD_DEVICE_INFO;
```

sCardInfo

[Output] Get the board hardware information. Variable type is PIXUD_CARD_INFO structure. Structure define as follow：

```
typedef struct _IXUD_CARD_INFO_
{
        DWORD dwSize;                   // Structure size
        DWORD dwModelNo;                //Model Number, refer to section 1.4
        UCHAR CardID;                   //Card ID.(CardID = 0xFF,unsupport)
        UCHAR wSingleEnded;             //1:S.E Analog Input  2:D.I.F, 0xFF:non
        WORD wReserved;                 //Hold
        WORD wAIChannels;               //AI channels
        WORD wAOChannels;               //AO channels
        WORD wDIPorts;                  // Singlex DI port
        WORD wDOPorts;                  // Singlex DO port
        WORD wDIOPorts;                 // Duplexing DI/DO port
        WORD wDIOPortWidth;             //DI/DO bandwidty (8/16/32bit)
        WORD wCounterChannels;          //Time/Counter channels
        WORD wMemorySize;               //Board memory size (KByte)
        DWORD dwReserved1[6];           //Hold
}IXUD_CARD_INFO,*PIXUD_CARD_INFO;
```

**szModelName*
[Output] Get the model name. It is a string that size are 20 char.

- **Return value**
  Refer to section 1.2.

2.2.2 lxud_ReadPort Function

Read the 8/16/32bit value from I/O port.

- **Syntax**

```
WORD lxud_ReadPort(
        DWORD dwAddress,
        WORD wSize,
        DWORD* dwVal
);
```

- **Parameters**
  dwAddress
  [Input] Setting I/O port address.

  wSize
  [Input] Setting read value length.

wSize	Length(bit)
8	8(Byte)
16	16(WORD)
32	32(DWORD)

2-1 table is wSize argument setting.

  **dwVal*
  [Output] Get to I/O port value.

- **Return value**
  Refer to section 1.2.

2.2.3 Ixud_WritePort Function

From I/O port write the 8/16/32bit value.

- **Syntax**

```
WORD Ixud_WritePort(
        DWORD dwAddress,
        WORD wSize,
        DWORD  dwVal
);
```

- **Parameters**

dwAddress
[Input] Set I/O port address.

wSize
[Input] Set write value length.

wSize	Length(bit)
8	8(Byte)
16	16(WORD)
32	32(DWORD)

2-2 table is wSize argument setting.

dwVal
[Input] Write the I/O port value.

- **Return value**

Refer to section 1.2.

2.2.4 Ixud_ReadPort32 Function

- **Syntax**
- **Parameters**

- **Return value**

Refer to section 1.2.

2.2.5 Ixud_WritePort32 Function

- **Syntax**
- **Parameters**

- **Return value**

Refer to section 1.2.

2.3 Digital Input/ Output Function Member

2.3.1 Ixud_SetDIOModes32 Function

Set I/O mode of multi-port at the same time. It only supports the I/O function port.

- **Syntax**

```
WORD Ixud_SetDIOModes32(
        WORD wBoardNo,
        DWORD dwDioMode
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

dwDioMode
[Input] Set digital I/O port for input or output mode, the every bit collate a digital I/O port.The maximum could set 32 digital I/O ports.

dwDioMode	Mode
0	Input mode
1	Output mode

2-3 table is dwDioMode argument setting.

- **Return value**

Refer to section 1.2.

2.3.2 Ixud_SetDIOMode Function

User select port number and then set this I/O mode of port. It only supports the I/O function port.

- **Syntax**

```
WORD Ixud_SetDIOMode(
        WORD wBoardNo,
        WORD wPortNo,
        WORD wDioMode
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wPortNo
[Input] User assigns the port number.

wDioMode
[Input] Set digital I/O port for input or output mode.

dwDioMode	Mode
0	Input mode
1	Output mode

2-4 table is dwDioMode argument setting.

- **Return value**

Refer to section 1.2.

2.3.3 Ixud_ReadDI Function

Read the user assignation of input port data.

- **Syntax**

```
WORD Ixud_ReadDI(
        WORD wBoardNo,
        WORD wPortNo,
        DWORD *dwDIVal
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wPortNo
[Input] User assign of port number.

