D2K-OCX 32-bit ActiveX controls for NuDAQ DAQ-2K/PXI-2K Data Acquisition Cards

Programmer's Guide

Manual Rev: 1.00: November 15, 2002

Getting Service from ADLINK

? ? **Customer Satisfaction is always the most important thing for ADLINK Tech Inc. If you need any help or service, please contact us and get it.**

ADLINK Technology Inc.			
Web Site	http://www.adlinktech.com		
Sales & Service	service@adlinktech.com		
Technical	NuDAQ + USBDAQ	nudaq@adlinktech.com	
Support	NuDAM	automation@adlinktech.com	
	NuIPC	nuipc@adlinktech.com	
	NuPRO/EBC	nupro@adlinktech.com	
TEL	+886-2-82265877	FAX	+886-2-82265717
Address	9F, No. 166, Jian Yi Road, Chungho City, Taipei, 235 Taiwan, R.O.C.		

? ? **Please inform or FAX us of your detailed information for a prompt, satisfactory and constant service.**

Detailed Company Information			
Company/Organization			
Contact Person			
E-mail Address			
Address			
Country			
TEL		FAX	
Web Site			
Questions			
Product Model			
Environment to Use	✍OS: _____ ✍Computer Brand: _____ ✍M/B: ✍CPU: ✍Chipset: ✍Bios: ✍Video Card: ✍Network Interface Card: ✍Other:		
Challenge Description			
Suggestions for ADLINK			

Table of Contents

How to Use This Guide

This manual is designed to help you use the NuDAQ D2K-OCX ActiveX controls to control NuDAQ PCI/PXI data acquisition cards.

The *Programmer's Guide* is organized as follows:

Part 1, *NuDAQ Configuration*, describes how you can use the NuDAQ Configuration Utility to configure NuDAQ cards on Windows 98/NT/2000.

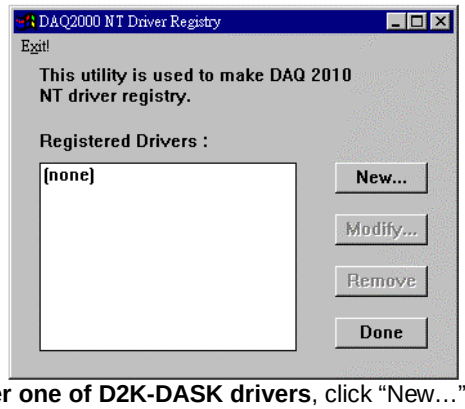
Part 2, *ActiveX Controls Function Reference*, contains the detailed descriptions of each NuDAQ D2K-OCX ActiveX controls.

DAQ-2000 Registry/Configuration utility (D2kUtil)

D2kUtil is used for the users to ***register*** D2K-DASK drivers (Windows NT4 only), ***remove*** installed drivers (Windows NT4 only), and ***set/modify*** the allocated buffer sizes of AI, A O, DI and DO. The default location of this utility is <InstallDir> \Util directory.

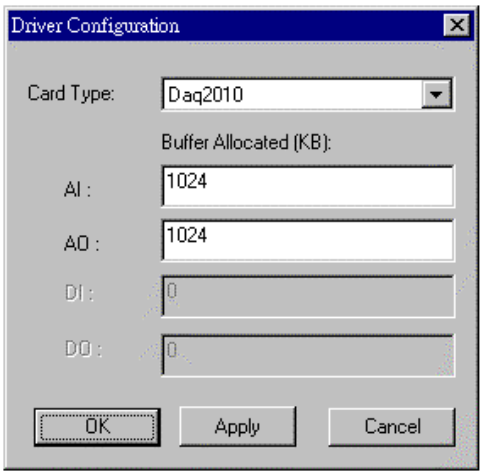
[D2kUtil in Windows NT]

The *D2kUtil* main window is shown as the following window. If any D2K -DASK/NT driver has been registered, it will be shown on the *Registered Driver* list.



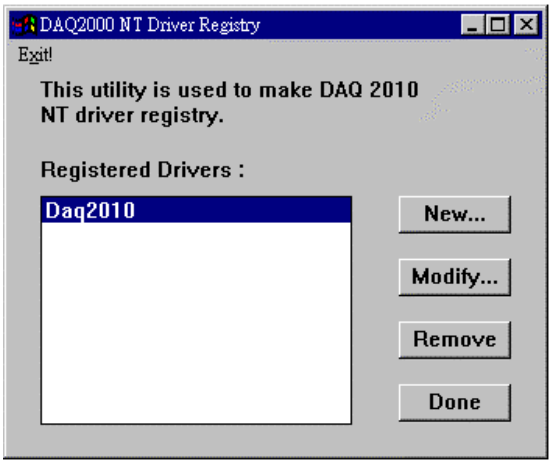
To register one of D2K-DASK drivers, click "New…" button and a *Driver Configuration* window appears.

In this window, users can select the driver you want to register and input the parameters in the box

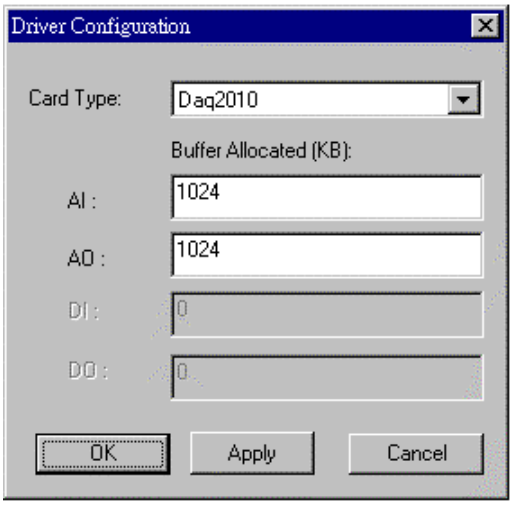


corresponding to AI, AO, DI, or DO for the r equirement of your applications. The "Buffer Allocated" of AI, AO, DI, DO represent the sizes of contiguous Initially Allocated memory for continuous analog input, analog output, digital input, digital output respectively. Its unit is KB, i.e. 1024 bytes. Device driver will try to allocate these sizes of memory at system startup time. The size of initially allocated memory is the maximum memory size that DMA or Interrupt transfer can be performed. It will induce an unexpected result in that DMA or Interrupt transfer performed exceeds the initially allocated size.

After the device configurations of the driver you select is finished, click "OK" to register the driver and return to the *D2kUtil* main window. The driver you just registered will be shown on the re gistered driver list as the following figure:



Using *D2kUtil* to ***change the buffer allocated settings*** of one of the D2K -DASK drivers, select the driver from the *Registered Driver* list and click "Modify…" button and then a "Driver Configuration" window is shown as below.



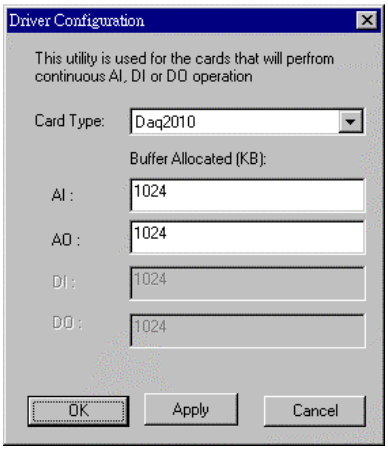
Inside the allocated buffer size fields of AI, AO, DI and DO are the originally set values. Type the value in the box corresponding to AI, AO, DI, or DO according to the requirement of your applications, and then click "OK" button.

To ***remove*** a registered driver, select the driver from the *Registered Driver* list in The *D2kUtil* main window and click "Remove" button. The selected driver will be deleted from the registry table.

[D2kUtil in Windows 98]

This utility is used to ***set/modify*** the allocated buffer sizes of AI, AO, DI and DO. The allocated buffer sizes of AI, AO, DI, DO represent the sizes of contiguous Initially Allocated memory for continuous analog input, analog output, digital input, digital output respectively. Its unit is pag e *KB*, i.e. 1024 bytes. Device driver will try to allocate these sizes of memory at system startup time. The size of initially allocated memory is the maximum memory size that DMA or Interrupt transfer can be performed. It will induce an unexpected result i n that DMA or Interrupt transfer performed exceeds the initially allocated size.

The "Driver Configuration" window is shown as below.



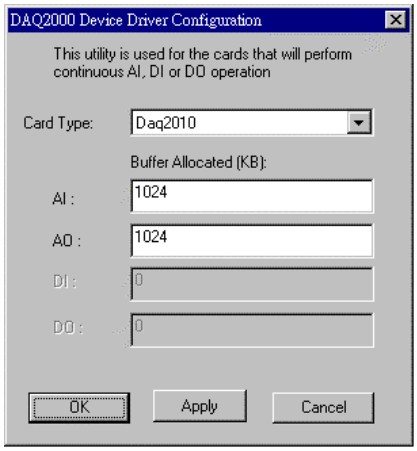
Using *D2kUtil* to ***change the buffer allocated settings*** of one of the D2K -DASK drivers, select the driver from the *Card Type combo box*.

Inside the allocated buffer size fields of AI, AO, DI and DO are the originally set values. Type the value in the box corresponding to AI, AO, DI, or DO according to the requirement of your applications, and then click "Apply" button.

[D2kUtil in Windows 2000]

This utility is used to *set/modify* the allocated buffer sizes of AI, AO, DI and DO. The allocated buffer sizes of AI, AO, DI, DO represent the sizes of contiguous Initially Allocated memory for continuous analog input, analog output, digital input, digital output respectively. Its unit is page *KB*, i.e. 1024 bytes. Device driver will try to allocate these sizes of memory at system startup time. The size of initially allocated memory is the maximum memory size that DMA or Interrupt transfer can be performed. It will induce an unexpected result in that DMA or Interrupt transfer performed exceeds the initially allocated size.

The "Driver Configuration" window is shown as below.



Using *D2kUtil* to *change the buffer allocated settings* of one of the D2K-DASK drivers, select the driver from the *Card Type combo box*.

Inside the allocated buffer size fields of AI, AO, DI and DO are the originally set values. Type the value in the box corresponding to AI, AO, DI, or DO according to the requirement of your applications, and then click "Apply" button.

Daq2005 ActiveX Control

The Daq2005 ActiveX control is a software component that provides the interface for users to control PCI-2005 card.

DAQ2005 ActiveX Control Overview			
Type (Group)	**Property**	**Method**	**Event**
General	DASKCardType	Open	DAQError
	CardNumber	AboutBox	
	OpenMode	ShowPropertyPages	
	DaskCardID		
DIO	P1ADir	ReadBackDOLine	
	P1BDir	ReadBackDOPort	
	P1CLowerDir	ReadDILine	
	P1CUpperDir	ReadDIPort	
		WriteDOLine	
		WriteDOPort	
GPTC	Mode	Start	
	ClockSource	Stop	
	ClockPolarity	Reset	
	GateSource	ReadStatus	
	GatePolarity		
	UpDownSource		
	UpDownPolarity		
	OutputPolarity		
	IntGATE		
	IntUpDnCTR		
	DelayCount		
	DurationCount		
	OutputValue		
AI	NumOfScan	StartContAI	AiComplete
	Channels(n).Enable	StopContAI	AiHalfReady
	Channels(n).Range	ReadChannels	
	ClockSource		
	ScanInterval		
	ConversionSource		
	TriggerMode		
	TriggerSource		
	DelaySource		
	ReTriggerModeEnable		
	MCounterEnable		
	PostTriggerCount		
	DelayCount		

DAQ2005 ActiveX Control Overview			
	MCount		
	ReTriggerCount		
	ExtTrigPolarity		
	ReturnType		
	DoubleBufferMode		
	StreamToFile		
	FileName		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
AO	CHUI	StartContAO	AoComplete
	DAWRSource	StopContAO	AoBufferReady
	TriggerSource	WriteChannel	
	TriggerMode		
	Delay1Source		
	Delay2Source		
	ExtTrigPolarity		
	ReTriggerCount		
	Delay1Count		
	Delay2Count		
	ClockSource		
	ReTriggerModeEnable		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
	DoubleBufferMode		
	Definite		
	StopMode		
	Channels(n).Enable		
	Channels(n).IntOrExtref		
	Channels(n).OutputPolarity		
	Channels(n).RefVoltage		
	Channels(n).Buffer1		
	Channels(n).Buffer2		
CALIBRATION	BankTemperature	AutoCalibration	AcquireADError
	BankDate	Load	AcquireDAError

DAQ2005 ActiveX Control Overview			
	CurrentTemperature	Save	
	CurrentDate	DisplayErrors	
SSI	TIMEBASE	ClearAll	
	ADCONV		
	DAWR		
	ADTRIG		
	DATRIG		

DAQ_2005 Properties

DASKCardType Property

Return a value that determines the card type. It is always DAQ_2005 in DAQ-2005 device.

.

Syntax

[Integer] =object.**CardType**

Remarks

DAQ_2005 (for DAQ-2005)

Settings

Value	Constant	Description
4	DAQ_2005	For DAQ-2005 Device

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Data Type

Integer

CardNumber Property

The sequence number of the card with ***the same card type*** plugged in the PCI slot. The card sequence number setting is according to the PCI slot sequence in the mainboard. The first card (in the most prior slot) is with CardNumber=0. For example, if there are two DAQ -2005 cards plugged on your PC, the DAQ -2005 card in the prior slot should be registered w ith CardNumber =0, and the other one with CardNumber =1.

Syntax

object.**CardNumber** [= short]

Remarks

This property will be used when Initializes the hardware states of a DAQ -2K data acquisition card.

Data Type

Integer

OpenMode Property

Return/Set a value that determines the mode of opening device.

Syntax

object.**OpenMode** [= short]

Settings

Value	Display	Description
0	Automatic	D2K-OCX will initialize by itself. Automatically open device when the control was created
1	Manual	Call object**.Open**() method to initialized D2K-OCX(open device),
2	Demonstration	D2K-OCX will provide software DAQ data to simulating hardware drivers.

Data Type

Integer

DaskCardID Property

Returns a value that determines the D2K_Register_Card() returns value,
the DaskCardID is a numeric card ID that will be used by other D2K -DASK library functions.

Syntax

[short] =object**.DaskCardID**

Remarks

The range of card id is between 0 and 31.

This property will be used when combine D2K-DASK and D2K-OCX two module in one program.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example:

```
'This sample will demonstration if user want to check AI completed by itself.

' OCX to starting continuous AI
  Dim nDaskID as Integer
  Daq2005.Open(TRUE)
  nDaskID = Daq2005.DaskCardID

Daq2005.AI.StartContAI

' Check AI Completed by DASK    API
Dim Stopped    As Byte
Dim AccessCnt As Long

Do While True
    D2K_AI_AsyncCheck nDaskID, Stopped, AccessCnt
    If Stopped = 1 Then
        Exit Do
    End If
Loop
MsgBox "AI Complete"
```

DIO.P1Adir Property

Return/Set a value that determines P1A port direction.

Syntax

object.**DIO.P1ADir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1Bdir Property

Return/Set a value that determines P1B port direction.

Syntax

object.**DIO.P1BDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CLowerdir Property

Return/Set a value that determines P1C lower port direction.

Syntax

object.**DIO.P1CLowerDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CUpperdir Property

Return/Set a value that determines P1C upper port direction.

Syntax

object.**DIO.P1CUpperDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.Mode Property

Return/Set a value that determines the Timer/Counter mode.

Syntax

object.**GPTC.Counter0.Mode** [= Integer]

Settings

Value	Constant	Description
1	SimpleGatedEventCNT	
2	SinglePeriodMSR	
3	SinglePulseWidthMSR	
4	SingleGatedPulseGen	
5	SingleTrigPulseGen	
6	RetrigSinglePulseGen	
7	SingleTrigContPulseGen	
8	ContGatedPulseGen	

Remarks

Please refer to the hardw are manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic

Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockSource Property
GPTC.Counter1.ClockSource Property

Return/Set a value that determines the Timer/Counter Source.

Syntax

object.**GPTC.Counter0.ClockSource** [= Integer]
object.**GPTC.Counter1.ClockSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKSRC_INT	internal time base
2	GPTC_CLKSRC_EXT	external time base from GPTC0_SRC or GPTC1_SRC pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockPolarity Property
GPTC.Counter1.ClockPolarity Property

Return/Set a value that determines the Timer/Counter clock polarity

Syntax

object.**GPTC.Counter0.ClockPolarity** [= Integer]
object.**GPTC.Counter1.ClockPolarity** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKEN_LACTIVE	Low active
2	GPTC_CLKEN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi

Unit: D2kDask.pas

GPTC.Counter0.GateSource Property
GPTC.Counter1.GateSource Property

Return/Set a value that determines the Timer/Counter gate source

Syntax

object.**GPTC.Counter0.GateSource** [= Integer]
object.**GPTC.Counter1.GateSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_GATESRC_INT	gate is controlled by software
4	GPTC_GATESRC_EXT	gate is controlled by GPTC0_GATE or GPTC1_GATE pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GatePolarity Property
GPTC.Counter1.GatePolarity Property

Return/Set a value that determines the Timer/Counter gate polarity

Syntax

object.**GPTC.Counter0.GatePolarity** [= Integer]
object.**GPTC.Counter1.GatePolarity** [= Integer]

Settings

Value	Constant	Description
2	GPTC_GATE_LACTIVE	Low active
0	GPTC_GATE_HACTIVE	High active

Please refer to the hardware manual for the mod e description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownSource Property
GPTC.Counter1.UpDownSource Property

Return/Set a value that determ ines the Timer/Counter UpDown Source

Syntax

object.**GPTC.Counter0.UpDownSource** [= Integer]
object.**GPTC.Counter1.UpDownSource** [= Integer]

Settings

Value	Constant	Description
0	GPTC_UPDOWN_SEL_INT	Up/Down controlled by software
16	GPTC_UPDOWN_SEL_EXT	Up/Down controlled by GPTC0_UPDOWN orGPTC1_UPDOWN pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownPolarity Property
GPTC.Counter1.UpDownPolarity Property

Return/Set a value that determines the Timer/Counter UpDown Polarity

Syntax

object.**GPTC.Counter0.UpDownPolarity** [= Integer]
object.**GPTC.Counter1.UpDownPolarity** [= Integer]

Settings

Value	Constant	Description
4	GPTC_UPDOWN_LACTIVE	Low active
0	GPTC_UPDOWN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.OutputPolarity Property
GPTC.Counter1.OutputPolarity Property

Return/Set a value that determines the Timer/Counter Output Polarity

Syntax

object.**GPTC.Counter0.OutputPolarity** [= Integer]
object.**GPTC.Counter1.OutputPolarity** [= Integer]

Settings

Value	Constant	Description
8	GPTC_OUTPUT_LACTIVE	Low active
0	GPTC_OUTPUT_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.IntGATE Property
GPTC.Counter1.IntGATE Property

Return/Set a value that determines the Timer/Counter Internal gate initialized status

Syntax

object.**GPTC.Counter0.IntGATE** [= Boolean]
object.**GPTC.Counter1.IntGATE** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type

Boolean

GPTC.Counter0.IntUpDnCTR Property
GPTC.Counter1.IntUpDnCTR Property

Return/Set a value that determines the Timer/Counter internal updown counter initialized status.

Syntax

object.**GPTC.Counter0.IntUpDnCTR** [= Boolean]
object.**GPTC.Counter1.IntUpDnCTR** [= Boolean]

Settings

Value	Constant	Description

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type
Boolean

GPTC.Counter0.DelayCount Property
GPTC.Counter1.DelayCount Property

Return/Set a value that determines the Timer/Counter internal initial count of the GPTC or pulse delay. The counter value of load register 1 of timer/counter. The meaning for the value depends on the mode the timer /counter performs. For mode 1 to mode 3, the value is the initial count of the GPTC. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse delay.

Syntax
object.**GPTC.Counter0.DelayCount** [= Integer]
object.**GPTC.Counter1.DelayCount** [= Integer]

Data Type
Integer

GPTC.Counter0.DurationCount Property
GPTC.Counter1.DurationCount Property

Return/Set a value that determines the Timer/Counter pulse width when mode 4 to mode 8.
The counter value of load register 2 of timer/counter. For mode 1 to mode 3, the value is not used. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse width.

Syntax
object.**GPTC.Counter0.DurationCount** [= Integer]
object.**GPTC.Counter1.DurationCount** [= Integer]

Data Type
Integer

GPTC.Counter0.OutputValue Property
GPTC.Counter1.OutputValue Property

Returns the counter value of the specified general -purpose timer/counter.
Range: 0 through 65535

Syntax
[Integer=] object.**GPTC.Counter0.OutputValue**
[Integer=] object.**GPTC.Counter1.OutputValue**

Data Type
Integer

GPTC.CALIBRATION.BankTemperature Property

Returns a value that since user's last calibrated temperature in the EEPRON Bank .

Syntax
object.**CALIBRATION.BankTemperature**(*[BankOfEEPROM as Integer]*) As Single

Data Type
Single

GPTC.CALIBRATION.BankDate Property

Returns a value that since user's last calibrated date in the EEPRON Bank .

Syntax
object.**CALIBRATION.BankDate**(*[BankOfEEPROM as Integer]*) As String

Data Type
String

GPTC.CALIBRATION.CurrentTemperature Property

Returns a value that determines the current temperature on card.

Syntax
[Single=] object.**CALIBRATION.CurrentTemperature**

Data Type
Single

GPTC.CALIBRATION.CurrentDate Property

Returns a value that determines the current date.

Syntax
[String=] object.**CALIBRATION.CurrentDate**

Data Type
String

AI.NumOfScan Property

If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, NumOf *Scan* is the size (in samples) allocated for each channel in the circular buffer. This value must be a multiple of 2.

Syntax
object.**AI.NumOfScan** [=Long]

Remarks

Non-double-buffer mode
This value multiply the total number of scan channels is the total number of A/D conversions to be performed.
Double-buffer-mode
This value multiply the total number of scan channels is the size (in sample) of the circular buffer.

Data Type
Long

AI.ClockSource Property

The clock source (Time Base) the device selected.

Syntax
object.**AI.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ScanInterval Property

The length of the scan interval (that is, the counter value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 2 through 16777215.

If the timer base is *Internal timer*, the valid range of the value is 80 thro ugh 16777215

Syntax
object.**AI.ScanInterval** [=Long]

Data Type
Long

AI.ConversionSource Property

The A/D Conversion Source Selection.

Syntax
object.**AI.ConversionSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_ADCONVSRC_Int	Internal timer
4	DAQ2K_AI_ADCONVSRC_AFI0	From AFI0 pin
8	DAQ2K_AI_ADCONVSRC_SSI	From SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerMode Property

The Trigger Mode Selection.

Syntax

object**.AI.TriggerMode** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_TRGMOD_POST	Post Trigger Mode
8	DAQ2K_AI_TRGMOD_DELAY	Delay Trigger Mode
16	DAQ2K_AI_TRGMOD_PRE	Pre-Trigger Mode
24	DAQ2K_AI_TRGMOD_MIDL	Middle-Trigger Mode

Remarks

Please refer to the hardware manual for the trigger mode description.

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerSource Property

The Trigger Source Selection.

Syntax

object**.AI.TriggerSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_TRGSRC_SOFT	Software
1	DAQ2K_AI_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_AI_TRGSRC_ExtD:	From external digital trigger pin
3	DAQ2K_AI_TRSRC_SSI	From SSI source

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.DelaySource Property

Delay source selection.

Syntax

object.**AI. DelaySource** [=Short]

Settings

Value	Constant	Description
256	DAQ2K_AI_Dly1InSamples	Delay in samples
0	DAQ2K_AI_Dly1InTimebase	Delay in time base

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReTriggerModeEnable Property

Re-trigger in an acquisition is enabled

Syntax

object.**AI.ReTriggerModeEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	ReTriggerMode is disabled.
1	True	ReTriggerMode is enabled.

Data Type

Boolean

AI.MCounterEnable Property

This constant is only valid for Pre -trigger and Middle trigger mode . Mcounter is enabled and then the trigger signal is ignore before M terminal count is reached.

Syntax

object.**AI.MCounterEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	MCounter is disabled.
1	True	Mcounter is enabled.

Data Type
Boolean

AI.PostTriggerCount Property

This constant is only valid for Middle trigger mode,
The PostTriggerCount indicates the number of data will be accessed after a specific trigger event.

Syntax
object.**AI.PostTriggerCount** [=Long]

Data Type
Long

AI.DelayCount Property

This constant is only valid for Delay trigger mode,
The DelayCount indicates the number of data or timer ticks will be ignored after a specific trigger event.

Syntax
object.**AI.DelayCount** [=Long]

Data Type
Long

AI.MCount Property

The counter value of MCounter . This argument is only valid for Pretrigger and Middle trigger mode.

Syntax
object.**AI.MCount** [=Long]

Data Type
Long

AI.ReTriggerCount Property

The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax
object.**AI.ReTriggerCount** [=Long]

Data Type
Long

AI.ExtTrigPolarity Property

External Digital **Trigger Polarity.**

Syntax
object.**AI.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_AI_TrgPositive	Trigger positive edge active
4096	DAQ2K_AI_TrgNegative	Trigger negative edge active

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReturnType Property

Return/Set a value that determines the return data type of analog input when AiComplete or AiHalfReady e vent would occur.

Syntax

object.**AI.ReturnType** [=Integer]

Settings

Value	Constant	Description
0		Scaled data only
1		Binary codes without channel only
2		Scaled data and binary codes without channel
3		No data return

Data Type

Integer

AI.Double BufferMode Property

Enables or disables double -buffered data acquisition mode.

Syntax

object.**AI.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type

Boolean

AI.StreamToFile Property

Return/Set a value that determines if the control is enabled the function of streaming data to disk file. This argument is only valid for Delay trigger

Syntax

object.**AI.StreamToFile** [=Boolean]

Settings

Value	Constant	Description
0	False	Disable the function of streaming data to disk file.
1	True	Enable the function of streaming data to disk file

Data Type

Boolean

AI.FileName Property

FileName specified the file name of streaming data to disk. This argument i s only valid for AI.StreamToFile is Enable.

Syntax

object.**AI.FileName** [=String]

Data Type

String

AI.AIOAnalogTrigCtrl Property

Return/Set a value that determines the t rigger source selection

Syntax

object.**AI.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	CH0ATRIG	AI channel 0
2	CH1ATRIG	AI channel 1
4	CH2ATRIG	AI channel 2
6	CH3ATRIG	AI channel 3
1	EXTATRIG	From external analog trigger pin

Data Type

Integer

AI.AIOTrigCondition Property

Return/Set a value that determi nes the trigger condition selection

Syntax

object.**AI.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above -High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks

Please refer to the hardware manual for the more description.

Data Type

Integer

AI.AIOHLevel Property

The High value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AI.AIOHLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AI.AIOLLevel Property

The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AI.AIOLLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AI.Channels(0).Enable Property
AI.Channels(1).Enable Property
AI.Channels(2).Enable Property
AI.Channels(3).Enable Property

DAQ-2000 channel-gain that can be set separately for each channel to perform multi-channel/gain analog input, The parameter 0~3 is channel id.

Syntax

object.**AI.Channels**(0).Enable [=Boolean]

Data Type

Boolean

AI.Channels(0).Range Property
AI.Channels(1).Range Property
AI.Channels(2).Range Property
AI.Channels(3).Range Property

DAQ-20XX channel-gain that can be set separately for each channel to perform multi-channel/gain analog input, The parameter 0~3 is channel id. This property can setting differential range for each channel.

Syntax

object.**AI.Channels**(0).Range [=Integer]

Settings

Value	Constant	Description
1	AD_B_10_V	Bipolar -10V to +10V
2	AD_B_5_V	Bipolar -5V to +5V
3	AD_B_2_5_V	Bipolar -2.5V to +2.5V
4	AD_B_1_25_V	Bipolar -1.25V to +1.25V
15	AD_U_10_V	Unipolar 0 to +10V
16	AD_U_5_V	Unipolar 0 to +5V
17	AD_U_2_5_V	Unipolar 0 to +2.5V
18	AD_U_1_25_V	Unipolar 0 to +1.25V

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.CHUI Property

The length of the Channel Update interval (that is, the counter value between the initiation of each update sequence).
If the timer base is from *external*, the valid range o f the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of the value is as follows:
Range: 40 through 16777215

Syntax

object.**AO.CHUI** [=Long]

Data Type
Long

AO.DAWRSource Property

Return/Set a value that determines the D/A R/W Source Selection

Syntax
object.**AO.DAWRSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_WRSRC_Int	Internal timer
1	DAQ2K_DA_WRSRC_AFI0	From AFI0 pin
2	DAQ2K_DA_WRSRC_SSI	From SSI source

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerSource Property

Return/Set a value that determines the trigger source selection

Syntax
object.**AO.TriggerSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGSRC_SOFT	Software
1	DAQ2K_DA_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_DA_TRGSRC_ExtD	From external digital trigger pin
3	DAQ2K_DA_TRSRC_SSI	From SSI source

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerMode Property

Return/Set a value that determines the trigger mode selection

Syntax
object.**AO.TriggerMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGMOD_POST	Post Trigger Mode
1	DAQ2K_DA_TRGMOD_DELAY	Delay Trigger Mode

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay1Source Property
Return/Set a va lue that determines the delay1 source selection

Syntax
object.**AO.Delay1Source** [=Integer]

Settings

Value	Constant	Description
64	DAQ2K_DA_Dly1InUI	Delay in samples
0	DAQ2K_DA_Dly1InTimebase	Delay in time base

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay2Source Property
Return/Set a value that determines the delay2 source selection

Syntax
object.**AO.Delay2Source** [=Integer]

Settings

Value	Constant	Description
128	DAQ2K_DA_Dly2InUI	Delay in samples
0	DAQ2K_DA_Dly2InTimebase	Delay in time base

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ExtTrigPolarity Property

Return/Set a value that determines the e xternal digital trigger polarity selection

Syntax
object.**AO.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TrgPositive	Trigger positive edge active
512	DAQ2K_DA_TrgNegative	Trigger negative edge active

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerCount Property

The accepted trigger times in an acquisition. This argument i s only valid for Delay trigger and Post trigger mode.

Syntax
object.**AO.ReTriggerCount** [=Long]

Data Type
Long

AO.Delay1Count Property

The counter value of DLY1 Counter (the delay time after the trigger signal to the start of the waveform generation) .
This argument is only valid for Delay trigger mode.

Syntax
object.**AO.Delay1Count** [=Long]

Data Type
Long

AO.Delay2Count Property

The counter value of DLY2 Counter (the Delay between two consecutive waveform generations).

Syntax
object.**AO.Delay2Count** [=Long]

Data Type
Long

AO.ClockSource Property

The clock source (Time Base) the device selected.

Syntax
object.**AO.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerModeEnable Property

Re-trigger in an acquisition is enabled

Syntax
object.**AO.ClockSource** [=Boolean]

Settings

Value	Constant	Description
0	False	ReTriggerMode is disabled
1	True	ReTriggerMode is enabled

Data Type
Boolean

AO.AIOAnalogTrigCtrl Property

Return/Set a value that determines the t rigger source selection

Syntax

object.**AO.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	CH0ATRIG	AI channel 0
2	CH1ATRIG	AI channel 1
4	CH2ATRIG	AI channel 2
6	CH3ATRIG	AI channel 3
1	EXTATRIG	From external analog trigger pin

Data Type

Integer

AO.AIOTrigCondition Property

Return/Set a value that determines the t rigger condition selection

Syntax

object.**AO.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above-High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks

Please refer to the hard ware manual for the more description.

Data Type

Integer

AO.AIOHLevel Property

The High value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AO.AIOHLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0

Trigger Level digital setting	trigger voltage
0x7F	-0.08V
0x01	-10V

Data Type
Long

AO.AIOLLevel Property
The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax
object.**AO.AIOLLevel** [=Long]

Remarks
***For example*:**
If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AO.DoubleBufferMode Property
Enables or disables double -buffered data acquisition mode.

Syntax
object.**AO.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type
Boolean

AO.Iterations Property
The times of number of the data in the buffer to output to the port. a value of zero is not allowed.

Syntax
object.**AO.Iterations** [=Long]

Data Type
Long

AO.Definite Property
Waveform generation proceeds definite or indefinitely. If double -buffered mode is enabled, this

parameter is of no use.

Syntax
object.**AO.Definite** [=Boolean]

Settings

Value	Constant	Description
0	False	Indefinitely
1	True	Definite

Data Type
Boolean

AO.StopMode Property

Return/Set a value that determines DA transfer termination mode selected.

Syntax
object.**AO. StopMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TerminateImmediate	Software terminate the DA continuous operation immediately
1	DAQ2K_DA_TerminateUC	Software terminate the DA continuous operation on next update counter terminal count
2	DAQ2K_DA_TerminateIC	Software terminate the DA continuous operation on iteration count

Remarks
Please refer to the hardware manual for the more description.

Data Type
Integer

AO.Channels(0).Enable Property
AO.Channels(1).Enable Property

DAQ-2000 output channel that can be set separately for each channel to perform multi-channel analog input, The parameter 0~1 is channel i d.

Syntax
object.**AO.Channels**(0).Enable [=Boolean]

Data Type
Boolean

AO.Channels(0).OutputPolarity Property
AO.Channels(1).OutputPolarity Property

Return/Set a value that determines polarity (unipolar or bipolar) of the output channel.

Syntax

object.**AO.Channels**(0).OutputPolarity [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_UniPolar	Unipolar
1	DAQ2K_DA_BiPolar	Bipolar

Data Type

Integer

AO.Channels(0).IntOrExtref Property
AO.Channels(1).IntOrExtref Property

Return/Set a value th at determines DA reference voltage source of the output channel.

Syntax

object.**AO.Channels**(0). IntOrExtref [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_Int_REF	internal reference
1	DAQ2K_DA_Ext_REF	external reference

Data Type

Integer

AO.Channels(0).RefVoltage Property
AO.Channels(1).RefVoltage Property

If the D/A reference voltage source your device use is internal reference, the valid values is 10.
If the D/A reference voltage source your device use is external reference, the val id range is -10 to +10.

Syntax

object.**AO.Channels**(0).RefVoltage [=Single]

Data Type

Single

AO.Channels(0).Buffer1 Property
AO.Channels(1).Buffer1 Property

This property set up the buffer for continuous analog output operation.
A buffer data or a array of buffer data, data type is integer.

Syntax

object.**AO.Channels**(0).Buffer1 [=Variant]

Remarks

You must assign this property before call StartContAO() method.
This property will be used when single or double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type

Variant (Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer

Dim i As Double

  For i = 0 To 4095
        buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
  Next i
  Daq2005.AO.Channels(0).buffer1 = buffer1
  Daq2005.AO.Channels(0).Enable = True
  Daq2005.AO.StartContAO
```

AO.Channels(0).Buffer2 Property
AO.Channels(1).Buffer2 Property

This property set up the buffer for continuous analog output operation.
A buffer data or a array of buffer data, data type is integer.
This property only available when double buffer mode.

Syntax

object.**AO.Channels**(0).Buffer2 [=Variant]

Remarks

You must assign this property before call StartContAO() method.
This property will be used when double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type

Variant (Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double

  For i = 0 To 4095
        buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
  Next i
```

```
For i = 0 To 4095
    If i < 2048 Then
        buffer2(i) = (&H800 + i Mod 2048) And &HFFF
    Else
        buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
    End If
Next i

Daq2005.AO.Channels(0).buffer1 = buffer1
Daq2005.AO.Channels(0).buffer2 = buffer2
Daq2005.AO.DoubleBufferMode = True
Daq2005.AO.Channels(0).Enable = True
Daq2005.AO.StartContAO
```

SSI.ADCONV Property

Connect / Disconnect a SSI_ADCONV device signal to the specified SSI bus trigger line.

Syntax

object.**SSI.ADCONV** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.ADTRIG Property

Connect / Disconnect a SSI_ADTRIG device signal to the specified SSI bus trigger line.

Syntax

object.**SSI. ADTRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trig ger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.DATRIG Property

Connect / Disconnect a SSI_DATRIG device signal to the specified SSI bus trigger line.

Syntax

object.**SSI.DATRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.DAWR Property
Connect / Disconnect a SSI_DAWR device signal to the specified SSI bus trigger line.

Syntax
object**.SSI.DAWR** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.TIMEBASE Property
Connect / Disconnect a SSI_TIMEBASE device signal to the specified SSI bus trigger line.

Syntax
object**.SSI.TIMEBASE** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

DAQ_2005 Methods

Open Method
Syntax
Function object**.Open** ([ErrMsgBox As Variant]) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
[ErrMsgBox As Variant]

Boolean type.
True: It will popup error message dialog box when operation error.
False: It will fire DAQError event instead of popping up dialog when operation error.

Remarks
This method will be used when the OpenMode property is Manual.

Note
In VC++, *ErrMsgBox* is a VARIANT of VT_I2.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic

Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

ShowPropertyPages Method

This method will show propertypages of ActiveX Control.

Syntax

Function object.***ShowPropertyPages***() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AboutBox Method

This method will show About ADLINK dialog box.

Syntax

Function object.**AboutBox**( ) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DIO.ReadDIPort Method

Syntax

Function object.**DIO.ReadDIPort** (port As Integer, value as Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Returns the digital data read from the specified port. The returned value is 8 -bit data.

Remarks

You can read data from the digital input port.

Note

In VC++, value is a VARIANT of VT_I4.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi

Unit: D2kDask.pas

DIO.ReadDILine Method

Syntax

Function object**.DIO.ReadDILine** (port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

The digital line to be read. The valid value is 0 through 7

value As Variant

Returns the digital logic state, 0 or 1, of the specified line.

Remarks

You can read data from the digital input port.

Note

In VC++, value is a VARIANT of VT_I2.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.WriteDOPort Method

Syntax

Function object**.DIO.WriteDOPort** (port As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value as Variant

8-bit data that will be written to the digital output port.

Remarks
Users can write data to the digital output port.

Note
In VC++, value is a VARIANT of VT_I4.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.WriteDOLine Method

Syntax
Function object**.DIO.WriteDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

Selects one line number from the indicated port: from 0 to 7 (8-bit port).

value As Variant

Sets 0 or 1 to the indicated line.

Note
in VC++, value is a VARIANT of VT_I2.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.ReadBackDOPort Method

Reads back data from the indicated digital output port.

Syntax
Function object**.DIO.ReadBackDOPort** (port As Integer, value As Variant) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Data that is read back from the indicated port.

Note

In VC++, value is a VARIANT of VT_I4.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.ReadBackDOLine Method

Reads back data from the indicated digital output line of the indicated digital output port.

Syntax

Function object.**DIO.ReadBackDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

Selects one line number from the indicated port: from 0 to 7 (8-bit port).

value As Variant

Data that is read back from the indicated line.

Note

In VC++, value is a VARIANT of VT_I2.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.Start Method

GPTC.Counter1.Start Method

Start counter operation with the specified mode.

Syntax

Function object.**GPTC.Counter0.Start**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Remarks

You can start the indicated counter to operate in the specified mode.

GPTC.Counter0.Stop Method
GPTC.Counter1.Stop Method

Stop counter operation.

Syntax

Function object.**GPTC.Counter0.Stop**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

GPTC.Counter0.Reset Method
GPTC.Counter1.Reset Method

Halts the specified general -purpose timer/counter operation and reload the initial value of the timer/counter.

Syntax

Function object.**GPTC.Counter0.Reset**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

GPTC.Counter0.ReadStatus Method
GPTC.Counter1.ReadStatus Method

Reads the latched GPTC status of the general -purpose counter from the GPTC status register.

Syntax

Function object.**GPTC.Counter0.ReadStatus**(Clock *As Integer, Output As Integer, Gate As Integer, Updown As Integer,*) As Boolean

Return Value

True if the function is successful; otherwise False.

AI.StartContAI Method

This method performs continuous A/D conversions on the specified analog input channels at a rate as close to the rate you specified. This method takes advantage of the DAQ-2000 channel-gain that can be set separately for each channel to perform multi-channel/gain analog input. This method will

fire AiComplete or AiHalfReady event depends on AI.Double BufferMode property.

Syntax
Function object.**AI.StartContAI**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Remarks
You can use this method to start the DMA analog input function. If the AI.StreamToFile property is True then the DMA data will be written to the file specified by AI.FileName. Otherwise, the AI.FileName property will be ignored.
The data file is written in binary format, with the lower byte first (little endian). Data type is "Binary codes with miscellaneous data". DAQBench provides a convenient tool DAQCvt to convert the binary file to the file format read easily. See DAQBench User's Guide for the usage of the utility. If you want to handle the data by yourself, please refer to Appendix D Data File Format for the file structure.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example
```
Daq2005.AI.Channels(0).Range = AD_B_10_V
Daq2005.AI.Channels(2).Enable = True
Daq2005.AI.Channels(2).Range = AD_U_1_25_V
Daq2005.AI.Channels(0).Enable = True
Daq2005.AI.StartContAI

Private Sub Daq2005_AiComplete(ScaledData As Variant, BinaryCodes As Variant)
    ' Get Data in    ScaledData
End Sub

Private Sub Daq2005_AiHalfReady(ScaledData As Variant, BinaryCodes As Variant)
    ' Get Data in    ScaledData
End Sub
```

AI.StopContAI Method
You can use this method to force stop DMA analog input.

Syntax
Function object.**AI.StopContAI** () As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

AI.ReadChannels Method
This method performs a software triggered A/D conversion (analog input) on analog input channels and returns the values converted.

Syntax

Function object**.AI.ReadChannels**(Buffer As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Buffer As Variant

An integer array to contain the acquired data. Please refer to Appendix C AD Data Format for the data format in the Buffer.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Daq2005.AI.Channels(0).Range = AD_B_10_V
Daq2005.AI.Channels(2).Enable = True
Daq2005.AI.Channels(2).Range = AD_U_1_25_V
Daq2005.AI.Channels(0).Enable = True

Private Sub Timer1_Timer()
    Dim vBuffer As Variant
    Daq2005.AI.ReadChannels vBuffer
    ' Get Data in    vBuffer
End Sub
```

AO.StartContAO Method

This method performs continuous D/A conversions on the specified analog output channel at a r ate as close to the rate you specified. This method will fire AoComplete or AoBufferReady event depends on AO.DoubleBufferMode property.

Syntax

Function object**.AO.StartContAO**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double
```

```
For i = 0 To 4095
    buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
Next i

For i = 0 To 4095
    If i < 2048 Then
        buffer2(i) = (&H800 + i Mod 2048) And &HFFF
    Else
        buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
    End If
Next i

Daq2005.AO.Channels(0).buffer1 = buffer1
Daq2005.AO.Channels(0).buffer2 = buffer2
Daq2005.AO.DoubleBufferMode = True
Daq2005.AO.Channels(0).Enable = True
Daq2005.AO.StartContAO
```

AO.StopContAO Method

You can use this method to force stop DMA analog output.

Syntax

Function object**.AO.StopContAO** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AO.WriteChannel Method

Accepts a voltage value (or a current v alue), scales it to the proper binary value and writes a binary value to the specified analog output channel.

Syntax

Function object**.AO.WriteChannel**(Channel as Integer, Voltage as Single) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Channel as Integer

The analog output channel number.
Range: 0 or 1 for DAQ -2204

Voltage as Single

The value to be scaled and written to the analog output channel. The range of voltages depends on the type of device, on th e output polarity, And on the voltage reference (external or internal).

CALIBRATION.AutoCalibration Method

Uses this method to calibrate your DAQ -2000 device. When the method is called, the device goes into a self -calibration cycle. The method does not r eturn until the self -calibration is completed.

Syntax

Function object**.CALIBRATION.AutoCalibration** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments
None

CALIBRATION.DisplayErrors Method

Uses this method to fire Ac quireADError and AcquireDAError events. Through those events user can identification DAQ-2000 device current status.

Syntax
Function object**.CALIBRATION.DisplayErrors** () As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Example
Daq2005.CALIBRATION.DisplayErrors

```
Private Sub Daq2005_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "BiPolar"
    End If
    strMsg = "AD channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub

Private Sub Daq2005_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub
```

CALIBRATION.Load Method

Load calibration constants from the spe cified bank of EEPROM.

Syntax
Function object**.CALIBRATION.Load**(BankOfEEPROM As Integer) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the v alue of bank is 0 through 3.

CALIBRATION.Save Method

Save calibration constants to the specified bank of EEPROM.

Syntax

Function object**.CALIBRATION.Save**(BankOfEEPROM As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

SSI.ClearAll Method

Disconnects all of the device signals from the SSI bus trigger lines.

Syntax

Function object**.ClearAll** As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DAQError Event

Syntax

sub ControlName_DAQError (ErrString As String)

Arguments

ErrString As String
The string of error reasion

Remarks

This event will occur when some error occur in control

AiComplete Event

Syntax

sub ControlName_AiComplete(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant
The analog input data array that have been translated to the engineering data (voltage) according to AI.Channels(n).Range property.

BinaryCodes As Variant
The analog input data array with digital format.

Remarks

This event occurs when continuous analog input function is completed. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting.*"BinaryCodes without channel only".*
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AiHalfReady Event

Syntax

sub ControlName_AiHalfReady(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant
The analog input data array that have been translated to the engineering data (voltage) according to AIRange property.

BinaryCodes As Variant
The analog input data array with digital format.

Remarks

This event occurs when one half-buffer of the circular buffer is full at continuous analog input operation. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting. "*BinaryCodes without channel only*"
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AoComplete Event

Syntax
sub ControlName _AoComplete( )

Arguments
None

Remarks
This event occurs when continuous analog output function is completed.

AoBufferReady Event

Syntax
sub ControlName_AoBufferReady(BufferIndex as Integer)

Arguments
BufferIndex as Integer,
The index of the buffer just output.

Remarks
This event occurs when enable the AO.DoubleBufferMode, If User want to dynamic change the output pattern, process it when receive AoBufferReady event.

Example
'This sample code show that how to output four buffers (pattern) in one channel. You can modify this code for dynamic changes pattern.

```
Dim varArray(0 To 3) As Variant
Dim nCounter As Integer

Private Sub AO_Click()

    Dim buffer0(0 To 4095) As Integer
    Dim buffer1(0 To 4095) As Integer
    Dim buffer2(0 To 4095) As Integer
    Dim buffer3(0 To 4095) As Integer

    Dim i As Double

    For i = 0 To 4095
        buffer0(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer1(i) = (&H800 + i Mod 2048) And &HFFF
        Else
            buffer1(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        End If
    Next i
    For i = 0 To 4095
        buffer2(i) = (Cos(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer3(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        Else
            buffer3(i) = (&H800 + i Mod 2048) And &HFFF
        End If
    Next i

    varArray(0) = buffer0
```

```
    varArray(1) = buffer1
    varArray(2) = buffer2
    varArray(3) = buffer3

    Daq2005.AO.Channels(0).buffer1 = varArray(0)
    Daq2005.AO.Channels(0).buffer2 = varArray(1)
    nCounter = 1
    Daq2005.AO.Channels(0).Enable = True

    Daq2005.AO.DoubleBufferMode = True
    Daq2005.AO.StartContAO
End Sub

Private Sub Daq2005_AoBufferReady(ByVal BufferIndex As Integer)

nCounter = nCounter + 1
nCounter = nCounter Mod 4

If BufferIndex = 1 Then
    ' It can change buffer1 in here
    Daq2005.AO.Channels(0).buffer1 = varArray(nCounter)
End If

If BufferIndex = 2 Then
    ' It can change buffer2 in here
    Daq2005.AO.Channels(0).buffer2 = varArray(nCounter)
End If

End Sub
```

AcquireADError Event

Acquires the offset and gain errors of the specified AI channel in the specified polarity mode.

Syntax

sub *ControlName*_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)

Arguments

***channel as Integer*,**

Indicate cha nnel id.

polarity *as Integer,*

Indicate polarity

Value	Description
0	Unipolar
1	Bipolar

gain_err as Double

Indicate gain error

offset_err as Double

Indicate offset error

Example

```
Daq2005.CALIBRATION.DisplayErrors

Private Sub Daq2005_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If
```

```
    strMsg = "AD channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub
```

AcquireDAError Event

Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

Syntax

sub *ControlName*_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)

Arguments

***channel as Integer*,**

Indicate channel id.

polarity *as Integer,*

Indicate polarity

Value	Description
0	Unipolar
1	Bipolar

gain_err as Double

Indicate gain error

offset_err as Double

Indicate offset error

Example

Daq2005.CALIBRATION.DisplayErrors

```
Private Sub Daq2005_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
 List1.AddItem (strMsg)
 List1.Refresh
 End Sub
```

Daq2006 ActiveX Control

The Daq2006 ActiveX control is a software component that provides the interface for users to control PCI-2006 card.

Daq2006 ActiveX Control Overview			
Type (Group)	**Property**	**Method**	**Event**
General	DASKCardType	Open	DAQError
	CardNumber	AboutBox	
	OpenMode	ShowPropertyPages	
	DaskCardID		
DIO	P1ADir	ReadBackDOLine	
	P1BDir	ReadBackDOPort	
	P1CLowerDir	ReadDILine	
	P1CUpperDir	ReadDIPort	
		WriteDOLine	
		WriteDOPort	
GPTC	Mode	Start	
	ClockSource	Stop	
	ClockPolarity	Reset	
	GateSource	ReadStatus	
	GatePolarity		
	UpDownSource		
	UpDownPolarity		
	OutputPolarity		
	IntGATE		
	IntUpDnCTR		
	DelayCount		
	DurationCount		
	OutputValue		
AI	NumOfScan	StartContAI	AiComplete
	Channels(n).Enable	StopContAI	AiHalfReady
	Channels(n).Range	ReadChannels	
	ClockSource		
	ScanInterval		
	ConversionSource		
	TriggerMode		
	TriggerSource		
	DelaySource		
	ReTriggerModeEnable		
	MCounterEnable		
	PostTriggerCount		

Daq2006 ActiveX Control Overview			
	DelayCount		
	MCount		
	ReTriggerCount		
	ExtTrigPolarity		
	ReturnType		
	DoubleBufferMode		
	StreamToFile		
	FileName		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
AO	CHUI	StartContAO	AoComplete
	DAWRSource	StopContAO	AoBufferReady
	TriggerSource	WriteChannel	
	TriggerMode		
	Delay1Source		
	Delay2Source		
	ExtTrigPolarity		
	ReTriggerCount		
	Delay1Count		
	Delay2Count		
	ClockSource		
	ReTriggerModeEnable		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
	DoubleBufferMode		
	Definite		
	StopMode		
	Channels(n).Enable		
	Channels(n).IntOrExtref		
	Channels(n).OutputPolarity		
	Channels(n).RefVoltage		
	Channels(n).Buffer1		
	Channels(n).Buffer2		
CALIBRATION	BankTemperature	AutoCalibration	AcquireADError

Daq2006 ActiveX Control Overview			
	BankDate	Load	AcquireDAError
	CurrentTemperature	Save	
	CurrentDate	DisplayErrors	
SSI	TIMEBASE	ClearAll	
	ADCONV		
	DAWR		
	ADTRIG		
	DATRIG		

DAQ_2006 Properties

DASKCardType Property

Return a value that determines the card type. It is always DAQ_2006 in DAQ-2006 device.

.

Syntax

[Integer] =object.**CardType**

Remarks

DAQ_2006 (for DAQ-2006 Device)

Settings

Value	Constant	Description
6	DAQ_2006	DAQ-2006 Device

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Data Type

Integer

CardNumber Property

The sequence number of the card with ***the same card type*** plugged in the PCI slot.
The card sequence number setting is according to the PCI slot sequence in the mainboard.
The first card (in the most prior slot) is with CardNumber=0. For example, if there are two DAQ-2006 cards plugged on your PC, the DAQ -2006 card in the prior slot should be registered with CardNumber =0, and the other one with CardNumber =1.

.

Syntax

object.**CardNumber** [= short]

Remarks

This property will be used when Initializes the hardw are states of a DAQ -2K data acquisition card.

Data Type

Integer

OpenMode Property

Return/Set a value that determines the mode of opening device.

Syntax

object.**OpenMode** [= short]

Settings

Value	Display	Description
0	Automatic	D2K-OCX will initialize by itself. Automatically open device when the control was created
1	Manual	Call object.**Open**() method to initialized D2K-OCX(open device)
2	Demonstration	D2K-OCX will provide software DAQ data to simulating hardware drivers.

Data Type

Integer

DaskCardID Property

Returns a value that determines the D2K_Register_Card() returns value,
the DaskCardID is a numeric card ID that will be used by other D2K -DASK library functions.

Syntax

[short] =object.**DaskCardID**

Remarks

The range of card id is between 0 and 31.
This property will be used when combine D2K-DASK and D2K-OCX two module in one program.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example:

```
‘ This sample will demonstration If user want to check AI completed by itself.

' OCX to starting continuous AI
  Dim nDaskID as Integer
  Daq2006.Open(TRUE)
  nDaskID = Daq2006.DaskCardID

Daq2006.AI.StartContAI

' Check AI Completed by DASK    API
Dim Stopped    As Byte
Dim AccessCnt As Long

Do While True
    D2K_AI_AsyncCheck nDaskID, Stopped, AccessCnt
    If Stopped = 1 Then
        Exit Do
```

```
    End If
  Loop
  MsgBox "AI Complete"
```

DIO.P1Adir Property

Return/Set a v alue that determines P1A port direction.

Syntax

object.**DIO.P1ADir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1Bdir Property

Return/Set a value that determines P1B port direction.

Syntax

object.**DIO.P1BDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CLowerdir Property

Return/Set a value that determines P1C lower port direction.

Syntax

object.**DIO.P1CLowerDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.

Value	Constant	Description
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CUpperdir Property

Return/Set a value that determines P1C upper port direction.

Syntax

object.**DIO.P1CUpperDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.Mode Property

Return/Set a value that determines the Timer/Counter mode.

Syntax

object.**GPTC.Counter0.Mode** [= Integer]

Settings

Value	Constant	Description
1	SimpleGatedEventCNT	
2	SinglePeriodMSR	
3	SinglePulseWidthMSR	
4	SingleGatedPulseGen	
5	SingleTrigPulseGen	
6	RetrigSinglePulseGen	
7	SingleTrigContPulseGen	
8	ContGatedPulseGen	

Remarks

Please refer to the hardware manual for the mode description.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockSource Property
GPTC.Counter1.ClockSource Property

Return/Set a value that determines the Timer/Counter Source.

Syntax
object.**GPTC.Counter0.ClockSource** [= Integer]
object.**GPTC.Counter1.ClockSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKSRC_INT	internal time base
2	GPTC_CLKSRC_EXT	external time base from GPTC0_SRC or GPTC1_SRC pin

Please refer to the hardware manual for the mode description.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockPolarity Property
GPTC.Counter1.ClockPolarity Property

Return/Set a value that determines the Timer/Counter clock polarity

Syntax
object.**GPTC.Counter0.ClockPolarity** [= Integer]
object.**GPTC.Counter1.ClockPolarity** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKEN_LACTIVE	Low active
2	GPTC_CLKEN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GateSource Property
GPTC.Counter1.GateSource Property

Return/Set a value that determines the Timer/Counter gate source

Syntax

object.**GPTC.Counter0.GateSource** [= Integer]
object.**GPTC.Counter1.GateSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_GATESRC_INT	gate is controlled by software
4	GPTC_GATESRC_EXT	gate is controlled by GPTC0_GATE or GPTC1_GATE pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GatePolarity Property
GPTC.Counter1.GatePolarity Property

Return/Set a value that determines the Timer/Counter gate polarity

Syntax

object.**GPTC.Counter0.GatePolarity** [= Integer]
object.**GPTC.Counter1.GatePolarity** [= Integer]

Settings

Value	Constant	Description
2	GPTC_GATE_LACTIVE	Low active
0	GPTC_GATE_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++

Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownSource Prope rty
GPTC.Counter1.UpDownSource Property

Return/Set a value that determines the Timer/Counter UpDown Source

Syntax

object.**GPTC.Counter0.UpDownSource** [= Integer]
object.**GPTC.Counter1.UpDownSource** [= Integer]

Settings

Value	Constant	Description
0	GPTC_UPDOWN_SEL_INT	Up/Down controlled by software
16	GPTC_UPDOWN_SEL_EXT	Up/Down controlled by GPTC0_UPDOWN orGPTC1_UPDOWN pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownPolarity Property
GPTC.Counter1.UpDownPolarity Property

Return/Set a value that determines the Timer/Counter UpDown Polarity

Syntax

object.**GPTC.Counter0.UpDownPolarity** [= Integer]
object.**GPTC.Counter1.UpDownPolarity** [= Integer]

Settings

Value	Constant	Description
4	GPTC_UPDOWN_LACTIVE	Low active
0	GPTC_UPDOWN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++

Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.OutputPolarity Property
GPTC.Counter1.OutputPolarity Property

Return/Set a value that determines the Timer/Counter Output Polarity

Syntax

object.**GPTC.Counter0.OutputPolarity** [= Integer]
object.**GPTC.Counter1.OutputPolarity** [= Integer]

Settings

Value	Constant	Description
8	GPTC_OUTPUT_LACTIVE	Low active
0	GPTC_OUTPUT_HACTIVE	High active

Please refer to the hardware m anual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.IntGATE Property
GPTC.Counter1.IntGATE Property

Return/Set a value that determines the Timer/Counter Internal gate initialized status

Syntax

object.**GPTC.Counter0.IntGATE** [= Boolean]
object.**GPTC.Counter1.IntGATE** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type

Boolean

GPTC.Counter0.IntUpD nCTR Property
GPTC.Counter1.IntUpDnCTR Property

Return/Set a value that determines the Timer/Counter internal updown counter initialized status.

Syntax

object.**GPTC.Counter0.IntUpDnCTR** [= Boolean]
object.**GPTC.Counter1.IntUpDnCTR** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type

Boolean

GPTC.Counter0.DelayCount Property
GPTC.Counter1.DelayCount Property

Return/Set a value that determines the Timer/Counter internal initial count of the GPTC or pulse delay. The counter value of load register 1 of timer/counter. The meaning for the value depends on the mode the timer /counter performs. For mode 1 to mode 3, the value is the initial count of the GPTC. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse delay.

Syntax

object.**GPTC.Counter0.DelayCount** [= Integer]
object.**GPTC.Counter1.DelayCount** [= Integer]

Data Type

Integer

GPTC.Counter0.DurationCount Property
GPTC.Counter1.DurationCount Property

Return/Set a value that determines the Timer/Counter pulse width when mode 4 to mode 8.
The counter value of load register 2 of timer/counter. For mode 1 to mode 3, the value is not used. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse width.

Syntax

object.**GPTC.Counter0.DurationCount** [= Integer]
object.**GPTC.Counter1.DurationCount** [= Integer]

Data Type

Integer

GPTC.Counter0.OutputValue Property
GPTC.Counter1.OutputValue Property

Returns the counter value of the specified general -purpose timer/cou nter.
Range: 0 through 65535

Syntax

[Integer=] object.**GPTC.Counter0.OutputValue**
[Integer=] object.**GPTC.Counter1.OutputValue**

Data Type
Integer

GPTC.CALIBRATION.BankTemperature Property

Returns a value that since user's last calibrated temperatu re in the EEPRON Bank .

Syntax
object.**CALIBRATION.BankTemperature**(*[BankOfEEPROM as Integer]*) As Single

Data Type
Single

GPTC.CALIBRATION.BankDate Property

Returns a value that since user's last calibrated date in the EEPRON Bank .