**dwDIVal*
[Output] This parameter will save read data from input port.

- **Return value**

Refer to section 1.2.

2.3.4 Ixud_WriteDO Function

Write data to user assignation of output port.

- **Syntax**

```
WORD Ixud_WriteDO(
        WORD wBoardNo,
        WORD wPortNo,
        DWORD dwDOVal
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wPortNo
[Input] User assigns the port number.

dwDOVal
[Input] User set data to write output port.

- **Return value**

Refer to section 1.2.

2.3.5 Ixud_ReadDI32 Function

- **Syntax**
  WORD

- **Parameters**
  wBoardNo
  [Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

- **Return value**
  Refer to section 1.2.

2.3.6 Ixud_WriteDO32 Function

- **Syntax**
  WORD

- **Parameters**
  wBoardNo
  [Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

- **Return value**
  Refer to section 1.2.

2.3.7 Ixud_SoftwareReadbackDO Function

The UniDaq SDK can record the DO operation in DLL, and users can then read back the status from DLL later (not from hardware).
Note: The Ixud_ReadbackDO and Ixud_ReadbackDO32 function reads back DO status from hardware, while the Ixud_SoftwareReadbackDO function reads back DO status from DLL directly.

- **Syntax**

```
WORD Ixud_SofwareReadbackDO(
        WORD wBoardNo,
        WORD wPortNo,
        DWORD *dwDOVal
);
```

- **Parameters**
  wBoardNo
  [Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

  wPortNo
  [Input] User assigns the port number.

  **dwDOVal*
  [Output] This parameter will save the readback data from the output port.

- **Return value**
  Refer to section 1.2.

2.4 Interrupt Functions

2.4.1 Ixud_SetEventCallback Function

- **Syntax**

```
WORD Ixud_SetEventCallback(
        WORD wBoardNo,
        WORD wInterruptSource,
        HANDLE *hEvent,
        PVOID CallbackFun,
        DWORD dwCallBackParameter
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wEventType
[Input] User setting the event type,each bit is one mode tha can use at the sam times.

wEventType	模式
1	硬體中斷
2	類比資料準備完成
4	Event Active High
8	Event Active Low

表格 2-5 wEventType 參數設定

wInterruptSource
[Input] User setting the interrupt sourc for event,When setting is 0xFFFF that is turn on the all interrupt source.

**hEvent*
[Input] The event pointer, this event use the Windows API- CreatEvent(..) to creat, when set to NULL that the system will atuo creat the Event .

CallbackFun
[Input] User setting the callback functions.

dwCallBackParameter
[Input] User setting the parameter of the Callback Functions.

- **Return value**

Refer to section 1.2.

2.4.2 Ixud_RemoveEventCallback Function

- **Syntax**

```
WORD Ixud_RemoveEventCallback(
        WORD wBoardNo,
        WORD wInterruptSource
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.
wInterruptSource

[Input] User turn off the interrupt source.

- **Return value**
  Refer to section 1.2.

2.4.3 Ixud_InstallIrq Function

- **Syntax**
  WORD Ixud_InstallIrq(
  WORD wBoardNo**,**
  DWORD dwInterruptMask
  );
- **Parameters**
  wBoardNo
  [Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

- **Return value**
  Refer to section 1.2.

2.4.4 Ixud_RemoveIrq Function

- **Syntax**
  WORD Ixud_RemoveIrq(
  WORD wBoardNo
  );

- **Parameters**
  wBoardNo
  [Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

- **Return value**
  Refer to section 1.2.

2.5 Analog Input Function Member

2.5.1 Ixud_ConfigAI Function

This function by setting analogy input parameters. First need the call this function before use Analog Input Function Member.

- **Syntax**