Syntax
object**.CALIBRATION.BankDate**(*[BankOfEEPROM as Integer]*) As String

Data Type
String

GPTC.CALIBRATION.CurrentTemperature Property

Returns a value that determines the current temperature on card.

Syntax
[Single=] object**.CALIBRATION.CurrentTemperature**

Data Type
Single

GPTC.CALIBRATION.CurrentDate Property

Returns a value that determines the current date.

Syntax
[String=] object**.CALIBRATION.CurrentDate**

Data Type
String

AI.NumOfScan Property

If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, NumOf *Scan* is the size (in samples) allocated for each channel in the circular buffer. This value must be a multiple of 2.

Syntax
object**.AI.NumOfScan** [=Long]

Remarks
Non-double-buffer mode
This value multiply the total number of scan channels is the total number of A/D conversions to be

performed.

Double-buffer-mode

This value multiply the total number of scan channels is the size (in sample) of the circular buffer.

Data Type

Long

AI.ClockSource Property

The clock source (Time Base) the device selected.

Syntax

object.**AI.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ScanInterval Property

The length of the scan interval (that is, the count er value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of the value is 160 through 16777215

Syntax

object.**AI.ScanInterval** [=Long]

Data Type

Long

AI.ConversionSource Property

The A/D Conversion Source Selection.

Syntax

object.**AI.ConversionSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_ADCONVSRC_Int	Internal timer
4	DAQ2K_AI_ADCONVSRC_AFI0	From AFI0 pin

Value	Constant	Description
8	DAQ2K_AI_ADCONVSRC_SSI	From SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerMode Property
The Trigger Mode Selection.

Syntax
object**.AI.TriggerMode** [=Short]

Settings
Value	Constant	Description
0	DAQ2K_AI_TRGMOD_POST	Post Trigger Mode
8	DAQ2K_AI_TRGMOD_DELAY	Delay Trigger Mode
16	DAQ2K_AI_TRGMOD_PRE	Pre-Trigger Mode
24	DAQ2K_AI_TRGMOD_MIDL	Middle-Trigger Mode

Remarks
Please refer to the hardware manual for the trigger mode description.

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerSource Property
The Trigger Source Selection.

Syntax
object**.AI.TriggerSource** [=Short]

Settings
Value	Constant	Description
0	DAQ2K_AI_TRGSRC_SOFT	Software
1	DAQ2K_AI_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_AI_TRGSRC_ExtD:	From external digital trigger pin
3	DAQ2K_AI_TRSRC_SSI	From SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.DelaySource Prope rty
The delay source selection.

Syntax
object.**AI. DelaySource** [=Short]

Settings

Value	Constant	Description
256	DAQ2K_AI_Dly1InSamples	Delay in samples
0	DAQ2K_AI_Dly1InTimebase	Delay in time base

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReTriggerModeEnable Property
Re-trigger in an acquisition is enabled

Syntax
object.**AI.ReTriggerModeEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	ReTriggerMode is disabled.
1	True	ReTriggerMode is enabled.

Data Type
Boolean

AI.MCounterEnable Property
This constant is only valid for Pre -trigger and Middle trigger mode

Mcounter is enabled and then the trigger signal is ignore before M termin al count is reached.

Syntax
object.**AI.MCounterEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	MCounter is disabled.
1	True	Mcounter is enabled.

Data Type
Boolean

AI.PostTriggerCount Property

This constant is only valid for Middl e trigger mode,
The PostTriggerCount indicates the number of data will be accessed after a specific trigger event.

Syntax
object.**AI.PostTriggerCount** [=Long]

Data Type
Long

AI.DelayCount Property

This constant is only valid for Delay trigger mod e,
The DelayCount indicates the number of data or timer ticks will be ignored after a specific trigger event.

Syntax
object.**AI.DelayCount** [=Long]

Data Type
Long

AI.MCount Property

The counter value of MCounter . This argument is only valid for Pretrigger and Middle trigger mode.

Syntax
object.**AI.MCount** [=Long]

Data Type
Long

AI.ReTriggerCount Property

The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax
object.**AI.ReTriggerCount** [=Long]

Data Type
Long

AI.ExtTrigPolarity Property
External Digital **Trigger Polarity.**

Syntax
object.**AI.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_AI_TrgPositive	Trigger positive edge active
4096	DAQ2K_AI_TrgNegative	Trigger negative edge active

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReturnType Property
Return/Set a value that determines the return data ty pe of analog input when AiComplete or AiHalfReady event would occur.

Syntax
object.**AI.ReturnType** [=Integer]

Settings

Value	Constant	Description
0		Scaled data only
1		Binary codes without channel only
2		Scaled data and binary codes without channel
3		No data return

Data Type
Integer

AI.DoubleBufferMode Property
Enables or disables double -buffered data acquisition mode.

Syntax
object.**AI.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type
Boolean

AI.StreamToFile Property

Return/Set a value that determines if the control is enabled the function of streaming data to disk file. This argument is only valid for Delay trigger

Syntax
object.**AI.StreamToFile** [=Boolean]

Settings

Value	Constant	Description
0	False	Disable the function of streaming data to disk file.
1	True	Enable the function of streaming data to disk file

Data Type
Boolean

AI.FileName Property

FileName specified the f ile name of streaming data to disk. This argument is only valid for AI.StreamToFile is Enable.

Syntax
object.**AI.FileName** [=String]

Data Type
String

AI.AIOAnalogTrigCtrl Property

Return/Set a value that determines the t rigger source selection

Syntax
object.**AI.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	CH0ATRIG	AI channel 0
2	CH1ATRIG	AI channel 1
4	CH2ATRIG	AI channel 2
6	CH3ATRIG	AI channel 3
1	EXTATRIG	From external analog trigger pin

Data Type
Integer

AI.AIOTrigCondition Property

Return/Set a value that determines the t rigger condition selection

Syntax
object.**AI.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above-High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks

Please refer to the hardware manual for the more description.

Data Type

Integer

AI.AIOHLevel Property

The High value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AI.AIOHLevel** [=Long]

Remarks

***For example*:**

If the trigger volt age is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AI.AIOLLevel Property

The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AI.AIOLLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	Trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AI.Channels(0).Enable Prope rty
AI.Channels(1).Enable Property
AI.Channels(2).Enable Property
AI.Channels(3).Enable Property

DAQ-2000 channel-gain that can be set separately for each channel to perform multi-channel/gain analog input, The parameter 0~3 is channel id.

Syntax
object.**AI.Channels**(0).Enable [=Boolean]

Data Type
Boolean

AI.Channels(0).Range Property
AI.Channels(1).Range Property
AI.Channels(2).Range Property
AI.Channels(3).Range Property

DAQ-20XX channel-gain that can be set separately for each channel to perform multi-channel/gain analog input, The parameter 0~3 is channel id. This property can setting differential range for each channel.

Syntax
object.**AI.Channels**(0).Range [=Integer]

Settings

Value	Constant	Description
1	AD_B_10_V	Bipolar -10V to +10V
2	AD_B_5_V	Bipolar -5V to +5V
3	AD_B_2_5_V	Bipolar -2.5V to +2.5V
4	AD_B_1_25_V	Bipolar -1.25V to +1.25V
15	AD_U_10_V	Unipolar 0 to +10V
16	AD_U_5_V	Unipolar 0 to +5V
17	AD_U_2_5_V	Unipolar 0 to +2.5V
18	AD_U_1_25_V	Unipolar 0 to +1.25V

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.CHUI Property

The length of the Channel Update interval (that is, the counter value between the initiation of each update sequence).
If the timer base is from *external*, the valid range of the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of the value is as follows:
Range: 40 through 16777215

Syntax

object.**AO.CHUI** [=Long]

Data Type

Long

AO.DAWRSource Property

Return/Set a value that determines the D/A R/W Source Selection

Syntax

object.**AO.DAWRSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_WRSRC_Int	Internal timer
1	DAQ2K_DA_WRSRC_AFI0	From AFI0 pin
2	DAQ2K_DA_WRSRC_SSI	From SSI source

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerSource Property

Return/Set a value that determines the trigger source selection

Syntax

object.**AO.TriggerSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGSRC_SOFT	Software
1	DAQ2K_DA_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_DA_TRGSRC_ExtD	From external digital trigger pin
3	DAQ2K_DA_TRSRC_SSI	From SSI source

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerMode Property

Return/Set a value that determines the trigger mode selection

Syntax

object.**AO.TriggerMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGMOD_POST	Post Trigger Mode
1	DAQ2K_DA_TRGMOD_DELAY	Delay Trigger Mode

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay1Source Property

Return/Set a value that determines the delay1 source selection

Syntax

object.**AO.Delay1Source** [=Integer]

Settings

Value	Constant	Description
64	DAQ2K_DA_Dly1InUI	Delay in samples
0	DAQ2K_DA_Dly1InTimebase	Delay in time base

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay2Source Property

Return/Set a value that determines the delay2 source selection

Syntax

object.**AO.Delay2Source** [=Integer]

Settings

Value	Constant	Description
128	DAQ2K_DA_Dly2InUI	Delay in samples
0	DAQ2K_DA_Dly2InTimebase	Delay in time base

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ExtTrigPolarity Property

Return/Set a value that determines the e xternal digital trigger polarity selection

Syntax

object.**AO.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TrgPositive	Trigger positive edge active
512	DAQ2K_DA_TrgNegative	Trigger negative edge active

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerCount Property

The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax

object.**AO.ReTriggerCount** [=Long]

Data Type
Long

AO.Delay1Count Property

The counter value of DLY1 Counter (the delay time after the trig ger signal to the start of the waveform generation) . This argument is only valid for Delay trigger mode.

Syntax
object.**AO.Delay1Count** [=Long]

Data Type
Long

AO.Delay2Count Property

The counter value of DLY2 Counter (the Delay between two consecutive waveform generations).

Syntax
object.**AO.Delay2Count** [=Long]

Data Type
Long

AO.ClockSource Property

The clock source (Time Base) the device selected.

Syntax
object.**AO.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerModeEnable Property

Re-trigger in an acquisition is enabled

Syntax
object.**AO.ClockSource** [=Boolean]

Settings

Value	Constant	Description
0	False	ReTriggerMode is disabled
1	True	ReTriggerMode is enabled

Data Type

Boolean

AO.AIOAnalogTrigCtrl Property

Return/Set a value that determines the t rigger source selection

Syntax

object**.AO.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	CH0ATRIG	AI channel 0
2	CH1ATRIG	AI channel 1
4	CH2ATRIG	AI channel 2
6	CH3ATRIG	AI channel 3
1	EXTATRIG	From external analog trigger pin

Data Type

Integer

AO.AIOTrigCondition Property

Return/Set a value that determines the t rigger condition selection

Syntax

object**.AO.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above-High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks

Please refer to the hardware manual for the more description.

Data Type

Integer

AO.AIOHLevel Property

The High value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object**.AO.AIOHLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AO.AIOLLevel Property

The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AO.AIOLLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	Trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AO.DoubleBufferMode Property

Enables or disables double -buffered data acquisition mode.

Syntax

object.**AO.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type

Boolean

AO.Iterations Property

The times of number of the data in the buffer to output to the port. a value of zero is not allowed.

Syntax
object.**AO.Iterations** [=Long]

Data Type
Long

AO.Definite Property
Waveform generation proceeds definite or indefinitely. If double -buffered mode is enabled, this parameter is of no use.

Syntax
object.**AO.Definite** [=Boolean]

Settings

Value	Constant	Description
0	False	Indefinitely
1	True	Definite

Data Type
Boolean

AO.StopMode Property
Return/Set a value that determines DA transfer termination mode selected.

Syntax
object.**AO. StopMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TerminateImmediate	Software terminate the DA continuous operation immediately
1	DAQ2K_DA_TerminateUC	Software terminate the DA continuous operation on next update counter terminal count
2	DAQ2K_DA_TerminateIC	Software terminate the DA continuous operation on iteration count

Remarks
Please refer to the hardware manual for the more description.

Data Type
Integer

AO.Channels(0).Enable Property
AO.Channels(1).Enable Property

DAQ-2000 output channel that can be set separately for each channel to perform multi-channel analog input, The parameter 0~1 is channel id.

Syntax

object.**AO.Channels**(0).Enable [=Boolean]

Data Type

Boolean

AO.Channels(0).OutputPolarity Property
AO.Channels(1).OutputPolarity Property

Return/Set a value that determines polarity (unipol ar or bipolar) of the output channel.

Syntax

object.**AO.Channels**(0).OutputPolarity [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_UniPolar	Unipolar
1	DAQ2K_DA_BiPolar	Bipolar

Data Type

Integer

AO.Channels(0).IntOrExtref Property
AO.Channels(1).IntOrExtref Property

Return/Set a value that determines DA reference voltage source of the output channel.

Syntax

object.**AO.Channels**(0). IntOrExtref [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_Int_REF	internal reference
1	DAQ2K_DA_Ext_REF	external reference

Data Type

Integer

AO.Channels(0).RefVoltage Property
AO.Channels(1).RefVoltage Property

If the D/A reference voltage source your device use is internal reference, the valid values is 10.
If the D/A reference voltage so urce your device use is external reference, the valid range is -10 to +10.

Syntax

object.**AO.Channels**(0).RefVoltage [=Single]

Data Type
Single

AO.Channels(0).Buffer1 Property
AO.Channels(1).Buffer1 Property

This property set up the buffer for conti nuous analog output operation. A buffer data or a array of buffer data, data type is integer.

Syntax
object**.AO.Channels**(0).Buffer1 [=Variant]

Remarks
You must assign this property before call StartContAO() method. This property will be used when single or double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type
Variant (Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i
 Daq2006.AO.Channels(0).buffer1 = buffer1
 Daq2006.AO.Channels(0).Enable = True
 Daq2006.AO.StartContAO
```

AO.Channels(0).Buffer2 Property
AO.Channels(1).Buffer2 Property

This property set up the buffer for continuous analog output operation. A buffer data or a array of buffer data, data type is integer. This property only available when double buffer mode.

Syntax
object**.AO.Channels**(0).Buffer2 [=Variant]

Remarks
You must assign this property before call StartContAO() method.
This property will be used when double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type
Variant(Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i

 For i = 0 To 4095
      If i < 2048 Then
           buffer2(i) = (&H800 + i Mod 2048) And &HFFF
      Else
           buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
      End If
 Next i

 Daq2006.AO.Channels(0).buffer1 = buffer1
 Daq2006.AO.Channels(0).buffer2 = buffer2
 Daq2006.AO.DoubleBufferMode = True
 Daq2006.AO.Channels(0).Enable = True
 Daq2006.AO.StartContAO
```

SSI.ADCONV Property
Connect / Disconnect a SSI_ADCONV device signal to the specified SSI bus trigger line.

Syntax
object.**SSI.ADCONV** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.ADTRIG Property
Connect / Disconnect a SSI_ADTRIG device signal to the specified SSI bus trigger line.

Syntax
object.**SSI. ADTRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.DATRIG Property

Connect / Disconnect a SSI_DATRIG device signal to the specified SSI bus trigger line.

Syntax
object.**SSI.DATRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.DAWR Property

Connect / Disconnect a SSI_DAWR device signal to the specified SSI bus trigger line.

Syntax
object.**SSI.DAWR** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.TIMEBASE Property

Connect / Disconnect a SSI_TIMEBASE device signal to the specified SSI bus trigger line.

Syntax
object.**SSI.TIMEBASE** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

DAQ_2006 Methods

Open Method

Syntax
Function object.**Open** ([ErrMsgBox As Variant]) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments

[ErrMsgBox As Variant]

Boolean type.
True: It will popup error message dialog box when operation error.
False: It will fire DAQError event instead of popping up dialog when operation error.

Remarks

This method will be used when the OpenMode property is Manual.

Note

In VC++, *ErrMsgBox* is a VARIANT of VT_I2.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

ShowPropertyPages Method

This method will show propertypages of ActiveX Control.

Syntax

Function object.***ShowPropertyPages***() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AboutBox Method

This method will show About ADLINK dialog box.

Syntax

Function object.**AboutBox**( ) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DIO.ReadDIPort Method

Syntax

Function object.**DIO.ReadDIPort** (port As Integer, value as Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Returns the digital data read from the specified port. The returned value is 8 -bit data.

Remarks
You can read data from the digital input port.

Note
In VC++, value is a VARIANT of VT_I4.

DIO.ReadDILine Method

Syntax
Function object.**DIO.ReadDILine** (port As Integer, line As Integer, value As Variant) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

The digital line to be read. The valid value is 0 through 7

value As Variant

Returns the digital logic state, 0 or 1, of the specified line.

Remarks
You can read data from the digital input port.

Note
In VC++, value is a VARIANT of VT_I2.

DIO.WriteDOPort Method

Syntax
Function object.**DIO.WriteDOPort** (port As Integer, value As Variant) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value as Variant

8-bit data that will be written to the digital output port.

Remarks
Users can write data to the digital output port.

Note
In VC++, value is a VARIANT of VT_I4.

DIO.WriteDOLine Method

Syntax

Function object.**DIO.WriteDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

Selects one line number from the indicated port: from 0 to 7 (8-bit port).

value As Variant

Sets 0 or 1 to the indicated line.

Note

In VC++, value is a VARIANT of VT_I2.

DIO.ReadBackDOPort Method

Reads back data from the indicated digital output port.

Syntax

Function object.**DIO.ReadBackDOPort** (port As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Data that is read back from the indicated port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.ReadBackDOLine Method

Reads back data from the indicated digital output line of the indicated digital output port.

Syntax

Function object.**DIO.ReadBackDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer
Selects one line number from the indicated port: from 0 to 7 (8-bit port).
value As Variant
Data that is read back from the indicated line.

Note
In VC++, value is a VARIANT of VT_I2.

GPTC.Counter0.Start Method
GPTC.Counter1.Start Method
Start counter operation with the specified mode.

Syntax
Function object**.GPTC.Counter0.Start**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Remarks
You can start the indicated counter to operate in the specified mode.

GPTC.Counter0.Stop Method
GPTC.Counter1.Stop Method
Stop counter operation.

Syntax
Function object**.GPTC.Counter0.Stop**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

GPTC.Counter0.Reset Method
GPTC.Counter1.Reset Method
Halts the specified general -purpose timer/counter operation and reload the initial value of the timer/counter.

Syntax
Function object**.GPTC.Counter0.Reset**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

GPTC.Counter0.ReadStatus Method
GPTC.Counter1.ReadStatus Method

Reads the latched GPTC status of the general -purpose counter from the GPTC status register.

Syntax

Function object.**GPTC.Counter0.ReadStatus**(Clock *As Integer, Output As Integer, Gate As Integer, Updown As Integer,*) As Boolean

Return Value

True if the function is successful; otherwise False.

AI.StartContAI Method

This method performs continuous A/D conversions on the specified analog input channels at a rate as close to the rate you specified. This method takes advantage of the DAQ-2000 channel-gain that can be set separately for each channel to perform multi-channel/gain analog input. This method will fire AiComplete or AiHalfReady event depends on AI.DoubleBufferMode property.

Syntax

Function object.**AI.StartContAI**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Remarks

You can use this method to start the DMA analog input function. If the AI.StreamToFile property is True then the DMA data will be written to the file specified by AI.FileName. Otherwise, the AI.FileName property will be ignored. The data file is written in binary format, with the lower byte first (little endian). Data type is “Binary codes with miscellaneous data”. DAQBench provides a convenient tool DAQCvt to convert the binary file to the file format read easily. See DAQBench User’s Guide for the usage of the utility. If you want to handle the data by yourself, please refer to Appendix D Data File Format for the file structure.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Daq2006.AI.Channels(0).Range = AD_B_10_V
Daq2006.AI.Channels(2).Enable = True
Daq2006.AI.Channels(2).Range = AD_U_1_25_V
Daq2006.AI.Channels(0).Enable = True
Daq2006.AI.StartContAI

Private Sub Daq2006_AiComplete(ScaledData As Variant, BinaryCodes As Variant)
    ' Get Data in    ScaledData
End Sub

Private Sub Daq2006_AiHalfReady(ScaledData As Variant, BinaryCodes As Variant)
```

```
        ' Get Data in    ScaledData
End Sub
```

AI.StopContAI Method

You can use this method to force stop DMA analog input.

Syntax

Function object.**AI.StopContAI** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AI.ReadChannels Method

This method performs a software triggered A/D conversion (analog input) on analog input channels and returns the values converted.

Syntax

Function object.**AI.ReadChannels**(Buffer As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Buffer As Variant

An integer array to contain the acquired data. Please refer to Appendix C AD Data Format for the data format in the Buffer.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
    Daq2006.AI.Channels(0).Range = AD_B_10_V
    Daq2006.AI.Channels(2).Enable = True
    Daq2006.AI.Channels(2).Range = AD_U_1_25_V
    Daq2006.AI.Channels(0).Enable = True

Private Sub Timer1_Timer()
    Dim vBuffer As Variant
    Daq2006.AI.ReadChannels vBuffer
    ' Get Data in    vBuffer
End Sub
```

AO.StartContAO Method

This method performs continuous D/A conversions on the specified analog output channel at a rate as close to the rate you specified. This method will fire AoComplete or AoBufferReady event depends on AO.DoubleBufferMode property.

Syntax

Function object.**AO.StartContAO**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example
```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i

 For i = 0 To 4095
      If i < 2048 Then
           buffer2(i) = (&H800 + i Mod 2048) And &HFFF
      Else
           buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
      End If
 Next i

 Daq2006.AO.Channels(0).buffer1 = buffer1
 Daq2006.AO.Channels(0).buffer2 = buffer2
 Daq2006.AO.DoubleBufferMode = True
 Daq2006.AO.Channels(0).Enable = True
 Daq2006.AO.StartContAO
```

AO.StopContAO Method
You can use this method to force stop DMA analog output.

Syntax
Function object**.AO.StopContAO** () As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

AO.WriteChannel Method
Accepts a voltage value (or a current value), scales it to the proper binary value and writes a binary value to the specified analog output channel.

Syntax
Function object**.AO.WriteChannel**(Channel as Integer, Voltage as Single) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
Channel as Integer
The analog output channel number.
Range: 0 or 1 for DAQ -2204

Voltage as Single
The value to be scaled and written to the analog output channel. The range of voltages depends on the type of device, on the output polarity, and on the voltage reference (external or internal).

CALIBRATION.AutoCalibration Method
Uses this method to calibrate your DAQ -2000 device. When the method is called, the device goes into a self-calibration cycle. The method does not return until the self -calibration is completed.

Syntax
Function object**.CALIBRATION.AutoCalibration** () As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

CALIBRATION.DisplayErrors Method
Uses this method to fire AcquireADError and AcquireDAError events. Through those events user can identification DAQ-2000 device current status.

Syntax
Function object**.CALIBRATION.DisplayErrors** () As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Example

```
Daq2006.CALIBRATION.DisplayErrors

Private Sub Daq2006_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "BiPolar"
    End If
    strMsg = "AD channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub

Private Sub Daq2006_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
```

```
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub
```

CALIBRATION.Load Method

Load calibration constants from the specified bank of EEPROM.

Syntax

Function object**.CALIBRATION.Load**(BankOfEEPROM As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

CALIBRATION.Save Method

Save calibration constants to the specified bank of EEPROM.

Syntax

Function object**.CALIBRATION.Save**(BankOfEEPROM As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

SSI.ClearAll Method

Disconnects all of the device signals from the SSI bus trigger lines.

Syntax

Function object**.ClearAll** As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DAQError Event

Syntax

sub ControlName_DAQError (ErrString As String)

Arguments

ErrString As String
The string of error reasion

Remarks

This event will occur when some error occur in control

AiComplete Event

Syntax

sub ControlName_AiComplete(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant

The analog input data array that have been translated to the engineering data (voltage) according to AI.Channels(n).Range property.

BinaryCodes As Variant

The analog input data array with digital format.

Remarks

This event occurs when continuous analog input function is completed. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting.*"BinaryCodes without channel only".*
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AiHalfReady Event

Syntax

sub ControlName_AiHalfReady(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant

The analog input data array that have been translated to the engineering data (voltage) according to AIRange property.

BinaryCodes As Variant

The analog input data array with digital format.

Remarks

This event occurs when one half-buffer of the circular buffer is full at continuous analog input operation. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting. "*BinaryCodes without channel only*"
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AoComplete Event

Syntax
sub ControlName _AoComplete( )

Arguments
None

Remarks
This event occurs when continuous ana log output function is completed.

AoBufferReady Event

Syntax
sub ControlName_AoBufferReady(BufferIndex as Integer)

Arguments
BufferIndex as Integer, The index of the buffer just output.

Remarks
This event occurs when enable the AO.DoubleBufferMode, If User want to dynamic change the output pattern, process it when receive AoBufferReady event.

Example

```
'This sample code show that how to output four buffers (pattern) in one channel.
'You can modify this code for dynamic changes pattern.

Dim varArray(0 To 3) As Variant
Dim nCounter As Integer

Private Sub AO_Click()

    Dim buffer0(0 To 4095) As Integer
    Dim buffer1(0 To 4095) As Integer
    Dim buffer2(0 To 4095) As Integer
    Dim buffer3(0 To 4095) As Integer

    Dim i As Double

    For i = 0 To 4095
        buffer0(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer1(i) = (&H800 + i Mod 2048) And &HFFF
        Else
            buffer1(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        End If
    Next i
    For i = 0 To 4095
        buffer2(i) = (Cos(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer3(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        Else
            buffer3(i) = (&H800 + i Mod 2048) And &HFFF
        End If
    Next i

    varArray(0) = buffer0
```

```
    varArray(1) = buffer1
    varArray(2) = buffer2
    varArray(3) = buffer3

    Daq2006.AO.Channels(0).buffer1 = varArray(0)
    Daq2006.AO.Channels(0).buffer2 = varArray(1)
    nCounter = 1
    Daq2006.AO.Channels(0).Enable = True

    Daq2006.AO.DoubleBufferMode = True
    Daq2006.AO.StartContAO
End Sub

Private Sub Daq2006_AoBufferReady(ByVal BufferIndex As Integer)

nCounter = nCounter + 1
nCounter = nCounter Mod 4

If BufferIndex = 1 Then
    ' It can change buffer1 in here
    Daq2006.AO.Channels(0).buffer1 = varArray(nCounter)
End If

If BufferIndex = 2 Then
    ' It can change buffer2 in here
    Daq2006.AO.Channels(0).buffer2 = varArray(nCounter)
End If

End Sub
```

AcquireADError Event

Acquires the offset and gain errors of the specified AI channel in the specified polarity mode.

Syntax

sub *ControlName*_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)

Arguments

***channel as Integer*,**

Indicate channel id.

polarity *as Integer,*

Indicate polarity

Value	Description
0	Unipolar
1	Bipolar

gain_err as Double

Indicate gain error

offset_err as Double

Indicate offset error

Example

```
Daq2006.CALIBRATION.DisplayErrors

Private Sub Daq2006_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
```

```
    End If
    strMsg = "AD channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub
```

AcquireDAError Event

Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

Syntax

sub *ControlName*_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal off set_err As Double)

Arguments

***channel as Integer*,**

Indicate channel id.

polarity *as Integer,*

Indicate polarity

Value	**Description**
0	Unipolar
1	Bipolar

gain_err as Double

Indicate gain error

offset_err as Double

Indicate offset error

Example

Daq2006.CALIBRATION.DisplayErrors

```
Private Sub Daq2006_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
 List1.AddItem (strMsg)
 List1.Refresh
 End Sub
```

Daq2010 ActiveX Control

The Daq2010 ActiveX control is a software component that provides the interface for users to control PCI-2010 card.

Daq2010 ActiveX Control Overview			
Type (Group)	**Property**	**Method**	**Event**
General	DASKCardType	Open	DAQError
	CardNumber	AboutBox	
	OpenMode	ShowPropertyPages	
	DaskCardID		
DIO	P1ADir	ReadBackDOLine	
	P1BDir	ReadBackDOPort	
	P1CLowerDir	ReadDILine	
	P1CUpperDir	ReadDIPort	
		WriteDOLine	
		WriteDOPort	
GPTC	Mode	Start	
	ClockSource	Stop	
	ClockPolarity	Reset	
	GateSource	ReadStatus	
	GatePolarity		
	UpDownSource		
	UpDownPolarity		
	OutputPolarity		
	IntGATE		
	IntUpDnCTR		
	DelayCount		
	DurationCount		
	OutputValue		
AI	NumOfScan	StartContAI	AiComplete
	Channels(n).Enable	StopContAI	AiHalfReady
	Channels(n).Range	ReadChannels	
	ClockSource		
	ScanInterval		
	ConversionSource		
	TriggerMode		
	TriggerSource		
	DelaySource		
	ReTriggerModeEnable		
	MCounterEnable		
	PostTriggerCount		

Daq2010 ActiveX Control Overview			
	DelayCount		
	MCount		
	ReTriggerCount		
	ExtTrigPolarity		
	ReturnType		
	DoubleBufferMode		
	StreamToFile		
	FileName		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
AO	CHUI	StartContAO	AoComplete
	DAWRSource	StopContAO	AoBufferReady
	TriggerSource	WriteChannel	
	TriggerMode		
	Delay1Source		
	Delay2Source		
	ExtTrigPolarity		
	ReTriggerCount		
	Delay1Count		
	Delay2Count		
	ClockSource		
	ReTriggerModeEnable		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
	DoubleBufferMode		
	Definite		
	StopMode		
	Channels(n).Enable		
	Channels(n).IntOrExtref		
	Channels(n).OutputPolarity		
	Channels(n).RefVoltage		
	Channels(n).Buffer1		
	Channels(n).Buffer2		
CALIBRATION	BankTemperature	AutoCalibration	AcquireADError

Daq2010 ActiveX Control Overview			
	BankDate	Load	AcquireDAError
	CurrentTemperature	Save	
	CurrentDate	DisplayErrors	
SSI	TIMEBASE	ClearAll	
	ADCONV		
	DAWR		
	ADTRIG		
	DATRIG		

DAQ_2010 Properties

DASKCardType Property

Return a value that determines the card type. It is always DAQ_2010 in DAQ-2010 device.