```
WORD Ixud_ConfigAI(
        WORD wBoardNo,
        WORD wFIFOSizeKB,
        DWORD BufferSizeCount,
        WORD wCardType,
        WORD wDelaySettlingTime
);
```

- **Parameters**

wBoardNo

[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wFIFOSizeKB

[Input] User set the FIFO size according to build-in FIFO on board, The Unit is KByte. Setting value refer to board FIFO spec. when wFIFOSizeKB is 0.This value will set the minmul size of build-in FIFO on board.

wBufferSizeCount

[Input] User set the buffer data count. When setting value is 0, the default count are 524288 count(DWORD size) that total are about 2MB.

wCardType

[Input] User set it that according to board spec.

wCardType	Type	Support Board
0	L type /LU type/10V	PIO-821L,PISO-813(JP1=10V),PCI-1002L,PCI-1002LU,PCI-1202L,PCI-1202 LU,PCI-1602,PCI-1800L,PCI-1802L,PCI-822LU,PCI-826LU,PCI-827LU
1	H type/HU type/F type/20V	PIO-821H,PIO-827H,PISO-813(JP1=20V),PCI-1002H,PCI-1202H, PCI-1202HU,PCI-1602F, PCI-1800 H,PCI-1802 H

2-6 table is wCardType argument setting.

wDelaySettingTime

[Input] The settling weighting time of ADC. Unit：microsecond (μ s). This value will effect the performance for analog input, proposal value is 0.

- **Return value**

Refer to section 1.2.

2.5.2 Ixud_ClearAIBuffer Function

Clear analog input data from buffer.

- **Syntax**

```
WORD Ixud_ClearAIBuffer(
        WORD wBoardNo
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

- **Return value**

Refer to section 1.2.

2.5.3 Ixud_GetBufferStatus Function

Get the status of analog input buffer and data count.

- **Syntax**

```
WORD Ixud_GetBufferStatus(
        WORD wBoardNo,
        WORD *wBufferStatus,
        DWORD *dwDataCount
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

**wBufferStatus*
[Output] Get to analogy buffer of use status.

**wBufferStatus*	Status Description
0	None data.
1	Normal. Had data and not overflow.
2	Buffer Overflow.
3	System none allocate buffer.
4	FIFO overflow.
5	Other status.

2-7 table is analogy buffer status code declaration.

**dwDataCount*
[Output] Get to analog data count from buffer.

- **Return value**

Refer to section 1.2.

2.5.4 Ixud_ReadAI Function

Read single analog input data, this data type is floating.

- **Syntax**

```
WORD Ixud_ReadAI(
        WORD wBoardNo,
        WORD wChannel,
        WORD wConfig,
        float *fValue
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set analog input channel number.

wConfig
[Input] User set configuration code for analog input channel, this code please refer to section 1.3.1. This code can influence the data of precision and measurement range.

**fValue*
[Output] To store analog input data, this data type is floating.

- **Return value**

Refer to section 1.2.

2.5.5 Ixud_ReadAIH Function

Read single analog input data, this data type is hexadecimal.

- **Syntax**

```
WORD Ixud_ReadAIH(
        WORD wBoardNo,
        WORD wChannel,
        WORD wConfig,
        DWORD *dwValue
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set analog input channel number.

wConfig
[Input] User set configuration code for analog input channel, this code please refer to section 1.3.1. This code can influence the data of precision and measurement range.
**dwValue*
[Output] To store analog input data, this data type is hexadecimal.

- **Return value**

Refer to section 1.2.

2.5.6 Ixud_PollingAI Function

This function use polling mode to get much analog input data from the same channel.Those data type are floating.

- **Syntax**

```
WORD Ixud_PollingAI(
        WORD wBoardNo,
        WORD wChannel,
        WORD wConfig,
        DWORD dwDataCount,
        float fValue[ ]
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set analog input channel number.

wConfig
[Input] User set configuration code for analog input channel, this code please refer to section 1.3.1. This code can influence the data of precision and measurement range.

dwDataCount
[Input] User set analog input data count.

fValue[ ]
[Output] Please declare a floating type array (Array size must more than dwDataCount argument setting size). This array will store the analog input polling data; those data type are floating.

- **Return value**

Refer to section 1.2.

2.5.7 Ixud_PollingAIH Function

This function use polling mode to get much analog input data from the same channel.Those data type are hexadecimal.