.

Syntax

[Integer] =object.**CardType**

Remarks

DAQ_2010 (for DAQ-2010 Device)

Settings

Value	constant	Description
1	DAQ_2010	For DAQ-2010 Device

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Data Type

Integer

CardNumber Property

The sequence number of the card with ***the same card type*** plugged in the PCI slot.
The card sequence number setting is according to the PCI slot sequence in th e mainboard.
The first card (in the most prior slot) is with CardNumber=0. For example, if there are two DAQ - 2010 cards plugged on your PC, the DAQ -2010 card in the prior slot should be registered with CardNumber =0, and the other one with CardNumber =1.

.

Syntax

object.**CardNumber** [= short]

Remarks

This property will be used when Initializes the hardware states of a DAQ -2K data acquisition card.

Data Type

Integer

OpenMode Property

Return/Set a value that determines the mode of opening device.

Syntax

object.**OpenMode** [= short]

Settings

Value	Display	Description
0	Automatic	D2K-OCX will initialize by itself. Automatically open device when the control was created
1	Manual	Call object.**Open**() method to initialized D2K-OCX(open device)
2	Demonstration	D2K-OCX will provide software DAQ data to simulating hardware drivers.

Data Type

Integer

DaskCardID Property

Returns a value that determines the D2K_Register_Card() returns value,
the DaskCardID is a numeric card ID that will be used by other D2K -DASK library functions.

Syntax

[short] =object.**DaskCardID**

Remarks

The range of card id is between 0 and 31.
This property will be used when combine D2K-DASK and D2K-OCX two module in one program.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

This sample will demonstration If user want to check AI completed by itself.

```
' OCX to starting continuous AI
  Dim nDaskID as Integer
  Daq2010.Open(TRUE)
  nDaskID = Daq2010.DaskCardID

 Daq2010.AI.StartContAI

' Check AI Completed by DASK  API
Dim Stopped  As Byte
Dim AccessCnt As Long

Do While True
    D2K_AI_AsyncCheck nDaskID, Stopped, AccessCnt
    If Stopped = 1 Then
```

```
            Exit Do
        End If
    Loop
    MsgBox "AI Complete"
```

DIO.P1Adir Property

Return/Set a value that determines P1A port direction.

Syntax

object.**DIO.P1ADir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1Bdir Property

Return/Set a value that determines P1B port direction.

Syntax

object.**DIO.P1BDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CLowerdir Property

Return/Set a value that determines P1C lower port direction.

Syntax

object.**DIO.P1CLowerDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CUpperdir Property
Return/Set a value that determines P1C upper port direction.

Syntax
object.**DIO.P1CUpperDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.Mode Property
Return/Set a value that determines the Timer/Counter mode.

Syntax
object.**GPTC.Counter0.Mode** [= Integer]

Settings

Value	Constant	Description
1	SimpleGatedEventCNT	
2	SinglePeriodMSR	
3	SinglePulseWidthMSR	
4	SingleGatedPulseGen	
5	SingleTrigPulseGen	
6	RetrigSinglePulseGen	
7	SingleTrigContPulseGen	
8	ContGatedPulseGen	

Remarks
Please refer to the hardware manual for the mode description.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockSource Property
GPTC.Counter1.ClockSource Property

Return/Set a value that determines the Timer/Counter Source.

Syntax
object.**GPTC.Counter0.ClockSource** [= Integer]
object.**GPTC.Counter1.ClockSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKSRC_INT	internal time base
2	GPTC_CLKSRC_EXT	external time base from GPTC0_SRC or GPTC1_SRC pin

Please refer to the hard ware manual for the mode description.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockPolarity Property
GPTC.Counter1.ClockPolarity Property

Return/Set a value that determines the Timer/Counter clock polarity

Syntax
object.**GPTC.Counter0.ClockPolarity** [= Integer]
object.**GPTC.Counter1.ClockPolarity** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKEN_LACTIVE	Low active
2	GPTC_CLKEN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GateSource Property
GPTC.Counter1.GateSource Property

Return/Set a value that determines the Timer/Counter gate source

Syntax

object.**GPTC.Counter0.GateSource** [= Integer]
object.**GPTC.Counter1.GateSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_GATESRC_INT	gate is controlled by software
4	GPTC_GATESRC_EXT	gate is controlled by GPTC0_GATE or GPTC1_GATE pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GatePolarity Property
GPTC.Counter1.GatePolarity Property

Return/Set a value that determines the Timer/Counter gate polarity

Syntax

object.**GPTC.Counter0.GatePolarity** [= Integer]
object.**GPTC.Counter1.GatePolarity** [= Integer]

Settings

Value	Constant	Description
2	GPTC_GATE_LACTIVE	Low active
0	GPTC_GATE_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++

Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownSource Property
GPTC.Counter1.UpDownSource Property

Return/Set a value that determines the Timer/Counter UpDown Source

Syntax

object.**GPTC.Counter0.UpDownSource** [= Integer]
object.**GPTC.Counter1.UpDownSource** [= Integer]

Settings

Value	Constant	Description
0	GPTC_UPDOWN_SEL_INT	Up/Down controlled by software
16	GPTC_UPDOWN_SEL_EXT	Up/Down controlled by GPTC0_UPDOWN orGPTC1_UPDOWN pin

Please refer to th e hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownPolarity Property
GPTC.Counter1.UpDownPolarity Proper ty

Return/Set a value that determines the Timer/Counter UpDown Polarity

Syntax

object.**GPTC.Counter0.UpDownPolarity** [= Integer]
object.**GPTC.Counter1.UpDownPolarity** [= Integer]

Settings

Value	Constant	Description
4	GPTC_UPDOWN_LACTIVE	Low active
0	GPTC_UPDOWN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++

Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.OutputPol arity Property
GPTC.Counter1.OutputPolarity Property

Return/Set a value that determines the Timer/Counter Output Polarity

Syntax

object.**GPTC.Counter0.OutputPolarity** [= Integer]
object.**GPTC.Counter1.OutputPolarity** [= Integer]

Settings

Value	Constant	Description
8	GPTC_OUTPUT_LACTIVE	Low active
0	GPTC_OUTPUT_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.IntGATE Property
GPTC.Counter1.IntGATE Property

Return/Set a value that determines the Timer/Counter Internal gate initialized status

Syntax

object.**GPTC.Counter0.IntGATE** [= Boolean]
object.**GPTC.Counter1.IntGATE** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type

Boolean

GPTC.Counter0.IntUpDnCTR Property
GPTC.Counter1.IntUpDnCTR Property

Return/Set a value that determines the Timer/Counter internal updown counter initialized st atus.

Syntax

object.**GPTC.Counter0.IntUpDnCTR** [= Boolean]
object.**GPTC.Counter1.IntUpDnCTR** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type

Boolean

GPTC.Counter0.DelayCount Property
GPTC.Counter1.DelayCount Property

Return/Set a value that determines the Timer/Counter internal initial count of the GPTC or pulse delay. The counter value of load register 1 of timer/counter. The meaning for the value depends on the mode the timer /counter performs. For mode 1 to mode 3, the value is the initial count of the GPTC. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse delay.

Syntax

object.**GPTC.Counter0.DelayCount** [= Integer]
object.**GPTC.Counter1.DelayCount** [= Integer]

Data Type

Integer

GPTC.Counter0.DurationCount Property
GPTC.Counter1.DurationCount Property

Return/Set a value that determines the Timer/Counter pulse width when mode 4 to mode 8.
The counter value of load register 2 of timer/counter. For mode 1 to mode 3, the value is not used. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse width.

Syntax

object.**GPTC.Counter0.DurationCount** [= Integer]
object.**GPTC.Counter1.DurationCount** [= Integer]

Data Type

Integer

GPTC.Counter0.OutputV alue Property
GPTC.Counter1.OutputValue Property

Returns the counter value of the specified general -purpose timer/counter.
Range: 0 through 65535

Syntax
[Integer=] object**.GPTC.Counter0.OutputValue**
[Integer=] object**.GPTC.Counter1.OutputValue**

Data Type
Integer

GPTC.CALIBRATION.BankTemperature Property

Returns a value that since user's last calibrated temperature in the EEPRON Bank .

Syntax
object**.CALIBRATION.BankTemperature**(*[BankOfEEPROM as Integer]*) As Single

Data Type
Single

GPTC.CALIBRATION.BankDate Property

Returns a value that since user's last calibrated date in the EEPRON Bank .

Syntax
object**.CALIBRATION.BankDate**(*[BankOfEEPROM as Integer]*) As String

Data Type
String

GPTC.CALIBRATION.CurrentTemperature Property

Returns a value that determines the current temperature on card.

Syntax
[Single=] object**.CALIBRATION.CurrentTemperature**

Data Type
Single

GPTC.CALIBRATION.CurrentDate Property

Returns a value that determines the current date.

Syntax
[String=] object.CALIBRATION.CurrentDate

Data Type
String

AI.NumOfScan Property

If double-buffered mode is disabled, the total number of scans to be performed. For double -buffered acquisition, NumOf *Scan* is the size (in samples) allocated for each channel in the circular buffer. This valu e must be a multiple of 2.

Syntax

object.**AI.NumOfScan** [=Long]

Remarks

Non-double-buffer mode

This value multiply the total number of scan channels is the total number of A/D conversions to be performed.

Double-buffer-mode

This value multiply the total number of scan channels is the size (in sample) of the circular buffer.

Data Type

Long

AI.ClockSource Property

The clock source (Time Base) the device selected.

Syntax

object.**AI.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ScanInterval Property

The length of the scan interval (that is, the counter value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.

If the timer base is from *external*, the valid range of the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of the value is 20 through 16777215

Syntax

object.**AI.ScanInterval** [=Long]

Data Type

Long

AI.ConversionSource Property

The A/D Conversion Source Selection.

Syntax
object.**AI.ConversionSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_ADCONVSRC_Int	Internal timer
4	DAQ2K_AI_ADCONVSRC_AFI0	From AFI0 pin
8	DAQ2K_AI_ADCONVSRC_SSI	From SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerMode Property
The Trigger Mode Selection.

Syntax
object.**AI.TriggerMode** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_TRGMOD_POST	Post Trigger Mode
8	DAQ2K_AI_TRGMOD_DELAY	Delay Trigger Mode
16	DAQ2K_AI_TRGMOD_PRE	Pre-Trigger Mode
24	DAQ2K_AI_TRGMOD_MIDL	Middle-Trigger Mode

Remarks
Please refer to the hardware manual for the trigger mode description.

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerSource Property
The Trigger Source Selection.

Syntax
object.**AI.TriggerSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_TRGSRC_SOFT	Software
1	DAQ2K_AI_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_AI_TRGSRC_ExtD:	From external digital trigger pin
3	DAQ2K_AI_TRSRC_SSI	From SSI source

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.DelaySource Property

The delay source selection.

Syntax

object.**AI. DelaySource** [=Short]

Settings

Value	Constant	Description
256	DAQ2K_AI_Dly1InSamples	Delay in samples
0	DAQ2K_AI_Dly1InTimebase	Delay in time base

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReTriggerModeEnable Property

Re-trigger in an acquisition is enabled

Syntax

object.**AI.ReTriggerModeEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	ReTriggerMode is disabled.
1	True	ReTriggerMode is enabled.

Data Type
Boolean

AI.MCounterEnable Property

This constant is only valid for Pre -trigger and Middle trigger mode
Mcounter is enabled and then the trigger signal is ignore before M terminal count is reached.

Syntax
object.**AI.MCounterEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	MCounter is disabled.
1	True	Mcounter is enabled.

Data Type
Boolean

AI.PostTriggerCount Property

This constant is only valid for Middle trigger mode,
The PostTriggerCount indicates the number of data will be accessed after a specific trigger event.

Syntax
object.**AI.PostTriggerCount** [=Long]

Data Type
Long

AI.DelayCount Property

This constant is only valid for Delay trigger mode,
The DelayCount indicates the number of data or timer ticks will be ignored after a specific trigger event.

Syntax
object.**AI.DelayCount** [=Long]

Data Type
Long

AI.MCount Property

The counter value of MCounter . This argument is only valid for Pretrigger and Middle trigger mode.

Syntax
object.**AI.MCount** [=Long]

Data Type
Long

AI.ReTriggerCount Property

The accepted trigge r times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax

object.**AI.ReTriggerCount** [=Long]

Data Type

Long

AI.ExtTrigPolarity Property

External Digital **Trigger Polarity.**

Syntax

object.**AI.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_AI_TrgPositive	Trigger positive edge active
4096	DAQ2K_AI_TrgNegative	Trigger negative edge active

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReturnType Property

Return/Set a value that determines the return data type of analog input when AiComplete or AiHalfReady event would occur.

Syntax

object.**AI.ReturnType** [=Integer]

Settings

Value	Constant	Description
0		Scaled data only
1		Binary codes without channel only
2		Scaled data and binary codes without channel
3		No data return

Data Type

Integer

AI.DoubleBufferMode Property

Enables or disables double -buffered data acquisition mode.

Syntax
object.**AI.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type
Boolean

AI.StreamToFile Property

Return/Set a value that determines if the contr ol is enabled the function of streaming data to disk file. This argument is only valid for Delay trigger

Syntax
object.**AI.StreamToFile** [=Boolean]

Settings

Value	Constant	Description
0	False	Disable the function of streaming data to disk file.
1	True	Enable the function of streaming data to disk file

Data Type
Boolean

AI.FileName Property

FileName specified the file name of streaming data to disk. This argument is only valid for AI.StreamToFile is Enable.

Syntax
object.**AI.FileName** [=String]

Data Type
String

AI.AIOAnalogTrigCtrl Property

Return/Set a value that determines the t rigger source selection

Syntax
object.**AI.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	CH0ATRIG	AI channel 0
2	CH1ATRIG	AI channel 1
4	CH2ATRIG	AI channel 2
6	CH3ATRIG	AI channel 3
1	EXTATRIG	From external analog trigger pin

Data Type
Integer

AI.AIOTrigCondition Property

Return/Set a value that determines the t rigger condition selection

Syntax

object.**AI.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above-High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks

Please refer to the hardware manual for the more description.

Data Type

Integer

AI.AIOHLevel Property

The High value setting of Trigger level. Please refer to the hardware manual for the relationship between the val ue of *TrgLevel* and trigger voltage.

Syntax

object.**AI.AIOHLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	Trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AI.AIOLLevel Property

The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AI.AIOLLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AI.Channels(0).Enable Property
AI.Channels(1).Enable Property
AI.Channels(2).Enable Property
AI.Channels(3).Enable Property

DAQ-2000 channel-gain that can be set sep arately for each channel to perform multi-channel/gain analog input, The parameter 0~3 is channel id.

Syntax
object.**AI.Channels**(0).Enable [=Boolean]

Data Type
Boolean

AI.Channels(0).Range Property
AI.Channels(1).Range Property
AI.Channels(2).Range Property
AI.Channels(3).Range Property

DAQ-20XX channel-gain that can be set separately for each channel to perform multi-channel/gain analog input, The parameter 0~3 is channel id. This property can setting differential range for each channel.

Syntax
object.**AI.Channels**(0).Range [=Integer]

Settings

Value	Constant	Description
1	AD_B_10_V	Bipolar -10V to +10V
2	AD_B_5_V	Bipolar -5V to +5V
3	AD_B_2_5_V	Bipolar -2.5V to +2.5V
4	AD_B_1_25_V	Bipolar -1.25V to +1.25V
15	AD_U_10_V	Unipolar 0 to +10V
16	AD_U_5_V	Unipolar 0 to +5V
17	AD_U_2_5_V	Unipolar 0 to +2.5V
18	AD_U_1_25_V	Unipolar 0 to +1.25V

Data Type
Integer

Microsoft C/C++ and Borland C++

Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.CHUI Property

The length of the Channel Update interval (that is, the counter value between the initiation of each update sequence).
If the timer base is from *external*, the valid range of the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid ran ge of the value is as follows:
Range: 40 through 16777215

Syntax

object.**AO.CHUI** [=Long]

Data Type

Long

AO.DAWRSource Property

Return/Set a value that determines the D/A R/W Source Selection

Syntax

object.**AO.DAWRSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_WRSRC_Int	Internal timer
1	DAQ2K_DA_WRSRC_AFI0	From AFI0 pin
2	DAQ2K_DA_WRSRC_SSI	From SSI source

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerSource Property

Return/Set a value that determines the trigger source selection

Syntax

object.**AO.TriggerSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGSRC_SOFT	Software
1	DAQ2K_DA_TRGSRC_ANA	From analog trigger pin

Value	Constant	Description
2	DAQ2K_DA_TRGSRC_ExtD	From external digital trigger pin
3	DAQ2K_DA_TRSRC_SSI	From SSI source

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerMode Property
Return/Set a value that determines the trigger mode selection

Syntax
object**.AO.TriggerMode** [=Integer]

Settings
Value	Constant	Description
0	DAQ2K_DA_TRGMOD_POST	Post Trigger Mode
1	DAQ2K_DA_TRGMOD_DELAY	Delay Trigger Mode

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay1Source Property
Return/Set a value that determines the delay1 source selection

Syntax
object**.AO.Delay1Source** [=Integer]

Settings
Value	Constant	Description
64	DAQ2K_DA_Dly1InUI	Delay in samples
0	DAQ2K_DA_Dly1InTimebase	Delay in time base

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay2Source Property

Return/Set a value that determines the delay2 source selection

Syntax

object.**AO.Delay2Source** [=Integer]

Settings

Value	Constant	Description
128	DAQ2K_DA_Dly2InUI	Delay in samples
0	DAQ2K_DA_Dly2InTimebase	Delay in time base

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ExtTrigPolarity Property

Return/Set a value that determines the e xternal digital trigger polarity selection

Syntax

object.**AO.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TrgPositive	Trigger positive edge active
512	DAQ2K_DA_TrgNegative	Trigger negative edge active

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerCount Property

The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax
object.**AO.ReTriggerCount** [=Long]

Data Type
Long

AO.Delay1Count Property

The counter value of DLY1 Counter (the delay time after the trigger signal to the start of the waveform generation) . This argument is only valid for Delay trigger mode.

Syntax
object.**AO.Delay1Count** [=Long]

Data Type
Long

AO.Delay2Count Property

The counter value of DLY2 Counter (the Delay between two consecutive waveform generations).

Syntax
object.**AO.Delay2Count** [=Long]

Data Type
Long

AO.ClockSource Property

The clock source (Time Base) the device selected.

Syntax
object.**AO.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerModeEnable Property

Re-trigger in an acquisition is enabled

Syntax
object.**AO.ClockSource** [=Boolean]

Settings

Value	Constant	Description
0	False	ReTriggerMode is disabled
1	True	ReTriggerMode is enabled

Data Type
Boolean

AO.AIOAnalogTrigCtrl Property
Return/Set a value that determines the t rigger source selection

Syntax
object.**AO.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	CH0ATRIG	AI channel 0
2	CH1ATRIG	AI channel 1
4	CH2ATRIG	AI channel 2
6	CH3ATRIG	AI channel 3
1	EXTATRIG	From external analog trigger pin

Data Type
Integer

AO.AIOTrigCondition Property
Return/Set a value that determines the t rigger condition selection

Syntax
object.**AO.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above-High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks
Please refer to the hardware manual for the more description.

Data Type
Integer

AO.AIOHLevel Property
The High value setting of Trigger level. Please refer to the hardware manual for the relationship

between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AO.AIOHLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AO.AIOLLevel Property

The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AO.AIOLLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AO.DoubleBufferMode Property

Enables or disables double-buffered data acquisition mode.

Syntax

object.**AO.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type
Boolean

AO.Iterations Property

The times of number of the data in the buffer to output to the port. a value of zero is not allowed.

Syntax
object.**AO.Iterations** [=Long]

Data Type
Long

AO.Definite Property

Waveform generation proceeds definite or indefinitely. If double -buffered mode is enabled, this parameter is of no use.

Syntax
object.**AO.Definite** [=Boolean]

Settings

Value	Constant	Description
0	False	Indefinitely
1	True	Definite

Data Type
Boolean

AO.StopMode Property

Return/Set a value that determines DA transfer terminati on mode selected.

Syntax
object.**AO. StopMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TerminateImmediate	Software terminate the DA continuous operation immediately
1	DAQ2K_DA_TerminateUC	Software terminate the DA continuous operation on next update counter terminal count
2	DAQ2K_DA_TerminateIC	Software terminate the DA continuous operation on iteration count

Remarks
Please refer to the hardware manual for the more description.

Data Type
Integer

AO.Channels(0).Enable Property

AO.Channels(1).Enable Property

DAQ-2000 output channel that can be set separately for each channel to perform multi-channel analog input, The parameter 0~1 is channel id.

Syntax

object.**AO.Channels**(0).Enable [=Boolean]

Data Type

Boolean

AO.Channels(0).OutputPolarity Property
AO.Channels(1).OutputPolarity Property

Return/Set a value that determines polarity (unipolar or bipolar) of the output channel.

Syntax

object.**AO.Channels**(0).OutputPolarity [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_UniPolar	Unipolar
1	DAQ2K_DA_BiPolar	Bipolar

Data Type

Integer

AO.Channels(0).IntOrExtref Property
AO.Channels(1).IntOrExtref Property

Return/Set a value that determines DA reference voltage source of the output channel.

Syntax

object.**AO.Channels**(0). IntOrExtref [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_Int_REF	internal reference
1	DAQ2K_DA_Ext_REF	external reference

Data Type

Integer

AO.Channels(0).RefVoltage Property
AO.Channels(1).RefVoltage Property

If the D/A reference voltage source your device use is internal reference, the valid values is 10.
If the D/A reference voltage source your device use is external reference, the valid range is -10 to +10.

Syntax
object**.AO.Channels**(0).RefVoltage [=Single]

Data Type
Single

AO.Channels(0).Buffer1 Property
AO.Channels(1).Buffer1 Property

This property set up the buffer for continuous analog output operation.
A buffer data or a array of buffer data, data type is integer.

Syntax
object**.AO.Channels**(0).Buffer1 [=Variant]

Remarks
You must assign this property before call StartContAO() method.
This property will be used when single or double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type
Variant(Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i
 Daq2010.AO.Channels(0).buffer1 = buffer1
 Daq2010.AO.Channels(0).Enable = True
 Daq2010.AO.StartContAO
```

AO.Channels(0).Buffer2 Property
AO.Channels(1).Buffer2 Property

This property set up the buffer for continuous analog output operation. A buffer data or a array of buffer data, data type is integer.
This property only available when double buffer mode.

Syntax
object**.AO.Channels**(0).Buffer2 [=Variant]

Remarks
You must assign this property before call StartContAO() method.
This property will be used when double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type
Variant(Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i

 For i = 0 To 4095
      If i < 2048 Then
           buffer2(i) = (&H800 + i Mod 2048) And &HFFF
      Else
           buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
      End If
 Next i

 Daq2010.AO.Channels(0).buffer1 = buffer1
 Daq2010.AO.Channels(0).buffer2 = buffer2
 Daq2010.AO.DoubleBufferMode = True
 Daq2010.AO.Channels(0).Enable = True
 Daq2010.AO.StartContAO
```

SSI.ADCONV Property
Connect / Disconnect a SSI_ADCONV device signal to the specified SSI bus trigger line.

Syntax
object**.SSI.ADCONV** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.ADTRIG Property
Connect / Disconnect a SSI_ADTRIG device signal to the specified SSI b us trigger line.

Syntax
object**.SSI. ADTRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.DATRIG Property

Connect / Disconnect a SSI_DATRIG device signal to the specified SSI bus trigger line.

Syntax

object**.SSI.DATRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified S SI bus trigger line

Data Type

Boolean

SSI.DAWR Property

Connect / Disconnect a SSI_DAWR device signal to the specified SSI bus trigger line.

Syntax

object**.SSI.DAWR** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specif ied SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.TIMEBASE Property

Connect / Disconnect a SSI_TIMEBASE device signal to the specified SSI bus trigger line.

Syntax

object**.SSI.TIMEBASE** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

DAQ_2010 Methods

Open Method

Syntax
Function object.**Open** ([ErrMsgBox As Variant]) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
[ErrMsgBox As Variant]

Boolean type.
True: It will popup error message dialog box when operation error.
False: It will fire DAQError event instead of popping up dialog when operation error.

Remarks
This method will be used when the OpenMode property is Manual.

Note
In VC++, *ErrMsgBox* is a VARIANT of VT_I2.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

ShowPropertyPages Method
This method will show propertypages of ActiveX Control.

Syntax
Function object.***ShowPropertyPages***() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

AboutBox Method
This method will show About ADLINK dialog box.

Syntax
Function object.**AboutBox**( ) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

DIO.ReadDIPort Method

Syntax
Function object.**DIO.ReadDIPort** (port As Integer, value as Variant) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port

Value	Constant	Description
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Returns the digital data read from the specified port. The returned value is 8 -bit data.

Remarks

You can read data from the digital input port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.ReadDILine Method

Syntax

Function object.**DIO.ReadDILine** (port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

The digital line to be read. The valid va lue is 0 through 7

value As Variant

Returns the digital logic state, 0 or 1, of the specified line.

Remarks

You can read data from the digital input port.

Note

In VC++, value is a VARIANT of VT_I2.

DIO.WriteDOPort Method

Syntax

Function object.**DIO.WriteDOPort** (port As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value as Variant

8-bit data that will be written to the digital output port.

Remarks

Users can write data to the digital output port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.WriteDOLine Method

Syntax

Function object**.DIO.WriteDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

Selects one line number from the indicated port: from 0 to 7 (8-bit port).

value As Variant

Sets 0 or 1 to the indicated line.

Note

In VC++, value is a VARIANT of VT_I2.

DIO.ReadBackDOPort Method

Reads back data from the indicated digital output port.

Syntax

Function object**.DIO.ReadBackDOPort** (port As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Data that is read back from the indicated port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.ReadBackDOLine Method

Reads back data from the indicated digital output line of the indicated digital output port.

Syntax
Function object.**DIO.ReadBackDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

Selects one line number from the indicated port: from 0 to 7 (8-bit port).

value As Variant

Data that is read back from the indicated line.

Note
In VC++, value is a VARIANT of VT_I2.

GPTC.Counter0.Start Method
GPTC.Counter1.Start Method
Start counter operation with the specified mode.

Syntax
Function object.**GPTC.Counter0.Start**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Remarks
You can start the indicated counter to operate in the specified mode.

GPTC.Counter0.Stop Method
GPTC.Counter1.Stop Method
Stop counter operation.

Syntax
Function object.**GPTC.Counter0.Stop**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

GPTC.Counter0.Reset Method
GPTC.Counter1.Reset Method
Halts the specified general-purpose timer/counter operation and reload the initial value of the timer/counter.

Syntax
Function object.**GPTC.Counter0.Reset**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

GPTC.Counter0.ReadStatus Method
GPTC.Counter1.ReadStatus Method

Reads the latched GPTC status of the general -purpose counter from the GPTC status register.

Syntax
Function object.**GPTC.Counter0.ReadStatus**(Clock *As Integer, Output As Integer, Gate As Integer, Updown As Integer,*) As Boolean

Return Value
True if the function is successful; otherwise False.

AI.StartContAI Method

This method performs continuous A/D conversions on the specified analog input channels at a rate as close to the rate you specified. This method takes advan tage of the DAQ-2000 channel-gain that can be set separately for each channel to perform multi-channel/gain analog input. This method will fire AiComplete or AiHalfReady event depends on AI.DoubleBufferMode property.

Syntax
Function object.**AI.StartContAI**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Remarks
You can use this method to start the DMA analog input function. If the AI.StreamToFile property is True then the DMA data will be written to the file specified by AI.FileName. Otherwise, the AI.FileName property will be ignored.
The data file is written in binary format, with the lower byte first (little endian). Data type is "Binary codes with miscellaneous data". DAQBench provides a convenient tool DAQCvt to convert the binary file to the file format read easily. See DAQBench User's Guide for the usage of the utility. If you want to handle the data by yourself, please refer to Appendix D Data File Format for the file structure.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example
```
Daq2010.AI.Channels(0).Range = AD_B_10_V
Daq2010.AI.Channels(2).Enable = True
Daq2010.AI.Channels(2).Range = AD_U_1_25_V
```

```
Daq2010.AI.Channels(0).Enable = True
Daq2010.AI.StartContAI

Private Sub Daq2010_AiComplete(ScaledData As Variant, BinaryCodes As Variant)
    ' Get Data in    ScaledData
End Sub

Private Sub Daq2010_AiHalfReady(ScaledData As Variant, BinaryCodes As Variant)
    ' Get Data in    ScaledData
End Sub
```

AI.StopContAI Method

You can use this method to force stop DMA analog input.

Syntax

Function object**.AI.StopContAI** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AI.ReadChannels Method

This method performs a software triggered A/D conversion (analog input) on analog input channels and returns the values converted.

Syntax

Function object**.AI.ReadChannels**(Buffer As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Buffer As Variant

An integer array to contain the acquired data. Please refer to Appendix C AD Data Format for the data format in the Buffer.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Daq2010.AI.Channels(0).Range = AD_B_10_V
Daq2010.AI.Channels(2).Enable = True
Daq2010.AI.Channels(2).Range = AD_U_1_25_V
Daq2010.AI.Channels(0).Enable = True

Private Sub Timer1_Timer()
    Dim vBuffer As Variant
    Daq2010.AI.ReadChannels vBuffer
    ' Get Data in    vBuffer
End Sub
```

AO.StartContAO Method

This method performs continuous D/A conversions on the specified analog output channel at a rate as close to the rate you specified. This method will fire AoComplete or AoBufferReady event depends on AO.DoubleBufferMode property.

Syntax

Function object**.AO.StartContAO**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i

 For i = 0 To 4095
      If i < 2048 Then
           buffer2(i) = (&H800 + i Mod 2048) And &HFFF
      Else
           buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
      End If
 Next i

 Daq2010.AO.Channels(0).buffer1 = buffer1
 Daq2010.AO.Channels(0).buffer2 = buffer2
 Daq2010.AO.DoubleBufferMode = True
 Daq2010.AO.Channels(0).Enable = True
 Daq2010.AO.StartContAO
```

AO.StopContAO Method

You can use this method to force stop DMA analog output.