- **Syntax**

```
WORD Ixud_PollingAIH(
        WORD wBoardNo,
        WORD wChannel,
        WORD wConfig,
        DWORD dwDataCount,
        DWORD dwValue[ ]
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set analog input channel number.

wConfig
[Input] User set configuration code for analog input channel, this code please refer to section 1.3.1. This code can influence the data of precision and measurement range.

dwDataCount
[Input] User set analog input data counts.

dwValue[ ]
[Output] Please declare a DWORD type array (Array size must more than dwDataCount argument setting size). This array will store the analog input polling data; those data type are hexadecimal.

- **Return value**

Refer to section 1.2.

2.5.8 Ixud_PollingAIScan Function

This function use polling mode to get analog input data from muilt-channels.This data type is floating.

- **Syntax**

```
WORD Ixud_PollingAIScan(
        WORD wBoardNo,
        WORD wChannels,
        WORD wChannelList[ ],
        WORD wConfigList[ ],
        DWORD dwDataCountPerChannel,
        float fValue[ ]
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannels
[Input] User set **wChannels** channels to acquire data.

wChannelList[ ]
[Input] Please declare a WORD type array (Array size must more than **wChannels** argument setting size). User set analog input channel number into array. From elements number 0 of the array start to set the first scan analog input channel number.

wConfigList[ ]
[Input] Please declare a WORD type array (Array size must more than **wChannels** argument setting size). User set analog input channel configuration code into array. From elements number 0 of the array start to set the first scan analog input channel configuration code, this configuration value code refer to section 1.3.1.

dwDataCountPerChannel
[Input] User set the sampling count for every channel.

fValue[ ]
[Output] Please declare a floating type array (Array size must more than wChannels X dwDataCountPerChannel argument setting size). This array will store the analog input polling data for every channel; those data type are floating. Every channel how to store this array, please refer to table 2-8.

- **Return value**

Refer to section 1.2.

- **Use example**

```
wChannes = 3                    // Total scan 3 channels
wChannelList[0]=5               // First, acquire analog input channel 5 of data
wChannelList[1]=3               // Second, acquire analog input channel 3 of data
wChannelList[2]=6               // Third, acquire analog input channel 6 of data
wConfigList[0]= IXUD_BI_10V     // Set measure range  +/-10V  for the first scan channel
wConfigList[1]= IXUD_BI_5V      // Set measure range +/-5V for the second scan channel
wConfigList[2]= IXUD_BI_2V5     // Set measure range +/-2.5V for the third scan channel
```

Call function after each channel of input value to store in array(fValue[]), and then to store sequence as follow table：

0	Channel5 Val0
1	Channel3 Val0
2	Channel6 Val0
3	Channel5 Val1

4	Channel3 Val1
5	Channel6 Val1

2-8 table is array to store sequence mode.

2.5.9 Ixud_PollingAIScanH Function

This function use polling mode to get analog input data from muilt-channels.This data type is hexadecimal.

- **Syntax**

```
WORD Ixud_PollingAIScanH(
        WORD wBoardNo,
        WORD wChannels,
        WORD wChannelList[ ],
        WORD wConfigList[ ],
        DWORD dwDataCountPerChannel,
        DWORD dwValue[ ]
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannels
[Input] User set **wChannels** channels to acquire data.

wChannelList[ ]
[Input] Please declare a WORD type array (Array size must more than **wChannels** argument setting size). User set analog input channel number into array. From elements number 0 of the array start to set the first scan analog input channel number.

wConfigList[ ]
[Input] Please declare a WORD type array (Array size must more than **wChannels** argument setting size). User set analog input channel configuration code into array. From elements number 0 of the array start to set the first scan analog input channel configuration code, this configuration value code refer to section 1.3.1.

dwDataCountPerChannel
[Input] User set the sampling count for every channel.

dwValue[ ]
[Output] Please declare a floating type array (Array size must more than wChannels X dwDataCountPerChannel argument setting size). This array will store the analog input polling data for every channel; those data type are hexadecimal. Every channel how to store this array, please refer to table 2-9.

- **Return value**

Refer to section 1.2.

- **Use example**