Syntax

Function object**.AO.StopContAO** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AO.WriteChannel Method

Accepts a voltage value (or a current value), scales it to the proper binary value and writes a binary value to the specified analog output channel.

Syntax

Function object.**AO.WriteChannel**(Channel as Integer, Voltage as Single) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Channel as Integer

The analog output channel number.
Range: 0 or 1 for DAQ -2204

Voltage as Single

The value to be scaled and written to the analog output channel. The range of voltages depends on the type of device, on the output polarity, and on the voltage reference (external or internal).

CALIBRATION.AutoCalibration Method

Uses this method to calibrate your DAQ -2000 device. When the method is called, the device goes into a self-calibration cycle. The method does not return until the self-calibration is completed.

Syntax

Function object.**CALIBRATION.AutoCalibration**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

CALIBRATION.DisplayErrors Method

Uses this method to fire AcquireADError and AcquireDAError events. Through those events user can identification DAQ-2000 device current status.

Syntax

Function object.**CALIBRATION.DisplayErrors** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Example

Daq2010.CALIBRATION.DisplayErrors

```
Private Sub Daq2010_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "BiPolar"
    End If
    strMsg = "AD channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####") & " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
```

```
    List1.Refresh
End Sub

Private Sub Daq2010_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub
```

CALIBRATION.Load Method

Load calibration constants from the specified bank of EEPROM.

Syntax

Function object**.CALIBRATION.Load**(BankOfEEPROM As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

CALIBRATION.Save Method

Save calibration constants to the specified bank of EEPROM.

Syntax

Function object**.CALIBRATION.Save**(BankOfEEPROM As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

SSI.ClearAll Method

Disconnects all of the device signals from the SSI bus trigger lines.

Syntax

Function object**.ClearAll** As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DAQError Event

Syntax

sub ControlName_DAQError (ErrString As String)

Arguments

ErrString As String
The string of error reasion

Remarks

This event will occur when some error occur in control

AiComplete Event

Syntax

sub ControlName_AiComplete(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant

The analog input data array that have been translated to the engineering data (voltage) according to AI.Channels(n).Range property.

BinaryCodes As Variant

The analog input data array with digital format.

Remarks

This event occurs when continuous analog input function is completed. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting.*"BinaryCodes without channel only"*.
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AiHalfReady Event

Syntax

sub ControlName_AiHalfReady(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant

The analog input data array that have been translated to the engineering data (voltage) according to AIRange property.

BinaryCodes As Variant

The analog input data array with digital format.

Remarks

This event occurs when one half-buffer of the circular buffer is full at continuous analog input operation. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting. "*BinaryCodes without channel only*"
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AoComplete Event

Syntax
sub ControlName _AoComplete( )

Arguments
None

Remarks
This event occurs when continuous analog output function is completed.

AoBufferReady Event

Syntax
sub ControlName_AoBufferReady(BufferIndex as Integer)

Arguments
BufferIndex as Integer, The index of the buffer just output.

Remarks
This event occurs when enable the AO.DoubleBufferMode, If User want to dynamic change the output pattern, process it when receive AoBufferReady event.

Example

```
'This sample code show that how to output four buffers (pattern) in one channel.
'You can modify this code for dynamic changes pattern.

Dim varArray(0 To 3) As Variant
Dim nCounter As Integer

Private Sub AO_Click()

    Dim buffer0(0 To 4095) As Integer
    Dim buffer1(0 To 4095) As Integer
    Dim buffer2(0 To 4095) As Integer
    Dim buffer3(0 To 4095) As Integer

    Dim i As Double

    For i = 0 To 4095
        buffer0(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer1(i) = (&H800 + i Mod 2048) And &HFFF
        Else
            buffer1(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        End If
    Next i
    For i = 0 To 4095
        buffer2(i) = (Cos(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer3(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        Else
            buffer3(i) = (&H800 + i Mod 2048) And &HFFF
        End If
    Next i

    varArray(0) = buffer0
```

```
    varArray(1) = buffer1
    varArray(2) = buffer2
    varArray(3) = buffer3

    Daq2010.AO.Channels(0).buffer1 = varArray(0)
    Daq2010.AO.Channels(0).buffer2 = varArray(1)
    nCounter = 1
    Daq2010.AO.Channels(0).Enable = True

    Daq2010.AO.DoubleBufferMode = True
    Daq2010.AO.StartContAO
End Sub

Private Sub Daq2010_AoBufferReady(ByVal BufferIndex As Integer)

nCounter = nCounter + 1
nCounter = nCounter Mod 4

If BufferIndex = 1 Then
    ' It can change buffer1 in here
    Daq2010.AO.Channels(0).buffer1 = varArray(nCounter)
End If

If BufferIndex = 2 Then
    ' It can change buffer2 in here
    Daq2010.AO.Channels(0).buffer2 = varArray(nCounter)
End If

End Sub
```

AcquireADError Event

Acquires the offset and gain errors of the specified AI channel in the specified polari ty mode.

Syntax

sub *ControlName*_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)

Arguments

***channel as Integer*,**
Indicate channel id.

polarity *as Integer,*
Indicate polarity

Value	Description
0	Unipolar
1	Bipolar

gain_err as Double
Indicate gain error

offset_err as Double
Indicate offset error

Example

```
Daq2010.CALIBRATION.DisplayErrors

Private Sub Daq2010_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
```

```
    End If
    strMsg = "AD channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub
```

AcquireDAError Event

Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

Syntax

sub *ControlName*_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)

Arguments

***channel as Integer*,**

Indicate channel id.

polarity *as Integer,*

Indicate polarity

Value	**Description**
0	Unipolar
1	Bipolar

gain_err as Double

Indicate gain error

offset_err as Double

Indicate offset error

Example

Daq2010.CALIBRATION.DisplayErrors

```
Private Sub Daq2010_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
 List1.AddItem (strMsg)
 List1.Refresh
 End Sub
```

Daq2204 ActiveX Control

The Daq2204 ActiveX control is a software component that provides the interface for users to control PCI-2204 card.

Daq2204 ActiveX Control Overview			
Type (Group)	**Property**	**Method**	**Event**
General	DASKCardType	Open	DAQError
	CardNumber	AboutBox	
	OpenMode	ShowPropertyPages	
	DaskCardID		
DIO	P1Adir	ReadBackDOLine	
	P1Bdir	ReadBackDOPort	
	P1ClowerDir	ReadDILine	
	P1CupperDir	ReadDIPort	
		WriteDOLine	
		WriteDOPort	
GPTC	Mode	Start	
	ClockSource	Stop	
	ClockPolarity	Reset	
	GateSource	ReadStatus	
	GatePolarity		
	UpDownSource		
	UpDownPolarity		
	OutputPolarity		
	IntGATE		
	IntUpDnCTR		
	DelayCount		
	DurationCount		
	OutputValue		
AI	NumOfScan	StartContAI	AiComplete
	ClockSource	StopContAI	AiHalfReady
	ScanInterval	ReadChannels	
	SamplingInterval	MuxScanSetup	
	ConversionSource		
	TriggerMode		
	TriggerSource		
	DelaySource		
	ReTriggerModeEnable		
	McounterEnable		
	PostTriggerCount		
	DelayCount		

Daq2204 ActiveX Control Overview			
	Mcount		
	ReTriggerCount		
	ExtTrigPolarity		
	ReturnType		
	DoubleBufferMode		
	StreamToFile		
	FileName		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
AO	CHUI	StartContAO	AoComplete
	DAWRSource	StopContAO	AoBufferReady
	TriggerSource	WriteChannel	
	TriggerMode		
	Delay1Source		
	Delay2Source		
	ExtTrigPolarity		
	ReTriggerCount		
	Delay1Count		
	Delay2Count		
	ClockSource		
	ReTriggerModeEnable		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
	DoubleBufferMode		
	Iterations		
	Definite		
	StopMode		
	Channels(n).Enable		
	Channels(n).IntOrExtref		
	Channels(n).OutputPolarity		
	Channels(n).RefVoltage		
	Channels(n).Buffer1		
	Channels(n).Buffer2		
CALIBRATION	BankTemperature	AutoCalibration	Acquire22XXADError
	BankDate	Load	AcquireDAError

Daq2204 ActiveX Control Overview			
	CurrentTemperature	Save	
	CurrentDate	DisplayErrors	
SSI	TIMEBASE	ClearAll	
	ADCONV		
	DAWR		
	ADTRIG		
	DATRIG		

DASKCardType Property

Return a value that determines the card type. It is always DAQ_2204 in DAQ-2204 device.
.

Syntax

[Integer] =object**.CardType**

Remarks

Always return DAQ_2204 (for DAQ -2204 device)

Settings

Value	Constant	Description
5	DAQ_2204	For DAQ-2004 Device

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Data Type

Integer

CardNumber Property

The sequence number of the card w ith ***the same card type*** plugged in the PCI slot.
The card sequence number setting is according to the PCI slot sequence in the mainboard.
The first card (in the most prior slot) is with CardNumber=0. For example,
If there are two DAQ -2204 cards plugged on your PC, the DAQ -2204 card in the prior slot should be registered with CardNumber =0, and the other one with CardNumber =1.
.

Syntax

object**.CardNumber** [= short]

Remarks

This property will be used when Initializes the hardware states of a DAQ -2K data acquisition card.

Data Type

Integer

OpenMode Property

Return/Set a value that determines the mode of opening device.

Syntax

object**.OpenMode** [= short]

Settings

Value	Display	Description
0	Automatic	D2K-OCX will initialize by itself. Automatically open device when the control was created
1	Manual	Call object**.Open**() method to initialized D2K-OCX(open device)
2	Demonstration	D2K-OCX will provide software DAQ data to simulating hardware

Value	Display	Description
		drivers.

Data Type
Integer

DaskCardID Property

Returns a value that determines the D2K_Register_Card() returns value,
the DaskCardID is a numeric card ID that will be used by other D2K -DASK library functions.

Syntax
[short] =object.**DaskCardID**

Remarks
The range of card id i s between 0 and 31.
This property will be used when combine D2K-DASK and D2K-OCX two module in one program.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example
This sample will demonstration If user want to check AI completed by itself.

```
' OCX to starting continuous AI
 Dim nDaskID as Integer
 Daq2204.Open(TRUE)
 nDaskID = Daq2204.DaskCardID

Daq2204.AI.StartContAI

' Check AI Completed by DASK    API
Dim Stopped    As Byte
Dim AccessCnt As Long

Do While True
    D2K_AI_AsyncCheck nDaskID, Stopped, AccessCnt
    If Stopped = 1 Then
        Exit Do
    End If
Loop
MsgBox "AI Complete"
```

DIO.P1Adir Property

Return/Set a value that determine s P1A port direction.

Syntax
object.**DIO.P1ADir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1Bdir Property
Return/Set a value that determines P1B port direction.

Syntax
object.**DIO.P1BDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CLowerdir Property
Return/Set a value that determines P1C lower port direction.

Syntax
object.**DIO.P1CLowerDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CUpperdir Property

Return/Set a value that determines P1C upper port direction.

Syntax

object.**DIO.P1CUpperDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.Mode Property

Return/Set a value that determines the Timer/Counter mode.

Syntax

object.**GPTC.Counter0.Mode** [= Integer]

Settings

Value	Constant	Description
1	SimpleGatedEventCNT	
2	SinglePeriodMSR	
3	SinglePulseWidthMSR	
4	SingleGatedPulseGen	
5	SingleTrigPulseGen	
6	RetrigSinglePulseGen	
7	SingleTrigContPulseGen	
8	ContGatedPulseGen	

Remarks

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic

Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockSource Property
GPTC.Counter1.ClockSource Property

Return/Set a value that determines the Timer/Counter Source.

Syntax

object.**GPTC.Counter0.ClockSource** [= Integer]
object.**GPTC.Counter1.ClockSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKSRC_INT	internal time base
2	GPTC_CLKSRC_EXT	external time base from GPTC0_SRC or GPTC1_SRC pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockPolarity Property
GPTC.Counter1.ClockPolarity Property

Return/Set a value that determines the Timer/Counter clock polarity

Syntax

object.**GPTC.Counter0.ClockPolarity** [= Integer]
object.**GPTC.Counter1.ClockPolarity** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKEN_LACTIVE	Low active
2	GPTC_CLKEN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GateSource Property
GPTC.Counter1.GateSource Property

Return/Set a value that determines the Timer/Counter gate source

Syntax

object.**GPTC.Counter0.GateSource** [= Integer]
object.**GPTC.Counter1.GateSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_GATESRC_INT	gate is controlled by software
4	GPTC_GATESRC_EXT	gate is controlled by GPTC0_GATE or GPTC1_GATE pin

Please refer to the hardw are manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GatePolarity Property
GPTC.Counter1.GatePolarity Property

Return/Set a value that determines the Timer/Counter gate polarity

Syntax

object.**GPTC.Counter0.GatePolarity** [= Integer]
object.**GPTC.Counter1.GatePolarity** [= Integer]

Settings

Value	Constant	Description
2	GPTC_GATE_LACTIVE	Low active
0	GPTC_GATE_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownSource Property
GPTC.Counter1.UpDownSource Property

Return/Set a value that determines the Timer/Counter UpDown Source

Syntax

object.**GPTC.Counter0.UpDownSource** [= Integer]
object.**GPTC.Counter1.UpDownSource** [= Integer]

Settings

Value	Constant	Description
0	GPTC_UPDOWN_SEL_INT	Up/Down controlled by software
16	GPTC_UPDOWN_SEL_EXT	Up/Down controlled by GPTC0_UPDOWN orGPTC1_UPDOWN pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownPolarity Property
GPTC.Counter1.UpDownPolarity Property

Return/Set a value that determines the Timer/Counter UpDown Polarity

Syntax

object.**GPTC.Counter0.UpDownPolarity** [= Integer]
object.**GPTC.Counter1.UpDownPolarity** [= Integer]

Settings

Value	Constant	Description
4	GPTC_UPDOWN_LACTIVE	Low active
0	GPTC_UPDOWN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi

Unit: D2kDask.pas

GPTC.Counter0.OutputPolarity Property
GPTC.Counter1.OutputPolarity Property

Return/Set a value that determines the Timer/Counter Output Polarity

Syntax

object.**GPTC.Counter0.OutputPolarity** [= Integer]
object.**GPTC.Counter1.OutputPolarity** [= Integer]

Settings

Value	Constant	Description
8	GPTC_OUTPUT_LACTIVE	Low active
0	GPTC_OUTPUT_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.IntGATE Property
GPTC.Counter1.IntGATE Property

Return/Set a value that determines the Timer/Counter Internal gate initialized status

Syntax

object.**GPTC.Counter0.IntGATE** [= Boolean]
object.**GPTC.Counter1.IntGATE** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type

Boolean

GPTC.Counter0.IntUpDnCTR Propert y
GPTC.Counter1.IntUpDnCTR Property

Return/Set a value that determines the Timer/Counter internal updown counter initialized status.

Syntax

object.**GPTC.Counter0.IntUpDnCTR** [= Boolean]
object.**GPTC.Counter1.IntUpDnCTR** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type

Boolean

GPTC.Counter0.DelayCount Property
GPTC.Counter1.DelayCount Property

Return/Set a value that determines the Timer/Counter internal initial count of the GPTC or pulse delay. The counter value of load register 1 of timer/counter. The meaning for the value depends on the mode the timer /counter performs. For mode 1 to mode 3, the value is the initial count of the GPTC. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse delay.

Syntax

object.**GPTC.Counter0.DelayCount** [= Integer]
object.**GPTC.Counter1.DelayCount** [= Integer]

Data Type

Integer

GPTC.Counter0.DurationCount Property
GPTC.Counter1.DurationCount Property

Return/Set a value that determines the Timer/Counter pulse width when mode 4 to mode 8.
The counter value of load register 2 of timer/counter. For mode 1 to mode 3, the value is not used. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse width.

Syntax

object.**GPTC.Counter0.DurationCount** [= Integer]
object.**GPTC.Counter1.DurationCount** [= Integer]

Data Type

Integer

GPTC.Counter0.OutputValue Property
GPTC.Counter1.OutputValue Property

Returns the counter value of the specified general -purpose timer/counter. Range: 0 through 65535

Syntax

[Integer=] object.**GPTC.Counter0.OutputValue**
[Integer=] object.**GPTC.Counter1.OutputValue**

Data Type

Integer

GPTC.CALIBRATION.BankTemperature Property

Returns a value that since user's last calibrated temperature in the E EPRON Bank .

Syntax

object.**CALIBRATION.BankTemperature**(*[BankOfEEPROM as Integer]*) As Single

Data Type

Single

GPTC.CALIBRATION.BankDate Property

Returns a value that since user's last calibrated date in the EEPRON Bank .

Syntax

object.**CALIBRATION.BankDate**(*[BankOfEEPROM as Integer]*) As String

Data Type

String

GPTC.CALIBRATION.CurrentTemperature Property

Returns a value that determines the current temperature on card.

Syntax

[Single=] object.**CALIBRATION.CurrentTemperature**

Data Type

Single

GPTC.CALIBRATION.CurrentDate Property

Returns a value that determines the current date.

Syntax

[String=] object.**CALIBRATION.CurrentDate**

Data Type

String

AI.NumOfScan Property

If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, NumOf *Scan* is the size (in samples) allocated for each channel in the circular buffer. This value must be a multiple of 2.

Syntax

object.**AI.NumOfScan** [=Long]

Remarks

Non-double-buffer mode

This value multiply the total number of scan channels is the total number of A/D conversions to be performed.

Double-buffer-mode

This value multiply the total number of scan channels is the size (in sample) of the circular buffer.

Data Type
Long

AI.ClockSource Property
The clock source (Time Base) the device selected.

Syntax
object.**AI.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ScanInterval Property
The length of the scan interval (that is, the counter value be tween the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.
If the timer base is from *external*, the valid range of the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of the value is 14 through 16777215

Syntax
object.**AI.ScanInterval** [=Long]

Data Type
Long

AI.SamplingInterval Property
The length of the sample interval (that is, the counter value between each A/D conversion within a scan sequence). The A/D conversion rate will be *TimeBase/SampIntrv*. The value of *TimeBase(AI.ClockSource)* depends on the card type.
If the timer base is from *external*, the valid range of the value is 2 through 65535.
If the timer base is *Internal timer*, the val id range of the value is as follows:
DAQ-2204 : 14 through 65535

Syntax
object.**AI.SamplingInterval** [=Long]

Data Type
Long

AI.ConversionSource Property

The A/D Conversion Source Selection.

Syntax

object.**AI.ConversionSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_ADCONVSRC_Int	Internal timer
4	DAQ2K_AI_ADCONVSRC_AFI0	From AFI0 pin
8	DAQ2K_AI_ADCONVSRC_SSI	From SSI source

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerMode Property

The Trigger Mode Selection.

Syntax

object.**AI.TriggerMode** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_TRGMOD_POST	Post Trigger Mode
8	DAQ2K_AI_TRGMOD_DELAY	Delay Trigger Mode
16	DAQ2K_AI_TRGMOD_PRE	Pre-Trigger Mode
24	DAQ2K_AI_TRGMOD_MIDL	Middle-Trigger Mode

Remarks

Please refer to the hardware manual for the trigger mode description.

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerSource Property

The trigger source selection.

Syntax
object.**AI.TriggerSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_TRGSRC_SOFT	Software
1	DAQ2K_AI_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_AI_TRGSRC_ExtD:	From external digital trigger pin
3	DAQ2K_AI_TRSRC_SSI	From SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.DelaySource Property
The delay source selection.

Syntax
object.**AI. DelaySource** [=Short]

Settings

Value	Constant	Description
256	DAQ2K_AI_Dly1InSamples	Delay in samples
0	DAQ2K_AI_Dly1InTimebase	Delay in time base

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReTriggerModeEnable Property
Re-trigger in an acquisition is enabled

Syntax
object.**AI.ReTriggerModeEnable** [=Boolean]

Settings

Value	Constant	Description

Value	Constant	Description
0	False	ReTriggerMode is disabled.
1	True	ReTriggerMode is enabled.

Data Type

Boolean

AI.MCounterEnable Property

This constant is only valid for Pre -trigger and Middle trigger mode
Mcounter is enabled and then the trigger signal is ignore before M term inal count is reached.

Syntax

object**.AI.MCounterEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	MCounter is disabled.
1	True	Mcounter is enabled.

Data Type

Boolean

AI.PostTriggerCount Property

This constant is only valid for Middl e trigger mode,
The PostTriggerCount indicates the number of data will be accessed after a specific trigger event.

Syntax

object**.AI.PostTriggerCount** [=Long]

Data Type

Long

AI.DelayCount Property

This constant is only valid for Delay trigger mod e,
The DelayCount indicates the number of data or timer ticks will be ignored after a specific trigger event.

Syntax

object**.AI.DelayCount** [=Long]

Data Type

Long

AI.MCount Property

The counter value of MCounter. This argument is only valid for Pretrigger and Middle trigger mode.

Syntax

object**.AI.MCount** [=Long]

Data Type
Long

AI.ReTriggerCount Property

The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax
object.**AI.ReTriggerCount** [=Long]

Data Type
Long

AI.ExtTrigPolarity Property

External Digital Trigger Polarity.

Syntax
object.**AI.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_AI_TrgPositive	Trigger positive edge active
4096	DAQ2K_AI_TrgNegative	Trigger negative edge active

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReturnType Property

Return/Set a value that determines the return data t ype of analog input when AiComplete or AiHalfReady event would occur.

Syntax
object.**AI.ReturnType** [=Integer]

Settings

Value	Constant	Description
0		Scaled data only
1		Binary codes without channel only
2		Scaled data and binary codes without channel
3		No data return

Data Type
Integer

AI.DoubleBufferMode Property

Enables or disables double -buffered data acquisition mode.

Syntax

object.**AI.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type

Boolean

AI.StreamToFile Property

Return/Set a value that determines if the control is enabled the function of streaming data to disk file. This argument is only valid for Delay trigger

Syntax

object.**AI.StreamToFile** [=Boolean]

Settings

Value	Constant	Description
0	False	Disable the function of streaming data to disk file.
1	True	Enable the function of streaming data to disk file

Data Type

Boolean

AI.FileName Property

FileName specified the f ile name of streaming data to disk. This argument is only valid for AI.StreamToFile is Enable.

Syntax

object.**AI.FileName** [=String]

Data Type

String

AI.AIOAnalogTrigCtrl Property

Return/Set a value that determines the t rigger source selection

Syntax

object.**AI.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	ADCATRIG	The first AI channel in the channel-gain queue
1	EXTATRIG	From external analog trigger pin

Data Type

Integer

AI.AIOTrigCondition Property

Return/Set a value th at determines the t rigger condition selection

Syntax

object.**AI.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above -High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks

Please refer to the hardware manual for the more description.

Data Type

Integer

AI.AIOHLevel Property

The High value s etting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AI.AIOHLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AI.AIOLLevel Property

The Low value setting of Trigg er level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AI.AIOLLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage

is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AO.CHUI Property

The length of the Channel Update interval (that is, the counter value between the initiation of each update sequence).
If the timer base is from *external*, the valid range of the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of the value is as follows:
Range: 40 through 16777215

Syntax
object.**AO.CHUI** [=Long]

Data Type
Long

AO.DAWRSource Property

Return/Set a value that determines the D/A R/W Source Selection

Syntax
object.**AO.DAWRSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_WRSRC_Int	Internal timer
1	DAQ2K_DA_WRSRC_AFI0	From AFI0 pin
2	DAQ2K_DA_WRSRC_SSI	From SSI source

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerSource Property

Return/Set a value that determines the trigger source selection

Syntax
object.**AO.TriggerSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGSRC_SOFT	Software
1	DAQ2K_DA_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_DA_TRGSRC_ExtD	From exter nal digital trigger pin
3	DAQ2K_DA_TRSRC_SSI	From SSI source

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerMode Property
Return/Set a value that determ ines the trigger mode selection

Syntax
object.**AO.TriggerMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGMOD_POST	Post Trigger Mode
1	DAQ2K_DA_TRGMOD_DELAY	Delay Trigger Mode

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay1Source Property
Return/Set a value that determines the delay1 source selection

Syntax
object.**AO.Delay1Source** [=Integer]

Settings

Value	Constant	Description
64	DAQ2K_DA_Dly1InUI	Delay in samples
0	DAQ2K_DA_Dly1InTimebase	Delay in time base

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay2Source Property
Return/Set a value that determines the delay2 source selection

Syntax
object.**AO.Delay2Source** [=Integer]

Settings

Value	Constant	Description
128	DAQ2K_DA_Dly2InUI	Delay in samples
0	DAQ2K_DA_Dly2InTimebase	Delay in time base

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ExtTrigPolarity Property
Return/Set a value that determines the e xternal digital trigger polarity selection

Syntax
object.**AO.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TrgPositive	Trigger positive edge active
512	DAQ2K_DA_TrgNegative	Trigger negative edge active

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerCount Property

The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax

object.**AO.ReTriggerCount** [=Long]

Data Type

Long

AO.Delay1Count Property

The counter value of DLY1 Counter (the delay time after the trigger signal to the start of the waveform generation) . This argument is only valid for Delay trigger mode.

Syntax

object.**AO.Delay1Count** [=Long]

Data Type

Long

AO.Delay2Count Property

The counter value of DLY2 Counter (the Delay between two consecutive waveform generations).

Syntax

object.**AO.Delay2Count** [=Long]

Data Type

Long

AO.ClockSource Property

The clock source (Time Base) the device selected.

Syntax

object.**AO.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic

Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerModeEnable Property

Re-trigger in an acquisition is enabled

Syntax

object**.AO.ClockSource** [=Boolean]

Settings

Value	Constant	Description
0	False	ReTriggerMode is disabled
1	True	ReTriggerMode is enabled

Data Type

Boolean

AO.AIOAnalogTrigCtrl Property

Return/Set a value that determines the t rigger source selection

Syntax

object**.AO.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	ADCATRIG	The first AI channel in the channel-gain queue
1	EXTATRIG	From external analog trigger pin

Data Type

Integer

AO.AIOTrigCondition Property

Return/Set a value that determines the t rigger condition selection

Syntax

object**.AO.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above -High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks

Please refer to the hardware manual for the more description.

Data Type

Integer

AO.AIOHLevel Property

The High value setting of Trigger level. Please refer to the hardware manual for the relatio nship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AO.AIOHLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AO.AIOLLevel Property

The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AO.AIOLLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	Trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AO.DoubleBufferMode Property

Enables or disables double -buffered data acquisition mode.

Syntax

object.**AO.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.

Value	Constant	Description
1	True	double-buffered mode is enabled.

Data Type

Boolean

AO.Iterations Property

The times of number of the data in the buffer to output to the port. a value of zero is not al lowed.

Syntax

object.**AO.Iterations** [=Long]

Data Type

Long

AO.Definite Property

Waveform generation proceeds definite or indefinitely. If double -buffered mode is enabled, this parameter is of no use.

Syntax

object.**AO.Definite** [=Boolean]

Settings

Value	Constant	Description
0	False	Indefinitely
1	True	Definite

Data Type

Boolean

AO.StopMode Property

Return/Set a value that determines DA transfer termination mode selected.

Syntax

object.**AO. StopMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TerminateImmediate	Software terminate the DA continuous operation immediately
1	DAQ2K_DA_TerminateUC	Software terminate the DA continuous operation on next update counter terminal count
2	DAQ2K_DA_TerminateIC	Software terminate the DA continuous operation on iteration count

Remarks

Please refer to the hardware manual for the more description.

Data Type

Integer

AO.Channels(0).Enable Property
AO.Channels(1).Enable Property

DAQ-2000 output channel that can be set separately for each channel to perform multi-channel analog input, The parameter 0~1 is channel id.

Syntax

object.**AO.Channels**(0).Enable [=Boolean]

Data Type

Boolean

AO.Channels(0).OutputPolarity Property
AO.Channels(1).OutputPolarity Property

Return/Set a value that determines polarity (unipolar or bipolar) of the output channel.

Syntax

object.**AO.Channels**(0).OutputPolarity [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_UniPolar	Unipolar
1	DAQ2K_DA_BiPolar	Bipolar

Data Type

Integer

AO.Channels(0).IntOrExtref Property
AO.Channels(1).IntOrExtref Property

Return/Set a value that determines DA reference voltage source of the output channel.

Syntax

object.**AO.Channels**(0). IntOrExtref [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_Int_REF	internal reference
1	DAQ2K_DA_Ext_REF	external reference

Data Type

Integer

AO.Channels(0).RefVoltage Property

AO.Channels(1).RefVoltage Property

If the D/A reference voltage source your device use is internal reference, the valid values is 10.
If the D/A reference voltage source your device use is external reference, the valid range is -10 to +10.

Syntax

object**.AO.Channels**(0).RefVoltage [=Single]

Data Type

Single

AO.Channels(0).Buffer1 Property
AO.Channels(1).Buffer1 Property

This property set up the buffer for continuous analog output operation. A buffer data or a array of buffer data, data type is integer.