```
wChannes = 3                    // Total scan 3 channels
wChannelList[0]=5               // First, acquire analog input channel 5 of data
wChannelList[1]=3               // Second, acquire analog input channel 3 of data
wChannelList[2]=6               // Third, acquire analog input channel 6 of data
wConfigList[0]= IXUD_BI_10V     // Set measure range  +/-10V  for the first scan channel
wConfigList[1]= IXUD_BI_5V      // Set measure range +/-5V for the second scan channel
wConfigList[2]= IXUD_BI_2V5     // Set measure range +/-2.5V for the third scan channel
```

Call function after each channel of input value to store in array(dwValue[]), and then to store mode as follow table：

0	Channel5 Val0
1	Channel3 Val0
2	Channel6 Val0
3	Channel5 Val1

4	Channel3 Val1
5	Channel6 Val1

2-9 table is array to store mode.

2.5.10 Ixud_StartAI Function

Set sampling rate for single analog input channel and then start analog input pacer to acquire analog input data for single channel that store into memory. Those data can through Ixud_GetAIBuffer and Ixud_GetAIBufferH function to get, want to stop anlog input operation function, it need call Ixud_StopAI function to stop.

- **Syntax**

```
WORD Ixud_StartAI(
        WORD wBoardNo,
        WORD wChannel,
        WORD wConfig,
        float fSamplingRate,
        DWORD dwDataCount
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set analog input channel number.

wConfig
[Input] User set configuration code for analog input channel, this code please refer to section 1.3.1. This code can influence the data of precision and measurement range.
fSamplingRate
[Input] User set sampling rate(number of samples per second) of analog input channel. This value type is floating.

dwDataCount
[Input] User set analog input data counts.

- **Return value**

Refer to section 1.2.

2.5.11 Ixud_StartAIScan Function

Set sampling rate for multi analog input channel and then start analog input pacer to acquire analog input data for multi-channel that store into memory. Those data can through Ixud_GetAIBuffer and Ixud_GetAIBufferH function to get, want to stop anlog input operation function, it need call Ixud_StopAI function to stop.

- **Syntax**

```
WORD Ixud_StartAIScan(
        WORD wBoardNo,
        WORD wChannels,
        WORD wChannelList[ ],
        WORD wConfigList[ ],
        float fSamplingRate,
        DWORD dwDataCountPerChannel
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannels
[Input] User set **wChannels** channels to acquire data.

wChannelList[ ]
[Input] Please declare a WORD type array (Array size must more than **wChannels** argument setting size). User set analog input channel number into array. From elements number 0 of the array start to set the first scan analog input channel number.

wConfigList[ ]
[Input] Please declare a WORD type array (Array size must more than **wChannels** argument setting size). User set analog input channel configuration code into array. From elements number 0 of the array start to set the first scan analog input channel configuration code, this configuration value code refer to section 1.3.1. .

fSamplingRate
[Input] User set sampling rate(number of samples per second) of total analog input channel. This value type is floating.This sampling rate will share to all scan channel average.

dwDataCountPerChannel
[Input] User set the sampling count for every channel.

- **Return value**

Refer to section 1.2.

2.5.12 Ixud_GetAIBuffer Function

This function will get to analog input data from memory. Those data type are floating.It need call Ixud_StartAI or Ixud_StartAIScan to get analog input data before call this function.

- **Syntax**

```
WORD Ixud_GetAIBuffer(
        WORD wBoardNo,
        DWORD dwDataCount,
        float fValue[ ]
        );
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

dwDataCount
[Input] User set analog input data counts.

fValue[ ]
[Output] Please declare a floating type array (Array size must more than dwDataCount argument setting size). This array will get the analog input pacer data from memory; those data type are floating.

- **Return value**

Refer to section 1.2.

2.5.13 Ixud_GetAIBufferH Function

This function will get to analog input data from memory. Those data type are hexadecimal.It need call Ixud_StartAI or Ixud_StartAIScan to get analog input data before call this function.

- **Syntax**

```
WORD Ixud_GetAIBufferH(
        WORD wBoardNo,
        DWORD dwDataCount,
        DWORD hValue[ ]
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

dwDataCount
[Input] User set analog input data counts.

hValue[ ]
[Output] Please declare a hexadecimal type array (Array size must more than dwDataCount argument setting size). This array will get the analog input pacer data from memory; those data type are hexadecimal.

- **Return value**

Refer to section 1.2.

2.5.14 Ixud_StopAI Function

Call this function could stop Ixud_StartAI and Ixud_StartAIScan operation.

- **Syntax**

```
WORD Ixud_StopAI(
        WORD wBoardNo
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

- **Return value**

Refer to section 1.2.