Syntax

object**.AO.Channels**(0).Buffer1 [=Variant]

Remarks

You must assign this property before call StartContAO() method.
This property will be used when single or double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type

Variant(Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i
 Daq2204.AO.Channels(0).buffer1 = buffer1
 Daq2204.AO.Channels(0).Enable = True
 Daq2204.AO.StartContAO
```

AO.Channels(0).Buffer2 Property
AO.Channels(1).Buffer2 Property

This property set up the buffer for continuous analog output operation. A buffer data or a array of buffer data, data type is integer. This property only available when double buffer mode.

Syntax

object.**AO.Channels**(0).Buffer2 [=Variant]

Remarks

You must assign this property before call StartContAO() method.
This property will be used when double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type

Variant(Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i

 For i = 0 To 4095
      If i < 2048 Then
           buffer2(i) = (&H800 + i Mod 2048) And &HFFF
      Else
           buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
      End If
 Next i

 Daq2204.AO.Channels(0).buffer1 = buffer1
 Daq2204.AO.Channels(0).buffer2 = buffer2
 Daq2204.AO.DoubleBufferMode = True
 Daq2204.AO.Channels(0).Enable = True
 Daq2204.AO.StartContAO
```

SSI.ADCONV Property

Connect / Disconnect a SSI_ADCONV device signal to the specified SSI bus trigger line.

Syntax

object.**SSI.ADCONV** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger l ine
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.ADTRIG Property

Connect / Disconnect a SSI_ADTRIG device signal to the specified SSI bus trigger line.

Syntax

object.**SSI. ADTRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.DATRIG Property

Connect / Disconnect a SSI_DATRIG device signal to the specified SSI bus trigger line.

Syntax

object.**SSI.DATRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.DAWR Property

Connect / Disconnect a SSI_DAWR device signal to the specified SSI bus trigger line.

Syntax

object.**SSI.DAWR** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.TIMEBASE Property

Connect / Disconnect a SSI_TIMEBASE device signal to the specified SSI bus trigger line.

Syntax

object.**SSI.TIMEBASE** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

Open Method

Syntax

Function object**.Open** ([ErrMsgBox As Variant]) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

[ErrMsgBox As Variant]

Boolean type.
True: It will popup error message dialog box when operation error.
False: It will fire DAQError event instead of popping up dialog when operation error.

Remarks

This method will be used when the OpenMode property is Manual.

Note

In VC++, *ErrMsgBox* is a VARIANT of VT_I2.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

ShowPropertyPages Method

This method will show propertypages of ActiveX Control.

Syntax

Function object***.ShowPropertyPages***() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AboutBox Method

This method will show About ADLINK dialog box.

Syntax

Function object**.AboutBox**( ) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DIO.ReadDIPort Method

Syntax

Function object**.DIO.ReadDIPort** (port As Integer, value as Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Returns the digital data read from the specified port. The returned value is 8 -bit data.

Remarks

You can read data from the digital input port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.ReadDILine Method

Syntax

Function object**.DIO.ReadDILine** (port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

The digital line to be read. The valid value is 0 through 7

value As Variant

Returns the digital logic state, 0 or 1, of th e specified line.

Remarks

You can read data from the digital input port.

Note

In VC++, value is a VARIANT of VT_I2.

DIO.WriteDOPort Method

Syntax

Function object**.DIO.WriteDOPort** (port As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port

Value	Constant	Description
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value as Variant

8-bit data that will be written to the digital output port.

Remarks

Users can write data to the digital output port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.WriteDOLine Method

Syntax

Function object**.DIO.WriteDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

Selects one line number from the indicated port: from 0 to 7 (8-bit port).

value As Variant

Sets 0 or 1 to the indicated line.

Note

In VC++, value is a VARIANT of VT_I2.

DIO.ReadBackDOPort Method

Reads back data from the indicated digital output port.

Syntax

Function object**.DIO.ReadBackDOPort** (port As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Data that is read back from the indicated port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.ReadBackDOLine Method

Reads back data from the indicated digital output line of the indicated digital output port.

Syntax

Function object.**DIO.ReadBackDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

Selects one line number from the indicated port: from 0 to 7 (8-bit port).

value As Variant

Data that is read back from the indicated line.

Note

In VC++, value is a VARIANT of VT_I2.

GPTC.Counter0.Start Method
GPTC.Counter1.Start Method

Start counter operation with the specified mode.

Syntax

Function object.**GPTC.Counter0.Start**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Remarks

You can start the indicated counter to operate in the specified mode.

GPTC.Counter0.Stop Method
GPTC.Counter1.Stop Method

Stop counter operation.

Syntax

Function object.**GPTC.Counter0.Stop**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

GPTC.Counter0.Reset Method
GPTC.Counter1.Reset Method

Halts the specified general -purpose timer/counter operation and reload the initial value of the timer/counter.

Syntax

Function object.**GPTC.Counter0.Reset**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

GPTC.Counter0.ReadStatus Method
GPTC.Counter1.ReadStatus Method

Reads the latched GPTC status of the genera l-purpose counter from the GPTC status register.

Syntax

Function object.**GPTC.Counter0.ReadStatus**(Clock *As Integer, Output As Integer, Gate As Integer, Updown As Integer,*) As Boolean

Return Value

True if the function is successful; otherwise False.

AI.MuxScanSetup Method

stores *channels*, *gain and reference ground* in the Channel -Gain Queue for a scanned data acquisition operation. The method uses this memory table during scanning operations `AI.StartContAI()` to automatically sequence through an arbitrary set of analog input channels and to allow gains to automatically change during scanning .

Syntax

Function object.**MuxScanSetup** (ChannelArray As Variant, AdRangeArray As Variant , refGndArray As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

ChannelArray As Variant
Array of analog input channel numbers.
numbers in channel must be within 0 and 63.

AdRangeArray As Variant
An integer array contains the analog input range.

Value	Constant	Description
1	AD_B_10_V	Bipolar -10V to +10V
2	AD_B_5_V	Bipolar -5V to +5V
3	AD_B_2_5_V	Bipolar -2.5V to +2.5V
23	AD_B_2_V	Bipolar -2V to +2V
4	AD_B_1_25_V	Bipolar -1.25V to +1.25V
10	AD_B_1_V	Bipolar -1V to +1V
7	AD_B_0_5_V	Bipolar -0.5V to +0.5V
24	AD_B_0_25_V	Bipolar -0.25V to +0.25V
25	AD_B_0_2_V	Bipolar -0.2V to +0.2V
8	AD_B_0_05_V	Bipolar -0.05V to +0.05V
15	AD_U_10_V	Unipolar 0 to +10V
16	AD_U_5_V	Unipolar 0 to +5V
26	AD_U_4_V	Unipolar 0 to +4V
17	AD_U_2_5_V	Unipolar 0 to +2.5V

Value	Constant	Description
27	AD_U_2_V	Unipolar 0 to +2V
19	AD_U_1_V	Unipolar 0 to +1V
28	AD_U_0_5_V	Unipolar 0 to +0.5V
29	AD_U_0_4_V	Unipolar 0 to +0.4V
20	AD_U_0_1_V	Unipolar 0 to +0.1V

refGndArray As Variant
An integer array contains the reference ground

0	AI_RSE	Referenced single ended mode (64chs common to ground system on board)
256	AI_DIFF	Differential mode
512	AI_NRSE	Non-referenced single ended mode (64chscommon to AISENSE pin)

Note

In VC++, ***ChannelArray*** is a VARIANT of VT_ARRAY | VT_I4 , *AdRangeArray* is a VARIANT of VT_ARRAY | VT_I4 , *refGndArray* is a VARIANT of VT_ARRAY | VT_I4

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Dim nChannelArray(0 To 1) As Integer
Dim nRangeArray(0 To 1) As Integer
Dim nRefGndArray(0 To 1) As Integer

nChannelArray(0) = 5
nRangeArray(0) = AD_B_10_V
nRefGndArray(0) = AI_RSE
nChannelArray(1) = 8
nRangeArray(1) = AD_B_2_5_V
nRefGndArray(1) = AI_RSE

vChannel = nChannelArray
vRange = nRangeArray
vRefGnd = nRefGndArray

Daq2204.AI.MuxScanSetup vChannel, vRange, vRefGnd
```

AI.StartContAI Method

This method performs continuous A/D conversions on the specified an alog input channels at a rate as close to the rate you specified. This method takes advantage of the DAQ-2000 channel-gain that can be set separately for each channel to perform multi-channel/gain analog input. This method will fire AiComplete or AiHalfRea dy event depends on AI.DoubleBufferMode property.

Syntax

Function object**.AI.StartContAI**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Remarks

You can use this method to start the DMA analog input function. If the AI.StreamToFile property is True then the DMA data will be written to the file specified by AI.FileName. Otherwise, the AI.FileName property will be ignored.

The data file is written in binary format, with the lower byte first (little endian). Data type is “Binary codes with miscellaneous data”. DAQBench provides a convenient tool DAQCvt to convert the binary file to the file format read easily. See DAQBench User’s Guide for the usage of the utility. If you want to handle the data by yourself, please refer to Appendix D Data File Format for the file structure.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Dim nChannelArray(0 To 1) As Integer
Dim nRangeArray(0 To 1) As Integer
Dim nRefGndArray(0 To 1) As Integer

nChannelArray(0) = 5
nRangeArray(0) = AD_B_10_V
nRefGndArray(0) = AI_RSE
nChannelArray(1) = 8
nRangeArray(1) = AD_B_2_5_V
nRefGndArray(1) = AI_RSE

vChannel = nChannelArray
vRange = nRangeArray
vRefGnd = nRefGndArray

Daq2204.AI.MuxScanSetup vChannel, vRange, vRefGnd

Daq2204.AI.StartContAI

Private Sub Daq2204_AiComplete(ScaledData As Variant, BinaryCodes As Variant)
    ' Get Data in    ScaledData
End Sub

Private Sub Daq2204_AiHalfReady(ScaledData As Variant, BinaryCodes As Variant)
     ' Get Data in    ScaledData
End Sub
```

AI.StopContAI Method

You can use this method to force stop DMA analog input.

Syntax

Function object**.AI.StopContAI** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments
None

AI.ReadChannels Method

This method performs a software triggered A/D conversion (analog input) on analog input channels and returns the values converted.

Syntax
Function object.**AI.ReadChannels**(Buffer As Variant) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
Buffer As Variant

An integer array to contain the acquired data. Please refer to Appendix C AD Data Format for the data format in the Buffer.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
    Dim nChannelArray(0 To 1) As Integer
    Dim nRangeArray(0 To 1) As Integer
    Dim nRefGndArray(0 To 1) As Integer

    nChannelArray(0) = 5
    nRangeArray(0) = AD_B_10_V
    nRefGndArray(0) = AI_RSE
    nChannelArray(1) = 8
    nRangeArray(1) = AD_B_2_5_V
    nRefGndArray(1) = AI_RSE

    vChannel = nChannelArray
    vRange = nRangeArray
    vRefGnd = nRefGndArray

   Daq2204.AI.MuxScanSetup vChannel, vRange, vRefGnd

Private Sub Timer1_Timer()
    Dim vBuffer As Variant
    Daq2204.AI.ReadChannels vBuffer
    ' Get Data in    vBuffer
End Sub
```

AO.StartContAO Method

This method performs continuous D/A conversions on the specified analog output channel at a rate as close to the rate you specified.
This method will fire AoComplete or AoBufferReady event depends on AO.DoubleBufferMode property.

Syntax
Function object.**AO.StartContAO**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example
```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i

 For i = 0 To 4095
      If i < 2048 Then
           buffer2(i) = (&H800 + i Mod 2048) And &HFFF
      Else
           buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
      End If
 Next i

 Daq2204.AO.Channels(0).buffer1 = buffer1
 Daq2204.AO.Channels(0).buffer2 = buffer2
 Daq2204.AO.DoubleBufferMode = True
 Daq2204.AO.Channels(0).Enable = True
 Daq2204.AO.StartContAO
```

AO.StopContAO Method
You can use this method to force stop DMA analog output.

Syntax
Function object.**AO.StopContAO** () As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

AO.WriteChannel Method
Accepts a voltage value (or a current value), scales it to the proper binary value and writes a binary value to the specified analog output channel.

Syntax
Function object.**AO.WriteChannel**(Channel as Integer, Voltage as Single) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
Channel as Integer

The analog output channel number.

Range: 0 or 1 for DAQ -2204

Voltage as Single

The value to be scaled and written to the analog output channel. The range of voltages depends on the type of device, on the output polarity, and on the voltage reference (external or internal).

CALIBRATION.AutoCalibration Method

Uses this method to calibrate your DAQ -2000 device. When the method is called, the device goes into a self-calibration cycle. The method does not return until the self -calibration is completed.

Syntax
Function object.**CALIBRATION.AutoCalibration** () As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

CALIBRATION.DisplayErrors Method

Uses this method to fire AcquireADError and AcquireDAError events. Through those events user can identification DAQ-2000 device current status.

Syntax
Function object.**CALIBRATION.DisplayErrors** () As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Example
Daq2204.CALIBRATION.DisplayErrors

```
Private Sub Daq2204_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "BiPolar"
    End If
    strMsg = "AD channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub

Private Sub Daq2204_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
```

```
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub
```

CALIBRATION.Load Method

Load calibration constants from the specified bank of EEPROM.

Syntax

Function object**.CALIBRATION.Load**(BankOfEEPROM As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

CALIBRATION.Save Method

Save calibration constants to the specified bank of EEPROM.

Syntax

Function object**.CALIBRATION.Save**(BankOfEEPROM As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

SSI.ClearAll Method

Disconnects all of the device signals from the SSI bus trigger lines.

Syntax

Function object**.ClearAll** As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DAQError Event

Syntax

sub ControlName_DAQError (ErrString As String)

Arguments

ErrString As String
The string of error reasion

Remarks

This event will occur when some error occur in control

AiComplete Event

Syntax

sub ControlName_AiComplete(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant

The analog input data array that have been translated to the engineering data (voltage) according to AI.Channels(n).Range property.

BinaryCodes As Variant

The analog input data array with digital format.

Remarks

This event occurs when continuous analog input function is completed. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting.*"BinaryCodes without channel only".*
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AiHalfReady Event

Syntax

sub ControlName_AiHalfReady(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant

The analog input data array that have been translated to the engineering data (voltage) according to AIRange property.

BinaryCodes As Variant

The analog input data array with digital format.

Remarks

This event occurs when one half-buffer of the circular buffer is full at continuous analog input operation. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting. "*BinaryCodes without channel only*"
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AoComplete Event

Syntax
sub ControlName _AoComplete( )

Arguments
None

Remarks
This event occurs when continuous anal og output function is completed.

AoBufferReady Event

Syntax
sub ControlName_AoBufferReady(BufferIndex as Integer)

Arguments
BufferIndex as Integer, The index of the buffer just output.

Remarks
This event occurs when enable the AO.DoubleBufferMode, If User want to dynamic change the output pattern, process it when receive AoBufferReady event.

Example
```
'This sample code show that how to output four buffers (pattern) in one channel.
'You can modify this code for dynamic changes pattern.

Dim varArray(0 To 3) As Variant
Dim nCounter As Integer

Private Sub AO_Click()

    Dim buffer0(0 To 4095) As Integer
    Dim buffer1(0 To 4095) As Integer
    Dim buffer2(0 To 4095) As Integer
    Dim buffer3(0 To 4095) As Integer

    Dim i As Double

    For i = 0 To 4095
        buffer0(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer1(i) = (&H800 + i Mod 2048) And &HFFF
        Else
            buffer1(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        End If
    Next i
    For i = 0 To 4095
        buffer2(i) = (Cos(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer3(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        Else
            buffer3(i) = (&H800 + i Mod 2048) And &HFFF
        End If
    Next i

    varArray(0) = buffer0
    varArray(1) = buffer1
```

```
    varArray(2) = buffer2
    varArray(3) = buffer3

    Daq2204.AO.Channels(0).buffer1 = varArray(0)
    Daq2204.AO.Channels(0).buffer2 = varArray(1)
    nCounter = 1
    Daq2204.AO.Channels(0).Enable = True

    Daq2204.AO.DoubleBufferMode = True
    Daq2204.AO.StartContAO
End Sub

Private Sub Daq2204_AoBufferReady(ByVal BufferIndex As Integer)

nCounter = nCounter + 1
nCounter = nCounter Mod 4

If BufferIndex = 1 Then
    ' It can change buffer1 in here
    Daq2204.AO.Channels(0).buffer1 = varArray(nCounter)
End If

If BufferIndex = 2 Then
    ' It can change buffer2 in here
    Daq2204.AO.Channels(0).buffer2 = varArray(nCounter)
End If

End Sub
```

Acquire22XXADError Event

Acquires the offset and gain errors.

Syntax

Sub ***ControlName*** _Acquire22XXADError(ByVal gain_err As Double, ByVal bioffset_err As Double, ByVal unioffset_err As Double, ByVal hg_bios_err As Double)

Arguments

gain_err As Double ,
The gain error of the ADC.

bioffset_err As Double,
The offset error of the ADC in bipolar mode.

unioffset_err As Double
The offset error of the ADC in unipolar mode.

hg_bios_err As Double
The high-gain offset error of the ADC in bipolar mode.

Example

Daq2204.CALIBRATION.DisplayErrors

```
Private Sub Daq2204_Acquire22XXADError(ByVal gain_err As Double, ByVal bioffset_err As Double,
ByVal unioffset_err As Double, ByVal hg_bios_err As Double)
    strMsg = "AD Gain error:" & Format(gain_err, "#0.#####")
    List1.AddItem (strMsg)
    strMsg = "Bio offset error:" & Format(bioffset_err, "#0.#####")
    List1.AddItem (strMsg)
    strMsg = "Uni offset error:" & Format(unioffset_err, "#0.#####")
    List1.AddItem (strMsg)
    strMsg = "Hign Gain bipolar Offset error:" & Format(hg_bios_err, "#0.#####")
```

```
    List1.AddItem (strMsg)
End Sub
```

AcquireDAError Event

Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

Syntax

sub *ControlName*_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)

Arguments

channel as Integer,
Indicate channel id.

polarity *as Integer,*
Indicate polarity

Value	Description
0	Unipolar
1	Bipolar

gain_err as Double
Indicate gain error

offset_err as Double
Indicate offset error

Example

Daq2204.CALIBRATION.DisplayErrors

```
Private Sub Daq2204_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####") & " Offset error:" & Format(offset_err, "#0.#####")
  List1.AddItem (strMsg)
  List1.Refresh
  End Sub
```

Daq2205 ActiveX Control

The Daq2205 ActiveX control is a software component that provides the interface for users to control PCI-2205 card.

Daq2205 ActiveX Control Overview			
Type (Group)	**Property**	**Method**	**Event**
General	DASKCardType	Open	DAQError
	CardNumber	AboutBox	
	OpenMode	ShowPropertyPages	
	DaskCardID		
DIO	P1Adir	ReadBackDOLine	
	P1Bdir	ReadBackDOPort	
	P1ClowerDir	ReadDILine	
	P1CupperDir	ReadDIPort	
		WriteDOLine	
		WriteDOPort	
GPTC	Mode	Start	
	ClockSource	Stop	
	ClockPolarity	Reset	
	GateSource	ReadStatus	
	GatePolarity		
	UpDownSource		
	UpDownPolarity		
	OutputPolarity		
	IntGATE		
	IntUpDnCTR		
	DelayCount		
	DurationCount		
	OutputValue		
AI	NumOfScan	StartContAI	AiComplete
	ClockSource	StopContAI	AiHalfReady
	ScanInterval	ReadChannels	
	SamplingInterval	MuxScanSetup	
	ConversionSource		
	TriggerMode		
	TriggerSource		
	DelaySource		
	ReTriggerModeEnable		
	McounterEnable		
	PostTriggerCount		
	DelayCount		

Daq2205 ActiveX Control Overview			
	Mcount		
	ReTriggerCount		
	ExtTrigPolarity		
	ReturnType		
	DoubleBufferMode		
	StreamToFile		
	FileName		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
AO	CHUI	StartContAO	AoComplete
	DAWRSource	StopContAO	AoBufferReady
	TriggerSource	WriteChannel	
	TriggerMode		
	Delay1Source		
	Delay2Source		
	ExtTrigPolarity		
	ReTriggerCount		
	Delay1Count		
	Delay2Count		
	ClockSource		
	ReTriggerModeEnable		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
	DoubleBufferMode		
	Iterations		
	Definite		
	StopMode		
	Channels(n).Enable		
	Channels(n).IntOrExtref		
	Channels(n).OutputPolarity		
	Channels(n).RefVoltage		
	Channels(n).Buffer1		
	Channels(n).Buffer2		
CALIBRATION	BankTemperature	AutoCalibration	Acquire22XXADError
	BankDate	Load	AcquireDAError

Daq2205 ActiveX Control Overview			
	CurrentTemperature	Save	
	CurrentDate	DisplayErrors	
SSI	TIMEBASE	ClearAll	
	ADCONV		
	DAWR		
	ADTRIG		
	DATRIG		

DASKCardType Property

Return a value that determines the card type. It is always DAQ_2205 in DAQ-2205 device.

.

Syntax

[Integer] =object**.CardType**

Remarks

Always return DAQ_2205 (for DAQ -2205 device)

Settings

Value	Constant	Description
2	DAQ_2205	For DAQ-2004 Device

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Data Type

Integer

CardNumber Property

The sequence number of the card with ***the same card type*** plugged in the PCI slot.
The card sequence number setting is according to the PCI slot sequence in the mainboard.
The first card (in the most prior slot) is with CardNumber=0. For example,
If there are two DAQ -2205 cards plugged on your PC, the DAQ -2205 card in the prior slot should be registered with CardNumber =0, and the other one with CardNumber =1.

.

Syntax

object**.CardNumber** [= short]

Remarks

This property will be used when Initializes the hardware states of a DAQ -2K data acquisition card.

Data Type

Integer

OpenMode Property

Return/Set a value that determines the mode of opening device.

Syntax

object**.OpenMode** [= short]

Settings

Value	Display	Description
0	Automatic	D2K-OCX will initialize by itself. Automatically open device when the control was created
1	Manual	Call object**.Open**() method to initialized D2K-OCX(open device)
2	Demonstration	D2K-OCX will provide software DAQ data to simulating hardware

Value	Display	Description
		drivers.

Data Type
Integer

DaskCardID Property

Returns a value that determines the D2K_Register_Card() returns value,
the DaskCardID is a numeric card ID that will be used by other D2K -DASK library functions.

Syntax
[short] =object.**DaskCardID**

Remarks
The range of card id is between 0 and 31.
This property will be used when combine D2K-DASK and D2K-OCX two module in one program.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example
This sample will demonstration If user want to check AI completed by itself.

```
' OCX to starting continuous AI
   Dim nDaskID as Integer
   Daq2205.Open(TRUE)
   nDaskID = Daq2205.DaskCardID

Daq2205.AI.StartContAI

' Check AI Completed by DASK    API
Dim Stopped    As Byte
Dim AccessCnt As Long

Do While True
     D2K_AI_AsyncCheck nDaskID, Stopped, AccessCnt
     If Stopped = 1 Then
          Exit Do
     End If
Loop
MsgBox "AI Complete"
```

DIO.P1Adir Property

Return/Set a value that determines P1A port direction.

Syntax
object.**DIO.P1ADir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1Bdir Property
Return/Set a value that determines P1B port direction.

Syntax
object.**DIO.P1BDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CLowerdir Property
Return/Set a value that determines P1C lower port direction.

Syntax
object.**DIO.P1CLowerDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CUpperdir Property

Return/Set a value that determines P1C upper port direction.

Syntax

object.**DIO.P1CUpperDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.Mode Property

Return/Set a value that determines the Timer/Counter mode.

Syntax

object.**GPTC.Counter0.Mode** [= Integer]

Settings

Value	Constant	Description
1	SimpleGatedEventCNT	
2	SinglePeriodMSR	
3	SinglePulseWidthMSR	
4	SingleGatedPulseGen	
5	SingleTrigPulseGen	
6	RetrigSinglePulseGen	
7	SingleTrigContPulseGen	
8	ContGatedPulseGen	

Remarks

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic

Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockSource Property
GPTC.Counter1.ClockSource Property

Return/Set a value that determines the Timer/Counter Source.

Syntax

object.**GPTC.Counter0.ClockSource** [= Integer]
object.**GPTC.Counter1.ClockSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKSRC_INT	internal time base
2	GPTC_CLKSRC_EXT	external time base from GPTC0_SRC or GPTC1_SRC pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockPolarity Property
GPTC.Counter1.ClockPolarity Property

Return/Set a value that determines the Timer/Counter clock polarity

Syntax

object.**GPTC.Counter0.ClockPolarity** [= Integer]
object.**GPTC.Counter1.ClockPolarity** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKEN_LACTIVE	Low active
2	GPTC_CLKEN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GateSource Property
GPTC.Counter1.GateSource Property

Return/Set a value that determines the Timer/Counter gate source

Syntax

object.**GPTC.Counter0.GateSource** [= Integer]
object.**GPTC.Counter1.GateSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_GATESRC_INT	gate is controlled by software
4	GPTC_GATESRC_EXT	gate is controlled by GPTC0_GATE or GPTC1_GATE pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GatePolarity Property
GPTC.Counter1.GatePolarity Property

Return/Set a value that determines the Timer/Counter gate polarity

Syntax

object.**GPTC.Counter0.GatePolarity** [= Integer]
object.**GPTC.Counter1.GatePolarity** [= Integer]

Settings

Value	Constant	Description
2	GPTC_GATE_LACTIVE	Low active
0	GPTC_GATE_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownSource Property
GPTC.Counter1.UpDownSource Property

Return/Set a value that determines the Timer/Counter UpDown Source

Syntax

object.**GPTC.Counter0.UpDownSource** [= Integer]
object.**GPTC.Counter1.UpDownSource** [= Integer]

Settings

Value	Constant	Description
0	GPTC_UPDOWN_SEL_INT	Up/Down controlled by software
16	GPTC_UPDOWN_SEL_EXT	Up/Down controlled by GPTC0_UPDOWN orGPTC1_UPDOWN pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownPolarity Property
GPTC.Counter1.UpDownPolarity Property

Return/Set a value that determines the Timer/Counter UpDown Polarity

Syntax

object.**GPTC.Counter0.UpDownPolarity** [= Integer]
object.**GPTC.Counter1.UpDownPolarity** [= Integer]

Settings

Value	Constant	Description
4	GPTC_UPDOWN_LACTIVE	Low active
0	GPTC_UPDOWN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi

Unit: D2kDask.pas

GPTC.Counter0.OutputPolarity Property
GPTC.Counter1.OutputPolarity Property

Return/Set a value that determines the Timer/Counter Output Polarity

Syntax

object.**GPTC.Counter0.OutputPolarity** [= Integer]
object.**GPTC.Counter1.OutputPolarity** [= Integer]

Settings

Value	Constant	Description
8	GPTC_OUTPUT_LACTIVE	Low active
0	GPTC_OUTPUT_HACTIVE	High active

Please refer to the hardware manual for the mo de description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.IntGATE Property
GPTC.Counter1.IntGATE Property

Return/Set a value that determines the Timer/Counter Internal gate initialized status

Syntax

object.**GPTC.Counter0.IntGATE** [= Boolean]
object.**GPTC.Counter1.IntGATE** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type

Boolean

GPTC.Counter0.IntUpDnCTR Property
GPTC.Counter1.IntUpDnCTR Property

Return/Set a value that determines the Timer/Counter internal updown counter initialized status.

Syntax

object.**GPTC.Counter0.IntUpDnCTR** [= Boolean]
object.**GPTC.Counter1.IntUpDnCTR** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type

Boolean

GPTC.Counter0.DelayCount Property
GPTC.Counter1.DelayCount Property

Return/Set a value that determines the Timer/Counter internal initial count of the GPTC or pulse delay. The counter value of load register 1 of timer/counter. The meaning for the value depends on the mode the timer /counter performs. For mode 1 to mode 3, the value is the initial count of the GPTC. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse delay.

Syntax

object.**GPTC.Counter0.DelayCount** [= Integer]
object.**GPTC.Counter1.DelayCount** [= Integer]

Data Type

Integer

GPTC.Counter0.DurationCount Property
GPTC.Counter1.DurationCount Property

Return/Set a value that determines the Timer/Counter pulse width when mode 4 to mode 8. The counter value of load register 2 of timer/counter. For mode 1 to mode 3, the value is not used. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse width.

Syntax

object.**GPTC.Counter0.DurationCount** [= Integer]
object.**GPTC.Counter1.DurationCount** [= Integer]

Data Type

Integer

GPTC.Counter0.OutputValue Property
GPTC.Counter1.OutputValue Property

Returns the counter value of the specified general -purpose timer/counter. Range: 0 through 65535

Syntax

[Integer=] object.**GPTC.Counter0.OutputValue**
[Integer=] object.**GPTC.Counter1.OutputValue**

Data Type

Integer

GPTC.CALIBRATION.BankTemperature Property

Returns a value that since user's last calibrated temperature in the EEPR ON Bank .

Syntax

object.**CALIBRATION.BankTemperature**(*[BankOfEEPROM as Integer]*) As Single

Data Type

Single

GPTC.CALIBRATION.BankDate Property

Returns a value that since user's last calibrated date in the EEPRON Bank .

Syntax

object.CALIBRATION.BankDate(*[BankOfEEPROM as Integer]*) As String

Data Type

String

GPTC.CALIBRATION.CurrentTemperature Property

Returns a value that determines the current temperature on card.

Syntax

[Single=] object**.CALIBRATION.CurrentTemperature**

Data Type

Single

GPTC.CALIBRATION.CurrentDate Property

Returns a value that determines the current date.

Syntax

[String=] object**.CALIBRATION.CurrentDate**

Data Type

String

AI.NumOfScan Property

If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, NumOf *Scan* is the size (in samples) allocated for each channel in the circular buffer. This value must be a multiple of 2.

Syntax

object**.AI.NumOfScan** [=Long]

Remarks

Non-double-buffer mode

This value multiply the total number of scan channels is the total number of A/D conversions to be performed.

Double-buffer-mode

This value multiply the total number of scan channels is the size (in sample) of the circular buffer.

Data Type
Long

AI.ClockSource Property
The clock source (Time Base) the device selected.

Syntax
object.**AI.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ScanInterval Property
The length of the scan interval (that is, the counter value betw een the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.
If the timer base is from *external*, the valid range of the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of the value is 80 through 16777215

Syntax
object.**AI.ScanInterval** [=Long]

Data Type
Long

AI.SamplingInterval Property
The length of the sample interval (that is, the counter value between each A/D conversion within a scan sequence). The A/D conversion rate will be *TimeBase/SampIntrv*. The value of *TimeBase(AI.ClockSource)* depends on the card type.
If the timer base is from *external*, the valid range of the value is 2 through 65535.
If the timer base is *Internal timer*, the valid range of the value is as follows:
DAQ-2205 : 80 through 65535

Syntax
object.**AI.SamplingInterval** [=Long]

Data Type
Long

AI.ConversionSource Property

The A/D Conversion Source Selection.

Syntax

object.**AI.ConversionSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_ADCONVSRC_Int	Internal timer
4	DAQ2K_AI_ADCONVSRC_AFI0	From AFI0 pin
8	DAQ2K_AI_ADCONVSRC_SSI	From SSI source

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerMode Property

The Trigger Mode Selection.

Syntax

object.**AI.TriggerMode** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_TRGMOD_POST	Post Trigger Mode
8	DAQ2K_AI_TRGMOD_DELAY	Delay Trigger Mode
16	DAQ2K_AI_TRGMOD_PRE	Pre-Trigger Mode
24	DAQ2K_AI_TRGMOD_MIDL	Middle-Trigger Mode

Remarks

Please refer to the hardware manual for the trigger mode description.

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerSource Property

The trigger source selection.

Syntax
object.**AI.TriggerSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_TRGSRC_SOFT	Software
1	DAQ2K_AI_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_AI_TRGSRC_ExtD:	From external digital trigger pin
3	DAQ2K_AI_TRSRC_SSI	From SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.DelaySource Prop erty
The delay source selection.

Syntax
object.**AI. DelaySource** [=Short]

Settings

Value	Constant	Description
256	DAQ2K_AI_Dly1InSamples	Delay in samples
0	DAQ2K_AI_Dly1InTimebase	Delay in time base

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReTriggerModeEnable Property
Re-trigger in an acquisition is enabled

Syntax
object.**AI.ReTriggerModeEnable** [=Boolean]

Settings

Value	Constant	Description

Value	Constant	Description
0	False	ReTriggerMode is disabled.
1	True	ReTriggerMode is enabled.

Data Type

Boolean

AI.MCounterEnable Property

This constant is only valid for Pre -trigger and Middle trigger mode
Mcounter is enabled and then the trigger signal is ignore before M termi nal count is reached.

Syntax

object**.AI.MCounterEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	MCounter is disabled.
1	True	Mcounter is enabled.

Data Type

Boolean

AI.PostTriggerCount Property

This constant is only valid for Middle trigger mode,
The PostTriggerCount indicates the number of data will be accessed after a specific trigger event.

Syntax

object**.AI.PostTriggerCount** [=Long]

Data Type

Long

AI.DelayCount Property

This constant is only valid for Delay trigger mode,
The DelayCount indicates the number of data or timer ticks will be ignored after a specific trigger event.

Syntax

object**.AI.DelayCount** [=Long]

Data Type

Long

AI.MCount Property

The counter value of MCounter . This argument is only valid for Pr etrigger and Middle trigger mode.

Syntax

object**.AI.MCount** [=Long]

Data Type
Long

AI.ReTriggerCount Property
The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax
object.**AI.ReTriggerCount** [=Long]

Data Type
Long

AI.ExtTrigPolarity Property
External Digital Trigger Polarity.

Syntax
object.**AI.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_AI_TrgPositive	Trigger positive edge active
4096	DAQ2K_AI_TrgNegative	Trigger negative edge active

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReturnType Property
Return/Set a value that determines the return data type of analog input when AiComplete or AiHalfReady event would occur.

Syntax
object.**AI.ReturnType** [=Integer]

Settings

Value	Constant	Description
0		Scaled data only
1		Binary codes without channel only
2		Scaled data and binary codes without channel
3		No data return

Data Type
Integer

AI.DoubleBufferMode Property

Enables or disables double -buffered data acquisition mode.

Syntax

object.**AI.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type

Boolean

AI.StreamToFile Property

Return/Set a value that determines if the control is enabled the function of streaming data to disk file. This argument is only valid for Delay trigger

Syntax

object.**AI.StreamToFile** [=Boolean]

Settings

Value	Constant	Description
0	False	Disable the function of streaming data to disk file.
1	True	Enable the function of streaming data to disk file

Data Type

Boolean

AI.FileName Property

FileName specified the fil e name of streaming data to disk. This argument is only valid for AI.StreamToFile is Enable.

Syntax

object.**AI.FileName** [=String]

Data Type

String

AI.AIOAnalogTrigCtrl Property

Return/Set a value that determines the t rigger source selection

Syntax

object.**AI.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	ADCATRIG	The first AI channel in the channel-gain queue
1	EXTATRIG	From external analog trigger pin

Data Type
Integer

AI.AIOTrigCondition Property
Return/Set a value that determines the t rigger condition selection

Syntax
object.**AI.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above -High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks
Please refer to the hardware manual for the more description.

Data Type
Integer

AI.AIOHLevel Property
The High value set ting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax
object.**AI.AIOHLevel** [=Long]

Remarks
***For example*:**
If the trigger voltage is ±10V, the relationship between the v alue of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	Trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AI.AIOLLevel Property
The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax
object.**AI.AIOLLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AO.CHUI Property

The length of the Channel Update interval (that is, t he counter value between the initiation of each update sequence).
If the timer base is from *external*, the valid range of the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of the value is as follows:
Range: 40 through 16 777215

Syntax

object.**AO.CHUI** [=Long]

Data Type

Long

AO.DAWRSource Property

Return/Set a value that determines the D/A R/W Source Selection

Syntax

object.**AO.DAWRSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_WRSRC_Int	Internal timer
1	DAQ2K_DA_WRSRC_AFI0	From AFI0 pin
2	DAQ2K_DA_WRSRC_SSI	From SSI source

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerSource Property

Return/Set a value that determines the trigger source selection

Syntax
object.**AO.TriggerSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGSRC_SOFT	Software
1	DAQ2K_DA_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_DA_TRGSRC_ExtD	From external digital trigger pin
3	DAQ2K_DA_TRSRC_SSI	From SSI source

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerMode Property
Return/Set a value that determines the trigger mode selection

Syntax
object.**AO.TriggerMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGMOD_POST	Post Trigger Mode
1	DAQ2K_DA_TRGMOD_DELAY	Delay Trigger Mode

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay1Source Property
Return/Set a value that determines the delay1 source selection

Syntax
object.**AO.Delay1Source** [=Integer]

Settings

Value	Constant	Description

Value	Constant	Description
64	DAQ2K_DA_Dly1InUI	Delay in samples
0	DAQ2K_DA_Dly1InTimebase	Delay in time base

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay2Source Property
Return/Set a v alue that determines the delay2 source selection

Syntax
object.**AO.Delay2Source** [=Integer]

Settings

Value	Constant	Description
128	DAQ2K_DA_Dly2InUI	Delay in samples
0	DAQ2K_DA_Dly2InTimebase	Delay in time base

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ExtTrigPolarity Property
Return/Set a value that determines the e xternal digital trigger polarity selection

Syntax
object.**AO.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TrgPositive	Trigger positive edge active
512	DAQ2K_DA_TrgNegative	Trigger negative edge active

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerCount Property

The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax

object.**AO.ReTriggerCount** [=Long]

Data Type

Long

AO.Delay1Count Propert y

The counter value of DLY1 Counter (the delay time after the trigger signal to the start of the waveform generation) . This argument is only valid for Delay trigger mode.

Syntax

object.**AO.Delay1Count** [=Long]

Data Type

Long

AO.Delay2Count Property

The counter value of DLY2 Counter (the Delay between two consecutive waveform generations).

Syntax

object.**AO.Delay2Count** [=Long]

Data Type

Long

AO.ClockSource Property

The clock source (Time Base) the device selected.

Syntax

object.**AO.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type

Short

Microsoft C/C++ and Borland C++

Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerModeEnable Property

Re-trigger in an acquisition is enabled

Syntax

object**.AO.ClockSource** [=Boolean]

Settings

Value	Constant	Description
0	False	ReTriggerMode is disabled
1	True	ReTriggerMode is enabled

Data Type

Boolean

AO.AIOAnalogTrigCtrl Property

Return/Set a value that determines the t rigger source selection

Syntax

object**.AO.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	ADCATRIG	The first AI channel in the channel-gain queue
1	EXTATRIG	From external analog trigger pin

Data Type

Integer

AO.AIOTrigCondition Property

Return/Set a value that determines the t rigger condition selection

Syntax

object**.AO.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above-High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks

Please refer to the hardware manual for the more description.

Data Type
Integer

AO.AIOHLevel Property

The High value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax
object.**AO.AIOHLevel** [=Long]

Remarks
***For example*:**
If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AO.AIOLLevel Property

The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax
object.**AO.AIOLLevel** [=Long]

Remarks
***For example*:**
If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AO.DoubleBufferMode Property

Enables or disables double -buffered data acquisition mode.

Syntax
object.**AO.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type

Boolean

AO.Iterations Property

The times of number of the data in the buffer to output to the port. a value of zero is not allowed.

Syntax

object.**AO.Iterations** [=Long]

Data Type

Long

AO.Definite Property

Waveform generation proceeds definite or indefinitely. If double -buffered mode is enabled, this parameter is of no use.

Syntax

object.**AO.Definite** [=Boolean]

Settings

Value	Constant	Description
0	False	Indefinitely
1	True	Definite

Data Type

Boolean

AO.StopMode Property

Return/Set a value that determines DA transfer termination mode selected.

Syntax

object.**AO. StopMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TerminateImmediate	Software terminate the DA continuous operation immediately
1	DAQ2K_DA_TerminateUC	Software terminate the DA continuous operation on next update counter terminal count
2	DAQ2K_DA_TerminateIC	Software terminate the DA continuous operation on iteration count

Remarks

Please refer to the hardware manual for the more description.

Data Type

Integer

AO.Channels(0).Enable Property
AO.Channels(1).Enable Property

DAQ-2000 output channel that can be set separately for each chan nel to perform multi-channel analog input, The parameter 0~1 is channel id.

Syntax

object.**AO.Channels**(0).Enable [=Boolean]

Data Type

Boolean

AO.Channels(0).OutputPolarity Property
AO.Channels(1).OutputPolarity Property

Return/Set a value that deter mines polarity (unipolar or bipolar) of the output channel.

Syntax

object.**AO.Channels**(0).OutputPolarity [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_UniPolar	Unipolar
1	DAQ2K_DA_BiPolar	Bipolar

Data Type

Integer

AO.Channels(0).IntOrExtref Property
AO.Channels(1).IntOrExtref Property

Return/Set a value that determines DA reference voltage source of the output channel.

Syntax

object.**AO.Channels**(0). IntOrExtref [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_Int_REF	internal reference
1	DAQ2K_DA_Ext_REF	external reference

Data Type

Integer

AO.Channels(0).RefVoltage Property
AO.Channels(1).RefVoltage Property

If the D/A reference voltage source your device use is internal reference, the valid values is 10.
If the D/A reference voltage source your device use is external reference, the valid range is -10 to

+10.

Syntax
object.**AO.Channels**(0).RefVoltage [=Single]

Data Type
Single

AO.Channels(0).Buffer1 Property
AO.Channels(1).Buffer1 Property

This property set up the buffer for continuous analog output operation. A buffer data or a array of buffer data, data type is integer.

Syntax
object.**AO.Channels**(0).Buffer1 [=Variant]

Remarks
You must assign this property before call StartContAO() method.
This property will be used when single or double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type
Variant (Integer array)

Example
```
Dim buffer1(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i
 Daq2205.AO.Channels(0).buffer1 = buffer1
 Daq2205.AO.Channels(0).Enable = True
 Daq2205.AO.StartContAO
```

AO.Channels(0).Buffer2 Property
AO.Channels(1).Buffer2 Property

This property set up the buffer for continuous analog output operation.
A buffer data or a array of buffer data, data type is integer.
This property only available when double buffer mode.

Syntax
object.**AO.Channels**(0).Buffer2 [=Variant]

Remarks
You must assign this property before call StartContAO() method.
This property will be used when double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type
Variant(Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i

 For i = 0 To 4095
      If i < 2048 Then
           buffer2(i) = (&H800 + i Mod 2048) And &HFFF
      Else
           buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
      End If
 Next i

 Daq2205.AO.Channels(0).buffer1 = buffer1
 Daq2205.AO.Channels(0).buffer2 = buffer2
 Daq2205.AO.DoubleBufferMode = True
 Daq2205.AO.Channels(0).Enable = True
 Daq2205.AO.StartContAO
```

SSI.ADCONV Property
Connect / Disconnect a SSI_ADCONV device signal to the specified SSI bus trigger line.

Syntax
object**.SSI.ADCONV** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.ADTRIG Property
Connect / Disconnect a SSI_ADTRIG device signal to the specified SSI bus trigger line.

Syntax
object**.SSI. ADTRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.DATRIG Property

Connect / Disconnect a SSI_DATRIG device signal to the specified SSI bus trigger line.

Syntax
object.**SSI.DATRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.DAWR Property

Connect / Disconnect a SSI_DAWR device signal to the specified SSI bus trigger line.

Syntax
object.**SSI.DAWR** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.TIMEBASE Property

Connect / Disconnect a SSI_TIMEBASE device signal to the specified SSI bus trigger line.

Syntax
object.**SSI.TIMEBASE** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

DAQ_2205 Methods

Open Method

Syntax
Function object.**Open** ([ErrMsgBox As Variant]) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments

[ErrMsgBox As Variant]

Boolean type.
True: It will popup error message dialog box when operation error.
False: It will fire DAQError event instead of popping up dialog when operation error.

Remarks

This method will be used when the OpenMode property is Manual.

Note

In VC++, *ErrMsgBox* is a VARIANT of VT_I2.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

ShowPropertyPages Method

This method will show propertypages of ActiveX Control.

Syntax

Function object.***ShowPropertyPages***() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AboutBox Method

This method will show About ADLINK dialog box.

Syntax

Function object.**AboutBox**( ) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DIO.ReadDIPort Method

Syntax

Function object.**DIO.ReadDIPort** (port As Integer, value as Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Returns the digital data read from the specified port. The returned value is 8-bit data.

Remarks
You can read data from the digital input port.

Note
In VC++, value is a VARIANT of VT_I4.

DIO.ReadDILine Method

Syntax
Function object**.DIO.ReadDILine** (port As Integer, line As Integer, value As Variant) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

The digital line to be read. The valid value is 0 through 7

value As Variant

Returns the digital logic state, 0 or 1, of the specified line.

Remarks
You can read data from the digital input port.

Note
In VC++, value is a VARIANT of VT_I2.

DIO.WriteDOPort Method

Syntax
Function object**.DIO.WriteDOPort** (port As Integer, value As Variant) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value as Variant

8-bit data that will be written to the digital output port.

Remarks
Users can write data to the digital output port.

Note
In VC++, value is a VARIANT of VT_I4.

DIO.WriteDOLine Method

Syntax

Function object**.DIO.WriteDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

Selects one line number from the indicated port: from 0 to 7 (8-bit port).

value As Variant

Sets 0 or 1 to the indicated line.

Note

In VC++, value is a VARIANT of VT_I2.

DIO.ReadBackDOPort Method

Reads back data from the indicated digital output port.

Syntax

Function object**.DIO.ReadBackDOPort** (port As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Data that is read back from the indicated port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.ReadBackDOLine Method

Reads back data from the indicated digital output line of the indicated digital output port.

Syntax

Function object**.DIO.ReadBackDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer
Selects one line number from the indicated port: from 0 to 7 (8-bit port).
value As Variant
Data that is read back from the indicated line.

Note
In VC++, value is a VARIANT of VT_I2.

GPTC.Counter0.Start Method
GPTC.Counter1.Start Method
Start counter operation with the specified mode.

Syntax
Function object**.GPTC.Counter0.Start**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Remarks
You can start the indicated counter to operate in the specified mode.

GPTC.Counter0.Stop Method
GPTC.Counter1.Stop Method
Stop counter operation.

Syntax
Function object**.GPTC.Counter0.Stop**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

GPTC.Counter0.Reset Method
GPTC.Counter1.Reset Method
Halts the specified general -purpose timer/counter operation and reload the initial value of the timer/counter.

Syntax
Function object**.GPTC.Counter0.Reset**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

GPTC.Counter0.ReadStatus Method
GPTC.Counter1.ReadStatus Method

Reads the latched GPTC status of the general -purpose counter from the GPT C status register.

Syntax

Function object.**GPTC.Counter0.ReadStatus**(Clock *As Integer, Output As Integer, Gate As Integer, Updown As Integer*,) As Boolean

Return Value

True if the function is successful; otherwise False.

AI.MuxScanSetup Method

stores *channels*, *gain and reference ground* in the Channel -Gain Queue for a scanned data acquisition operation. The method uses this memory table during scanning operations `AI.StartContAI()` to automatically sequence through an arbitrary set of analog input channels and to allow gains to automatically change during scanning .

Syntax

Function object.**MuxScanSetup** (ChannelArray As Variant, AdRangeArray As Variant , refGndArray As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

ChannelArray As Variant

Array of analog input channel numbers.
numbers in channel must be within 0 and 63.

AdRangeArray As Variant

An integer array contains the analog input range.

Value	Constant	Description
1	AD_B_10_V	Bipolar -10V to +10V
2	AD_B_5_V	Bipolar -5V to +5V
3	AD_B_2_5_V	Bipolar -2.5V to +2.5V
4	AD_B_1_25_V	Bipolar -1.25V to +1.25V
15	AD_U_10_V	Unipolar 0 to +10V
16	AD_U_5_V	Unipolar 0 to +5V
17	AD_U_2_5_V	Unipolar 0 to +2.5V
18	AD_U_1_25_V	Unipolar 0 to +1.25V

refGndArray As Variant

An integer array contains the reference ground

Value	Constant	Description
0	AI_RSE	Referenced single ended mode (64chs common to ground system on board)
256	AI_DIFF	Differential mode
512	AI_NRSE	Non-referenced single ended mode (64chscommon to AISENSE pin)

Note

In VC++, ***ChannelArray*** is a VARIANT of VT_ARRAY | VT_I4 , *AdRangeArray* is a VARIANT of VT_ARRAY | VT_I4 , *refGndArray* is a VARIANT of VT_ARRAY | VT_I4

Microsoft C/C++ and Borland C++

Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Dim nChannelArray(0 To 1) As Integer
Dim nRangeArray(0 To 1) As Integer
Dim nRefGndArray(0 To 1) As Integer

nChannelArray(0) = 5
nRangeArray(0) = AD_B_10_V
nRefGndArray(0) = AI_RSE
nChannelArray(1) = 8
nRangeArray(1) = AD_B_2_5_V
nRefGndArray(1) = AI_RSE

vChannel = nChannelArray
vRange = nRangeArray
vRefGnd = nRefGndArray

Daq2205.AI.MuxScanSetup vChannel, vRange, vRefGnd
```

AI.StartContAI Method

This method performs continuous A/D conversions on the specified analog input channels at a rate as close to the rate you specified. This method takes advantage of the DAQ-2000 channel-gain that can be set separately for each channel to perform multi-channel/gain analog input. This method will fire AiComplete or AiHalfReady event depends on AI.DoubleBufferMode property.

Syntax

Function object.**AI.StartContAI**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Remarks

You can use this method to start the DMA analog input function. If the AI.StreamToFile property is True then the DMA data will be written to the file specified by AI.FileName. Otherwise, the AI.FileName property will be ignored.
The data file is written in binary format, with the lower byte first (little endian). Data type is “Binary codes with miscellaneous data”. DAQBench provides a convenient tool DAQCvt to convert the binary file to the file format read easily. See DAQBench User’s Guide for the usage of the utility. If you want to handle the data by yourself, please refer to Appendix D Data File Format for the file structure.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Dim nChannelArray(0 To 1) As Integer
Dim nRangeArray(0 To 1) As Integer
Dim nRefGndArray(0 To 1) As Integer

nChannelArray(0) = 5
nRangeArray(0) = AD_B_10_V
nRefGndArray(0) = AI_RSE
nChannelArray(1) = 8
nRangeArray(1) = AD_B_2_5_V
nRefGndArray(1) = AI_RSE

vChannel = nChannelArray
vRange = nRangeArray
vRefGnd = nRefGndArray

Daq2205.AI.MuxScanSetup vChannel, vRange, vRefGnd

Daq2205.AI.StartContAI

Private Sub Daq2205_AiComplete(ScaledData As Variant, BinaryCodes As Variant)
    ' Get Data in    ScaledData
End Sub

Private Sub Daq2205_AiHalfReady(ScaledData As Variant, BinaryCodes As Variant)
    ' Get Data in    ScaledData
End Sub
```

AI.StopContAI Method

You can use this method to force stop DMA analog input.

Syntax

Function object**.AI.StopContAI** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AI.ReadChannels Method

This method performs a software triggered A/D conversion (analog input) on analog input channels and returns the values converted.

Syntax

Function object**.AI.ReadChannels**(Buffer As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Buffer As Variant

An integer array to contain the acquired data. Please refer to Appendix C AD Data Format for the data format in the Buffer.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
    Dim nChannelArray(0 To 1) As Integer
    Dim nRangeArray(0 To 1) As Integer
    Dim nRefGndArray(0 To 1) As Integer

    nChannelArray(0) = 5
    nRangeArray(0) = AD_B_10_V
    nRefGndArray(0) = AI_RSE
    nChannelArray(1) = 8
    nRangeArray(1) = AD_B_2_5_V
    nRefGndArray(1) = AI_RSE

    vChannel = nChannelArray
    vRange = nRangeArray
    vRefGnd = nRefGndArray

   Daq2205.AI.MuxScanSetup vChannel, vRange, vRefGnd

Private Sub Timer1_Timer()
    Dim vBuffer As Variant
    Daq2205.AI.ReadChannels vBuffer
    ' Get Data in    vBuffer
End Sub
```

AO.StartContAO Method

This method performs continuous D/A conversions on the specified analog output channel at a rate as close to the rate you specified . This method will fire AoComplete or AoBufferReady event depends on AO.DoubleBufferMode property.

Syntax

Function object.**AO.StartContAO**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double
```

```
For i = 0 To 4095
    buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
Next i

For i = 0 To 4095
    If i < 2048 Then
        buffer2(i) = (&H800 + i Mod 2048) And &HFFF
    Else
        buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
    End If
Next i

Daq2205.AO.Channels(0).buffer1 = buffer1
Daq2205.AO.Channels(0).buffer2 = buffer2
Daq2205.AO.DoubleBufferMode = True
Daq2205.AO.Channels(0).Enable = True
Daq2205.AO.StartContAO
```

AO.StopContAO Method

You can use this method to force stop DMA analog output.

Syntax

Function object**.AO.StopContAO** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AO.WriteChannel Method

Accepts a voltage value (or a current value), scales it to the proper binary value and writes a binary value to the specified analog output channel.

Syntax

Function object**.AO.WriteChannel**(Channel as Integer, Voltage as Single) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Channel as Integer

The analog output channel number.

Range: 0 or 1 for DAQ -2205

Voltage as Single

The value to be scaled and written to the analog output channel. The range of voltages depends on the type of device, on the output polarity, and on the voltage reference (external or internal).

CALIBRATION.AutoCalibration Method

Uses this method to calibrate your DAQ -2000 device. When the method is called, the device goes into a self -calibration cycle. The method does not return until the self -calibration is completed.

Syntax
Function object.**CALIBRATION.AutoCalibration** () As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

CALIBRATION.DisplayErrors Method

Uses this method to fire AcquireADError and AcquireDAError eve nts. Through those events user can identification DAQ-2000 device current status.

Syntax
Function object.**CALIBRATION.DisplayErrors** () As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Example
Daq2205.CALIBRATION.DisplayErrors

```
Private Sub Daq2205_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "BiPolar"
    End If
    strMsg = "AD channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub

Private Sub Daq2205_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub
```

CALIBRATION.Load Method

Load calibration constants from the specified bank of EEPROM.

Syntax
Function object.**CALIBRATION.Load**(BankOfEEPROM As Integer) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments

BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

CALIBRATION.Save Method

Save calibration constants to the specified bank of EEPROM.

Syntax

Function object**.CALIBRATION.Save**(BankOfEEPROM As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

SSI.ClearAll Method

Disconnects all of the device signals from the SSI bus trigger lines.

Syntax

Function object**.ClearAll** As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DAQError Event

Syntax

sub ControlName_DAQError (ErrString As String)

Arguments

ErrString As String
The string of error reasion

Remarks

This event will occur when some error occur in control

AiComplete Event

Syntax

sub ControlName_AiComplete(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant
The analog input data array that have been translated to the engineering data (voltage) according to AI.Channels(n).Range property.

BinaryCodes As Variant
The analog input data array with digital format.

Remarks

This event occurs when continuous analog input function is completed. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting.*"BinaryCodes without channel only".*
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AiHalfReady Event

Syntax

sub ControlName_AiHalfReady(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant
The analog input data array that have been translated to the engineering data (voltage) according to AIRange property.

BinaryCodes As Variant
The analog input data array with digital format.

Remarks

This event occurs when one half-buffer of the circular buffer is full at continuous analog input operation. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting. "*BinaryCodes without channel only*"
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AoComplete Event

Syntax
sub ControlName _AoComplete( )

Arguments
None

Remarks
This event occurs when continuous analog output function is completed.

AoBufferReady Event

Syntax
Sub ControlName_AoBufferReady(BufferIndex as Integer)

Arguments
BufferIndex as Integer,
The index of the buffer just output.

Remarks
This event occurs when enable the AO.DoubleBufferMode, If User want to dynamic change the output pattern, process it when receive AoBufferReady event.

Example
```
'This sample code show that how to output four buffers (pattern) in one channel.
'You can modify this code for dynamic changes pattern.

Dim varArray(0 To 3) As Variant
Dim nCounter As Integer

Private Sub AO_Click()

    Dim buffer0(0 To 4095) As Integer
    Dim buffer1(0 To 4095) As Integer
    Dim buffer2(0 To 4095) As Integer
    Dim buffer3(0 To 4095) As Integer

    Dim i As Double

    For i = 0 To 4095
        buffer0(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer1(i) = (&H800 + i Mod 2048) And &HFFF
        Else
            buffer1(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        End If
    Next i
    For i = 0 To 4095
        buffer2(i) = (Cos(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer3(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        Else
            buffer3(i) = (&H800 + i Mod 2048) And &HFFF
        End If
    Next i

    varArray(0) = buffer0
```

```
    varArray(1) = buffer1
    varArray(2) = buffer2
    varArray(3) = buffer3

    Daq2205.AO.Channels(0).buffer1 = varArray(0)
    Daq2205.AO.Channels(0).buffer2 = varArray(1)
    nCounter = 1
    Daq2205.AO.Channels(0).Enable = True

    Daq2205.AO.DoubleBufferMode = True
    Daq2205.AO.StartContAO
End Sub

Private Sub Daq2205_AoBufferReady(ByVal BufferIndex As Integer)

nCounter = nCounter + 1
nCounter = nCounter Mod 4

If BufferIndex = 1 Then
    ' It can change buffer1 in here
    Daq2205.AO.Channels(0).buffer1 = varArray(nCounter)
End If

If BufferIndex = 2 Then
    ' It can change buffer2 in here
    Daq2205.AO.Channels(0).buffer2 = varArray(nCounter)
End If

End Sub
```

Acquire22XXADError Event

Acquires the offset and gain errors.

Syntax

Sub ***ControlName*** _Acquire22XXADError(ByVal gain_err As Double, ByVal bioffset_err As Double, ByVal unioffset_err As Double, ByVal hg_bios_err As Double)

Arguments

gain_err As Double,
The gain error of the ADC.

bioffset_err As Double,
The offset error of the ADC in bipolar mode.

unioffset_err As Double
The offset error of the ADC in unipolar mode.

hg_bios_err As Double
The high-gain offset error of the ADC in bipolar mode.

Example

Daq2205.CALIBRATION.DisplayErrors

```
Private Sub Daq2205_Acquire22XXADError(ByVal gain_err As Double, ByVal bioffset_err As Double,
ByVal unioffset_err As Double, ByVal hg_bios_err As Double)
    strMsg = "AD Gain error:" & Format(gain_err, "#0.#####")
    List1.AddItem (strMsg)
    strMsg = "Bio offset error:" & Format(bioffset_err, "#0.#####")
    List1.AddItem (strMsg)
    strMsg = "Uni offset error:" & Format(unioffset_err, "#0.#####")
    List1.AddItem (strMsg)
```

```
    strMsg = "Hign Gain bipolar Offset error:" & Format(hg_bios_err, "#0.#####")
    List1.AddItem (strMsg)
End Sub
```

AcquireDAError Event

Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

Syntax

sub *ControlName*_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_er r As Double)

Arguments

channel as Integer,
Indicate channel id.

polarity as Integer,
Indicate polarity

Value	Description
0	Unipolar
1	Bipolar

gain_err as Double
Indicate gain error

offset_err as Double
Indicate offset error

Example

Daq2205.CALIBRATION.DisplayErrors

```
Private Sub Daq2205_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
 List1.AddItem (strMsg)
 List1.Refresh
 End Sub
```

Daq2206 ActiveX Control

The Daq2206 ActiveX control is a software component that provides the interface for users to control PCI-2206 card.