2.6 Analog Output Function Member

2.6.1 Ixud_ConfigAO Function

This function to use by setting analogy output of some parameter values, to need call this function before use Analog Output Function Member.

- **Syntax**

```
WORD Ixud_ConfigAO(
        WORD wBoardNo,
        WORD wChannel,
        WORD wCfgCode
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set analog output channel number.

wCfgCode
[Input] User set configuration code for analog output channel, this code please refer to section 1.3.2. This code can influence the data of precision and measurement range.

- **Return value**

Refer to section 1.2.

2.6.2 Ixud_WriteAOVoltage Function

This function will send the data of voltage to D/A port, this data type is floating.

- **Syntax**

```
WORD Ixud_WriteAOVoltage(
        WORD wBoardNo,
        WORD wChannel,
        float fValue
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set analog output channel number.

fValue
[Input] User set the data of voltage to send D/A port. This data type is floating. Unit：Volt (V)

- **Return value**

Refer to section 1.2.

2.6.3 Ixud_WriteAOVoltageH Function

This function will send the data of voltage to D/A port, this data type is floating.

- **Syntax**

```
WORD Ixud_WriteAOVoltageH(
        WORD wBoardNo,
        WORD wChannel,
        DWORD hValue
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set analog output channel number.

hValue
[Input] User set the data of voltage to send D/A port. This data type is hexadecimal.

- **Return value**

Refer to section 1.2.

2.6.4 Ixud_WriteAOCurrent Function

This function will send the data of current to D/A port, this data type is floating.

- **Syntax**

```
WORD Ixud_WriteAOCurrent(
        WORD wBoardNo,
        WORD wChannel,
        float fValue
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set analog output channel number.

fValue
[Input] User set the data of current.This data type is floating. Unit：mille Ampere (mA)

- **Return value**

Refer to section 1.2.

2.6.5 Ixud_WriteAOCurrentH Function

This function will send the data of current to D/A port, this data type is hexadecimal.

- **Syntax**

```
WORD Ixud_WriteAOCurrentH(
        WORD wBoardNo,
        WORD wChannel,
        DWORD hValue
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set analog output channel number.

hValue
[Input] User set the data of current.This data type is hexadecimal. Unit：mille Ampere (mA)

- **Return value**

Refer to section 1.2.

2.7 Timer/Counter Function Member

2.7.1 Ixud_ReadCounter Function

This function could read Timer/Counter data.

- **Syntax**

```
WORD Ixud_ReadCounter(
        WORD wBoardNo,
        WORD wChannel,
        DWORD *dwValue
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set the channel number of Timer/Counter, while **wChannel** is 0 that is first channel, and **wChannel** is1 that is second channel.And so on.

**dwValue*
[Output] Stroe the read data from Timer/Counter.This data type is hexadecimal.

- **Return value**

Refer to section 1.2.

2.7.2 Ixud_SetCounter Function

Set value and mode of the Timer/Counter.

- **Syntax**

```
WORD Ixud_SetCounter(
        WORD wBoardNo,
        WORD wChannel,
        WORD wMode,
        DWORD dwValue
);
```

- **Parameters**

wBoardNo

[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel

[Input] User set the channel number of Timer/Counter, while **wChannel** is 0 that is first channel, and **wChannel** is1 that is second channel

wMode

[Input] User set mode of the Timer/Counter.

wMode	Mode name
0	Interrupt on Terminal Count
1	Hardware Retriggerable One-Shot
2	Rate Generator
3	Square Wave Mode
4	Software Triggered Mode
5	Hardware Triggered Strobe

Table 2-10 is wMode parameter setting.

dwValue

[Input] User set value of the Timer/Counter.

- **Return value**

Refer to section 1.2.

2.7.3 Ixud_DisableCounter Function

Stop (Lunch) to Timer/Counter.

- **Syntax**

```
WORD Ixud_DisableCounter(
        WORD wBoardNo,
        WORD wChannel
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set the channel number of Timer/Counter, while **wChannel** is 0 that is first channel, and **wChannel** is1 that is second channel.

- **Return value**

Refer to section 1.2.

2.7.4 Ixud_SetFCChannelMode Function

Set mode of the Timer/Counter(only for PCI-FC16U).

- **Syntax**

WORD Ixud_SetFCChannelMode(
WORD wBoardNo**,**
WORD wChannel**,**
WORD wMode**,**
WORD wDelayMs
);

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set the channel number of Timer/Counter, while **wChannel** is 0 that is first channel, and **wChannel** is1 that is second channel

wMode
[Input] User set channel mode.

wMode	Mode name
0	-
1	-
2	Down Count
3	-
4	-
5	-

Table 2-11 is wMode parameter setting.

wDelayMs
[Input] User set the delay time, the delay time depend on measuring frequency.The unit is ms.

- **Return value**

Refer to section 1.2.

2.7.5 Ixud_ReadFrequency Function

This function could read frequency from signal. (Only for the PCI-FC16U)

- **Syntax**

WORD Ixud_ReadFrequency(
WORD wBoardNo**,**
WORD wChannel**,**
float ***fFrequency,**
DWORD dwTimeOutMs**,**
**WORD ***wStatus
);

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

wChannel
[Input] User set the channel number of Timer/Counter, while **wChannel** is 0 that is first channel, and **wChannel** is1 that is second channel.And so on.

**fFrequency*
[Output] Stroe the frequency data form signal.This data type is float.The unit is Hz.

dwTimeOutMs
[Input] User set the timeout for timer/counter.The unit is ms.

**wStatus*
[Output] get the timer/counter stauts.

wStatus	Status
0	Get frequency
1	Timeout
2	Launch frequency

Table 2-12 wStatus definition

- **Return value**
  Refer to section 1.2.

2.8 Memory Input/ Output Function Member

2.8.1 Ixud_ReadMemory Function

This function could read data from memory. The data size is 8/16/32 bit.

- **Syntax**

```
WORD Ixud_ReadMemory(
        WORD wBoardNo,
        DWORD dwOffset,
        WORD Size,
        DWORD *dwValue
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

dwOffset
[Input] User set the memory address.

wSize
[Input] User set the data size.

wSize	Size (bit)
8	8 (Byte)
16	16 (WORD)
32	32 (DWORD)

Table 2-13 wSize definition

dwValue
[Output] Read the data value.

- **Return value**

Refer to section 1.2.

2.8.2 Ixud_WriteMemory Function

This function could write data from memory. The data size is 8/16/32 bit.

- **Syntax**

```
WORD Ixud_WriteMemory(
        WORD wBoardNo,
        DWORD dwOffset,
        WORD Size,
        DWORD *dwValue
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

dwOffset
[Input] User set the memory address.

wSize
[Input] User set the data size.

wSize	Size (bit)
8	8 (Byte)
16	16 (WORD)
32	32 (DWORD)

Table 2-14 wSize definition

dwValue
[Input] Write the data value

- **Return value**

Refer to section 1.2.

2.8.3 Ixud_ReadMemory32 Function

This function could read data from memory. The data size is 32 bit, this function support the compiler that don't have unsigned integer data type, ex: Visual Basic 6.0.

- **Syntax**

```
WORD Ixud_ReadMemory(
        WORD wBoardNo,
        DWORD dwOffset,
        WORD *dwLow,
        DWORD *dwHigh
);
```

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

dwOffset
[Input] User set the memory address.

dwLow
[Output] Read the data value of low part.

dwHigh
[Output] Read the data value of high part.

- **Return value**

Refer to section 1.2.

2.8.4 Ixud_WriteMemory32 Function

This function could wirte data from memory. The data size is 32 bit, this function support the compiler that don't have unsigned integer data type, ex: Visual Basic 6.0.

- **Syntax**

WORD

- **Parameters**

wBoardNo
[Input] The user-assigned board number, where wBoardNo=0 is the first board, and wBoardNo=1 is the second board, and so on.

dwOffset
[Input] User set the memory address.

dwValue
[Input] Write the data value for low part.

dwValue
[Input] Write the data value for high part.

- **Return value**

Refer to section 1.2.