Daq2206 ActiveX Control Overview			
Type (Group)	**Property**	**Method**	**Event**
General	DASKCardType	Open	DAQError
	CardNumber	AboutBox	
	OpenMode	ShowPropertyPages	
	DaskCardID		
DIO	P1ADir	ReadBackDOLine	
	P1BDir	ReadBackDOPort	
	P1CLowerDir	ReadDILine	
	P1CUpperDir	ReadDIPort	
		WriteDOLine	
		WriteDOPort	
GPTC	Mode	Start	
	ClockSource	Stop	
	ClockPolarity	Reset	
	GateSource	ReadStatus	
	GatePolarity		
	UpDownSource		
	UpDownPolarity		
	OutputPolarity		
	IntGATE		
	IntUpDnCTR		
	DelayCount		
	DurationCount		
	OutputValue		
AI	NumOfScan	StartContAI	AiComplete
	ClockSource	StopContAI	AiHalfReady
	ScanInterval	ReadChannels	
	SamplingInterval	MuxScanSetup	
	ConversionSource		
	TriggerMode		
	TriggerSource		
	DelaySource		
	ReTriggerModeEnable		
	MCounterEnable		
	PostTriggerCount		
	DelayCount		

Daq2206 ActiveX Control Overview			
	MCount		
	ReTriggerCount		
	ExtTrigPolarity		
	ReturnType		
	DoubleBufferMode		
	StreamToFile		
	FileName		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
AO	CHUI	StartContAO	AoComplete
	DAWRSource	StopContAO	AoBufferReady
	TriggerSource	WriteChannel	
	TriggerMode		
	Delay1Source		
	Delay2Source		
	ExtTrigPolarity		
	ReTriggerCount		
	Delay1Count		
	Delay2Count		
	ClockSource		
	ReTriggerModeEnable		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
	DoubleBufferMode		
	Iterations		
	Definite		
	StopMode		
	Channels(n).Enable		
	Channels(n).IntOrExtref		
	Channels(n).OutputPolarity		
	Channels(n).RefVoltage		
	Channels(n).Buffer1		
	Channels(n).Buffer2		
CALIBRATION	BankTemperature	AutoCalibration	Acquire22XXADError
	BankDate	Load	AcquireDAError

Daq2206 ActiveX Control Overview			
	CurrentTemperature	Save	
	CurrentDate	DisplayErrors	
SSI	TIMEBASE	ClearAll	
	ADCONV		
	DAWR		
	ADTRIG		
	DATRIG		

DASKCardType Property

Return a value that determines the card type. It is always DAQ_2206 in DAQ-2206 device.

.

Syntax

[Integer] =object**.CardType**

Remarks

Always return DAQ_2206 (for DAQ -2206 device)

Settings

Value	Constant	Description
3	DAQ_2206	For DAQ-2206 Device

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Data Type

Integer

CardNumber Property

The sequence number of the card with ***the same card type*** plugged in the PCI slot.
The card sequence number setting is according to the PCI slot sequence in the mainboard.
The first card (in the most prior slot) is with CardNumber=0. For example,
If there are two DAQ -2206 cards plugged on your PC, the DAQ -2206 card in the prior slot should be registered with CardNumber =0, and the other one with CardNumbe r =1.

.

Syntax

object**.CardNumber** [= short]

Remarks

This property will be used when Initializes the hardware states of a DAQ -2K data acquisition card.

Data Type

Integer

OpenMode Property

Return/Set a value that determines the mode of opening device.

Syntax

object**.OpenMode** [= short]

Settings

Value	Display	Description
0	Automatic	D2K-OCX will initialize by itself. Automatically open device when the control was created
1	Manual	Call object**.Open**() method to initialized D2K-OCX(open device)
2	Demonstration	D2K-OCX will provide software DAQ data to simulating hardware drivers.

Data Type
Integer

DaskCardID Property
Returns a value that determines the D2K_Register_Card() returns value,
the DaskCardID is a numer ic card ID that will be used by other D2K -DASK library functions.

Syntax
[short] =object.**DaskCardID**

Remarks
The range of card id is between 0 and 31.
This property will be used when combine D2K-DASK and D2K-OCX two module in one program.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example:
This sample will demonstration If user want to check AI completed by itself.

```
' OCX to starting continuous AI
  Dim nDaskID as Integer
  Daq2206.Open(TRUE)
  nDaskID = Daq2206.DaskCardID

Daq2206.AI.StartContAI

' Check AI Completed by DASK    API
Dim Stopped    As Byte
Dim AccessCnt As Long

Do While True
     D2K_AI_AsyncCheck nDaskID, Stopped, AccessCnt
     If Stopped = 1 Then
          Exit Do
     End If
Loop
MsgBox "AI Complete"
```

DIO.P1Adir Property
Return/Set a value that determines P1A port direction.

Syntax
object.**DIO.P1ADir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1Bdir Property
Return/Set a value that determines P1B port direction.

Syntax
object.**DIO.P1BDir** [= Direction]

Settings
Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CLowerdir Property
Return/Set a value that determines P1C lower port direction.

Syntax
object.**DIO.P1CLowerDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CUpperdir Property

Return/Set a value that determines P1C upper port direction.

Syntax

object.**DIO.P1CUpperDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.Mode Property

Return/Set a value that determines the Timer/Counter mode.

Syntax

object.**GPTC.Counter0.Mode** [= Integer]

Settings

Value	Constant	Description
1	SimpleGatedEventCNT	
2	SinglePeriodMSR	
3	SinglePulseWidthMSR	
4	SingleGatedPulseGen	
5	SingleTrigPulseGen	
6	RetrigSinglePulseGen	
7	SingleTrigContPulseGen	
8	ContGatedPulseGen	

Remarks

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockSource Property
GPTC.Counter1.ClockSource Property

Return/Set a value that determines the Timer/Counter Source .

Syntax

object.**GPTC.Counter0.ClockSource** [= Integer]
object.**GPTC.Counter1.ClockSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKSRC_INT	internal time base
2	GPTC_CLKSRC_EXT	external time base from GPTC0_SRC or GPTC1_SRC pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockPolarity Property
GPTC.Counter1.ClockPolarit y Property

Return/Set a value that determines the Timer/Counter clock polarity

Syntax

object.**GPTC.Counter0.ClockPolarity** [= Integer]
object.**GPTC.Counter1.ClockPolarity** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKEN_LACTIVE	Low active
2	GPTC_CLKEN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GateSou rce Property
GPTC.Counter1.GateSource Property

Return/Set a value that determines the Timer/Counter gate source

Syntax

object.**GPTC.Counter0.GateSource** [= Integer]
object.**GPTC.Counter1.GateSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_GATESRC_INT	gate is controlled by software
4	GPTC_GATESRC_EXT	gate is controlled by GPTC0_GATE or GPTC1_GATE pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GatePolarity Property
GPTC.Counter1.GatePolarity Property

Return/Set a value that determines the Timer/Counter gate polarity

Syntax

object.**GPTC.Counter0.GatePolarity** [= Integer]
object.**GPTC.Counter1.GatePolarity** [= Integer]

Settings

Value	Constant	Description
2	GPTC_GATE_LACTIVE	Low active
0	GPTC_GATE_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownSource Property
GPTC.Counter1.UpDownSource Property

Return/Set a value that determines the Timer/Counter UpDown Source

Syntax

object.**GPTC.Counter0.UpDownSource** [= Integer]
object.**GPTC.Counter1.UpDownSource** [= Integer]

Settings

Value	Constant	Description
0	GPTC_UPDOWN_SEL_INT	Up/Down controlled by software
16	GPTC_UPDOWN_SEL_EXT	Up/Down controlled by GPTC0_UPDOWN orGPTC1_UPDOWN pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownPolarity Property
GPTC.Counter1.UpDown Polarity Property

Return/Set a value that determines the Timer/Counter UpDown Polarity

Syntax

object.**GPTC.Counter0.UpDownPolarity** [= Integer]
object.**GPTC.Counter1.UpDownPolarity** [= Integer]

Settings

Value	Constant	Description
4	GPTC_UPDOWN_LACTIVE	Low active
0	GPTC_UPDOWN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.OutputPolarity Property
GPTC.Counter1.OutputPolarity Property

Return/Set a value that determines the Timer/Counter Output Polarity

Syntax

object.**GPTC.Counter0.OutputPolarity** [= Integer]
object.**GPTC.Counter1.OutputPolarity** [= Integer]

Settings

Value	Constant	Description
8	GPTC_OUTPUT_LACTIVE	Low active
0	GPTC_OUTPUT_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.IntGATE Property
GPTC.Counter1.IntGATE Property

Return/Set a value that determines the Timer/Counter Internal gate initialized status

Syntax

object.**GPTC.Counter0.IntGATE** [= Boolean]
object.**GPTC.Counter1.IntGATE** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type

Boolean

GPTC.Counter0.IntUpDnCTR Property
GPTC.Counter1.IntUpDnCTR Property

Return/Set a value that determines the Timer/Counter internal updown counter initi alized status.

Syntax

object.**GPTC.Counter0.IntUpDnCTR** [= Boolean]
object.**GPTC.Counter1.IntUpDnCTR** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type

Boolean

GPTC.Counter0.DelayCount Property

GPTC.Counter1.DelayCount P roperty

Return/Set a value that determines the Timer/Counter internal initial count of the GPTC or pulse delay. The counter value of load register 1 of timer/counter. The meaning for the value depends on the mode the timer /counter performs. For mode 1 to mode 3, the value is the initial count of the GPTC. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse delay.

Syntax

object.**GPTC.Counter0.DelayCount** [= Integer]
object.**GPTC.Counter1.DelayCount** [= Integer]

Data Type

Integer

GPTC.Counter0.DurationCount Property
GPTC.Counter1.DurationCount Property

Return/Set a value that determines the Timer/Counter pulse width when mode 4 to mode 8.
The counter value of load register 2 of timer/counter. For mode 1 to mode 3, the value is not used. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse width.

Syntax

object.**GPTC.Counter0.DurationCount** [= Integer]
object.**GPTC.Counter1.DurationCount** [= Integer]

Data Type

Integer

GPTC.Counter0.OutputValue Property
GPTC.Counter1.OutputValue Property

Returns the counter value of the specified general -purpose timer/counter.
Range: 0 through 65535

Syntax

[Integer=] object.**GPTC.Counter0.OutputValue**
[Integer=] object.**GPTC.Counter1.OutputValue**

Data Type

Integer

GPTC.CALIBRATION.BankTemperature Property

Returns a value that since user's last calibrated temperature in the EEPRON Bank .

Syntax

object.**CALIBRATION.BankTemperature**(*[BankOfEEPROM as Integer]*) As Single

Data Type

Single

GPTC.CALIBRATION.BankDate Property

Returns a value that since user's last calibrated date in the EEPRON Bank .

Syntax
object.**CALIBRATION.BankDate**(*[BankOfEEPROM as Integer]*) As String

Data Type
String

GPTC.CALIBRATION.CurrentTemperature Property
Returns a value that determines the current temperature on card.

Syntax
[Single=] object.**CALIBRATION.CurrentTemperature**

Data Type
Single

GPTC.CALIBRATION.CurrentDate Property
Returns a value that determines the current date.

Syntax
[String=] object.**CALIBRATION.CurrentDate**

Data Type
String

AI.NumOfScan Property
If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, NumOf *Scan* is the size (in samples) allocated for each channel in the circular buffer. This value must be a multiple of 2.

Syntax
object.**AI.NumOfScan** [=Long]

Remarks
Non-double-buffer mode
This value multiply the total number of scan channels is the total number of A/D conversions to be performed.
Double-buffer-mode
This value multiply the total number of scan channels is the size (in sample) of the circular buffer.

Data Type
Long

AI.ClockSource Property
The clock source (Time Base) the device selected.

Syntax
object.**AI.ClockSource** [=Short]

Settings
Value	Constant	Description

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ScanInterval Property
The length of the scan interval (that is, the counter value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.
If the timer base is from *external*, the valid range of the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of the value is 160 through 16777215

Syntax
object.**AI.ScanInterval** [=Long]

Data Type
Long

AI.SamplingInterval Property
The length of the sample interval (that is, the counter value between each A/D conversion within a scan sequence). The A/D conversion rate will be *TimeBase/SampIntrv*.The value of *TimeBase(AI.ClockSource)* depends on the card type .
If the timer base is from *external*, the valid range of the value is 2 through 65535.
If the timer base is *Internal timer*, the valid range of the value is as follows:
DAQ-2206 : 160 through 65535

Syntax
object.**AI.SamplingInterval** [=Long]

Data Type
Long

AI.ConversionSource Property
The A/D Conversion Source Selection.

Syntax
object.**AI.ConversionSource** [=Short]

Settings
Value	Constant	Description
0	DAQ2K_AI_ADCONVSRC_Int	Internal timer
4	DAQ2K_AI_ADCONVSRC_AFI0	From AFI0 pin

Value	Constant	Description
8	DAQ2K_AI_ADCONVSRC_SSI	From SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerMode Property
The Trigger Mode Selection.

Syntax
object**.AI.TriggerMode** [=Short]

Settings
Value	Constant	Description
0	DAQ2K_AI_TRGMOD_POST	Post Trigger Mode
8	DAQ2K_AI_TRGMOD_DELAY	Delay Trigger Mode
16	DAQ2K_AI_TRGMOD_PRE	Pre-Trigger Mode
24	DAQ2K_AI_TRGMOD_MIDL	Middle-Trigger Mode

Remarks
Please refer to the hardware manual for the trigger mode description.

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerSource Property
The trigger source selection.

Syntax
object**.AI.TriggerSource** [=Short]

Settings
Value	Constant	Description
0	DAQ2K_AI_TRGSRC_SOFT	Software
1	DAQ2K_AI_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_AI_TRGSRC_ExtD:	From external digital trigger pin
3	DAQ2K_AI_TRSRC_SSI	From SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.DelaySource Property
The delay source selection.

Syntax
object**.AI. DelaySource** [=Short]

Settings

Value	Constant	Description
256	DAQ2K_AI_Dly1InSamples	Delay in samples
0	DAQ2K_AI_Dly1InTimebase	Delay in time base

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReTriggerModeEnable Property
Re-trigger in an acquisition is enabled

Syntax
object**.AI.ReTriggerModeEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	ReTriggerMode is disabled.
1	True	ReTriggerMode is enabled.

Data Type
Boolean

AI.MCounterEnable Property
This constant is only valid for Pre -trigger and Middle trigger mode

Mcounter is enabled and then the trigger signal is ignore before M terminal count is reached.

Syntax
object.**AI.MCounterEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	MCounter is disabled.
1	True	Mcounter is enabled.

Data Type
Boolean

AI.PostTriggerCount Property
This constant is only valid for Middle trigger mode,
The PostTriggerCount indicates the number of data will be accessed after a specific trigger event.

Syntax
object.**AI.PostTriggerCount** [=Long]

Data Type
Long

AI.DelayCount Property
This constant is only valid for Delay trigger mode, The DelayCount indicates the number of data or timer ticks will be ignored after a specific trigger event.

Syntax
object.**AI.DelayCount** [=Long]

Data Type
Long

AI.MCount Property
The counter value of MCounter .This argument is only valid for Pretrigger and Middle trigger mode.

Syntax
object.**AI.MCount** [=Long]

Data Type
Long

AI.ReTriggerCount Property
The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax
object.**AI.ReTriggerCount** [=Long]

Data Type
Long

AI.ExtTrigPolarity Property

External Digital Trigger Polarity.

Syntax

object.**AI.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_AI_TrgPositive	Trigger positive edge active
4096	DAQ2K_AI_TrgNegative	Trigger negative edge active

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReturnType Property

Return/Set a value that determines the return data type of analog input when AiComplete or AiHalfReady event would occur.

Syntax

object.**AI.ReturnType** [=Integer]

Settings

Value	Constant	Description
0		Scaled data only
1		Binary codes without channel only
2		Scaled data and binary codes without channel
3		No data return

Data Type

Integer

AI.DoubleBufferMode Property

Enables or disables double -buffered data acquisition mo de.

Syntax

object.**AI.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type

Boolean

AI.StreamToFile Property

Return/Set a value that determines i f the control is enabled the function of streaming data to disk file. This argument is only valid for Delay trigger

Syntax

object.**AI.StreamToFile** [=Boolean]

Settings

Value	Constant	Description
0	False	Disable the function of streaming data to disk file.
1	True	Enable the function of streaming data to disk file

Data Type

Boolean

AI.FileName Property

FileName specified the file name of streaming data to disk. This argument is only valid for AI.StreamToFile is Enable.

Syntax

object.**AI.FileName** [=String]

Data Type

String

AI.AIOAnalogTrigCtrl Property

Return/Set a value that determines the t rigger source selection

Syntax

object.**AI.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	ADCATRIG	The first AI channel in the channel-gain queue
1	EXTATRIG	From external analog trigger pin

Data Type

Integer

AI.AIOTrigCondition Property

Return/Set a value that determines the t rigger condition selection

Syntax

object.**AI.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above -High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks

Please refer to the hardware manual for the more description.

Data Type

Integer

AI.AIOHLevel Property

The High value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AI.AIOHLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AI.AIOLLevel Property

The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AI.AIOLLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AO.CHUI Property

The length of the Channel Update interval (that is, the counter value between the initiation of each update sequence).
If the timer base is from *external*, the valid range of th e value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of the value is as follows:
Range: 40 through 16777215

Syntax
object**.AO.CHUI** [=Long]

Data Type
Long

AO.DAWRSource Property
Return/Set a value that determines the D/ A R/W Source Selection

Syntax
object**.AO.DAWRSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_WRSRC_Int	Internal timer
1	DAQ2K_DA_WRSRC_AFI0	From AFI0 pin
2	DAQ2K_DA_WRSRC_SSI	From SSI source

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerSource Property
Return/Set a value that determines the trigger source selection

Syntax
object**.AO.TriggerSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGSRC_SOFT	Software
1	DAQ2K_DA_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_DA_TRGSRC_ExtD	From external digital trigger pin
3	DAQ2K_DA_TRSRC_SSI	From SSI source

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerMode Property

Return/Set a value that determines the trigger mode selection

Syntax

object.**AO.TriggerMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGMOD_POST	Post Trigger Mode
1	DAQ2K_DA_TRGMOD_DELAY	Delay Trigger Mode

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay1Source Property

Return/Set a value that determines the delay1 source selection

Syntax

object.**AO.Delay1Source** [=Integer]

Settings

Value	Constant	Description
64	DAQ2K_DA_Dly1InUI	Delay in samples
0	DAQ2K_DA_Dly1InTimebase	Delay in time base

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay2Source Property

Return/Set a value that determines the delay2 source selection

Syntax
object.**AO.Delay2Source** [=Integer]

Settings

Value	Constant	Description
128	DAQ2K_DA_Dly2InUI	Delay in samples
0	DAQ2K_DA_Dly2InTimebase	Delay in time base

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ExtTrigPolarity Property
Return/Set a value that determines the external digital trigger polarity selection

Syntax
object.**AO.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TrgPositive	Trigger positive edge active
512	DAQ2K_DA_TrgNegative	Trigger negative edge active

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerCount Property
The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax
object.**AO.ReTriggerCount** [=Long]

Data Type
Long

AO.Delay1Count Property

The counter value of DLY1 Counter (the delay time after the trigger signal to the start of the waveform generation) . Thi s argument is only valid for Delay trigger mode.

Syntax

object.**AO.Delay1Count** [=Long]

Data Type

Long

AO.Delay2Count Property

The counter value of DLY2 Counter (the Delay between two consecutive waveform generations).

Syntax

object.**AO.Delay2Count** [=Long]

Data Type

Long

AO.ClockSource Property

The clock source (Time Base) the device selected.

Syntax

object.**AO.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerModeEnable Property

Re-trigger in an acquisition is enabled

Syntax

object.**AO.ClockSource** [=Boolean]

Settings

Value	Constant	Description
0	False	ReTriggerMode is disabled

Value	Constant	Description
1	True	ReTriggerMode is enabled

Data Type
Boolean

AO.AIOAnalogTrigCtrl Property
Return/Set a value that determines the t rigger source selection

Syntax
object.**AO.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	ADCATRIG	The first AI channel in the channel-gain queue
1	EXTATRIG	From external analog trigger pin

Data Type
Integer

AO.AIOTrigCondition Property
Return/Set a value that determines the t rigger condition selection

Syntax
object.**AO.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above-High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks
Please refer to the hardware manual for the more description.

Data Type
Integer

AO.AIOHLevel Property
The High value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax
object.**AO.AIOHLevel** [=Long]

Remarks
***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AO.AIOLLevel Property
The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax
object.**AO.AIOLLevel** [=Long]

Remarks
***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AO.DoubleBufferMode Property
Enables or disables double -buffered data acquisition mode.

Syntax
object.**AO.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type
Boolean

AO.Iterations Property
The times of number of the data in the buffer to output to the port. a value of zero is not allowed.

Syntax
object.**AO.Iterations** [=Long]

Data Type
Long

AO.Definite Property

Waveform generation proceeds definite or indefinitely. If double -buffered mode is enabled, this parameter is of no use.

Syntax

object.**AO.Definite** [=Boolean]

Settings

Value	Constant	Description
0	False	Indefinitely
1	True	Definite

Data Type
Boolean

AO.StopMode Property

Return/Set a val ue that determines DA transfer termination mode selected.

Syntax

object.**AO. StopMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TerminateImmediate	Software terminate the DA continuous operation immediately
1	DAQ2K_DA_TerminateUC	Software terminate the DA continuous operation on next update counter terminal count
2	DAQ2K_DA_TerminateIC	Software terminate the DA continuous operation on iteration count

Remarks

Please refer to the hardware manual for the more description.

Data Type

Integer

AO.Channels(0).Enable Property
AO.Channels(1).Enable Property

DAQ-2000 output channel that can be set separately for each channel to perform multi-channel analog input, The parameter 0~1 is channel id.

Syntax

object.**AO.Channels**(0).Enable [=Boolean]

Data Type
Boolean

AO.Channels(0).OutputPolarity Property
AO.Channels(1).OutputPolarity Property

Return/Set a value that determines polarity (unipolar or bipolar) of the output channel.

Syntax
object.**AO.Channels**(0).OutputPolarity [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_UniPolar	Unipolar
1	DAQ2K_DA_BiPolar	Bipolar

Data Type
Integer

AO.Channels(0).IntOrExtref Property
AO.Channels(1).IntOrExtref Property

Return/Set a value that determines DA reference voltage source of the output channel.

Syntax
object.**AO.Channels**(0). IntOrExtref [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_Int_REF	internal reference
1	DAQ2K_DA_Ext_REF	external reference

Data Type
Integer

AO.Channels(0).RefVoltage Property
AO.Channels(1).RefVoltage Property

If the D/A reference voltage source your device use is internal reference, the valid values is 10.
If the D/A reference voltage source your device use is external reference, the valid range is -10 to +10.

Syntax
object.**AO.Channels**(0).RefVoltage [=Single]

Data Type
Single

AO.Channels(0).Buffer1 Property
AO.Channels(1).Buffer1 Property

This property set up the buffer for continuous analog output operation. A buffer data or a array of buffer data, data type is integer.

Syntax

object.**AO.Channels**(0).Buffer1 [=Variant]

Remarks

You must assign this property before call StartContAO() method.
This property will be used when single or double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type

Variant(Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i
 Daq2206.AO.Channels(0).buffer1 = buffer1
 Daq2206.AO.Channels(0).Enable = True
 Daq2206.AO.StartContAO
```

AO.Channels(0).Buffer2 Property
AO.Channels(1).Buffer2 Property

This property set up the buffer for continuous analog output operation. A buffer data or a array of buffer data, data type is integer. This property only available when double buffer mode.

Syntax

object.**AO.Channels**(0).Buffer2 [=Variant]

Remarks

You must assign this property before call StartContAO() method.
This property will be used when double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type

Variant(Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double
```

```
For i = 0 To 4095
    buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
Next i

For i = 0 To 4095
    If i < 2048 Then
        buffer2(i) = (&H800 + i Mod 2048) And &HFFF
    Else
        buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
    End If
Next i

Daq2206.AO.Channels(0).buffer1 = buffer1
Daq2206.AO.Channels(0).buffer2 = buffer2
Daq2206.AO.DoubleBufferMode = True
Daq2206.AO.Channels(0).Enable = True
Daq2206.AO.StartContAO
```

SSI.ADCONV Property

Connect / Disconnect a SSI_ADCONV device signal to the specified SSI bus trigger line.

Syntax

object**.SSI.ADCONV** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.ADTRIG Property

Connect / Disconnect a SSI_ADTRIG device signal to the specified SSI bus trigger line.

Syntax

object**.SSI. ADTRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.DATRIG Property

Connect / Disconnect a SSI_DATRIG device signal to the specified SSI bus trigger line.

Syntax

object**.SSI.DATRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigge r line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.DAWR Property

Connect / Disconnect a SSI_DAWR device signal to the specified SSI bus trigger line.

Syntax

object**.SSI.DAWR** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.TIMEBASE Property

Connect / Disconnect a SSI_TIMEBASE device signal to the specified SSI bus trigger line.

Syntax

object**.SSI.TIMEBASE** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

DAQ_2206 Methods

Open Method

Syntax

Function object**.Open** ([ErrMsgBox As Variant]) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

[ErrMsgBox As Variant]

Boolean type.
True: It will popup error message dialog box when operation error.
False: It will fire DAQError event instead of popping up dialog when operation error.

Remarks

This method will be used when the OpenMode property is Manual.

Note

In VC++, *ErrMsgBox* is a VARIANT of VT_I2.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

ShowPropertyPages Method

This method will show propertypages of ActiveX Control.

Syntax

Function object.***ShowPropertyPages***() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AboutBox Method

This method will show About ADLINK dialog box.

Syntax

Function object.**AboutBox**( ) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DIO.ReadDIPort Method

Syntax

Function object.**DIO.ReadDIPort** (port As Integer, value as Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Returns the digital data read from the specified port. The returned value is 8 -bit data.

Remarks

You can read data from the digital input port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.ReadDILine Method

Syntax

Function object.**DIO.ReadDILine** (port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

The digital line to be read. The valid value is 0 through 7

value As Variant

Returns the digital logic state, 0 or 1, of the specified line.

Remarks

You can read data from the digital input port.

Note

In VC++, value is a VARIANT of VT_I2.

DIO.WriteDOPort Method

Syntax

Function object.**DIO.WriteDOPort** (port As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value as Variant

8-bit data that will be written to the digital output port.

Remarks

Users can write data to the digital output port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.WriteDOLine Method

Syntax

Function object.**DIO.WriteDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

Selects one line number from the indicated port: from 0 to 7 (8-bit port).

value As Variant

Sets 0 or 1 to the indicated line.

Note

In VC++, value is a VARIANT of VT_I2.

DIO.ReadBackDOPort Method

Reads back data from the indicated digital output port.

Syntax

Function object**.DIO.ReadBackDOPort** (port As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Data that is read back from the indicated port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.ReadBackDOLine Method

Reads back data from the indicated digital output line of the indicated digital output port.

Syntax

Function object**.DIO.ReadBackDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

Selects one line number from the indicated port: from 0 to 7 (8-bit port).

value As Variant

Data that is read back from the indicated line.

Note

In VC++, value is a VARIANT of VT_I2.

GPTC.Counter0.Start Method
GPTC.Counter1.Start Method

Start counter operation with the specified mode.

Syntax

Function object**.GPTC.Counter0.Start**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Remarks

You can start the indicated counter to operate in the specified mode.

GPTC.Counter0.Stop Method
GPTC.Counter1.Stop Method

Stop counter operation.

Syntax

Function object**.GPTC.Counter0.Stop**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

GPTC.Counter0.Reset Method
GPTC.Counter1.Reset Method

Halts the specified general -purpose timer/counter operation and reload the initial value of the timer/counter.

Syntax

Function object**.GPTC.Counter0.Reset**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

GPTC.Counter0.ReadStatus Method
GPTC.Counter1.ReadStatus Method

Reads the latched GPTC status of the general -purpose counter from the GPTC status register.

Syntax

Function object.**GPTC.Counter0.ReadStatus**(Clock *As Integer, Output As Integer, Gate As Integer, Updown As Integer,*) As Boolean

Return Value

True if the function is successful; otherwise False.

AI.MuxScanSetup Method

stores *channels*, *gain and reference ground* in the Channel-Gain Queue for a scanned data acquisition operation . The method uses this memory table during scanning operations `AI.StartContAI()` to automatically sequence through an arbitrary set of analog input channels and to allow gains to automatically change during scanning .

Syntax

Function object.**MuxScanSetup** (ChannelArray As Variant, AdRangeArray As Variant , refGndArray As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

ChannelArray As Variant

Array of analog input channel numbers.
numbers in channel must be within 0 and 63.

AdRangeArray As Variant

An integer array contains the analog input range.

Value	Constant	Description
1	AD_B_10_V	Bipolar -10V to +10V
2	AD_B_5_V	Bipolar -5V to +5V
3	AD_B_2_5_V	Bipolar -2.5V to +2.5V
4	AD_B_1_25_V	Bipolar -1.25V to +1.25V
15	AD_U_10_V	Unipolar 0 to +10V
16	AD_U_5_V	Unipolar 0 to +5V
17	AD_U_2_5_V	Unipolar 0 to +2.5V
18	AD_U_1_25_V	Unipolar 0 to +1.25V

refGndArray As Variant

An integer array contains the reference ground

0	**AI_RSE**	**Referenced single ended mode (64chs common to ground system on board)**
256	AI_DIFF	Differential mode
512	AI_NRSE	Non-referenced single ended mode (64chscommon to AISENSE pin)

Note

In VC++, Channel Array is a VARIANT of VT_ARRAY | VT_I4 , *AdRangeArray* is a VARIANT of VT_ARRAY | VT_I4 , *refGndArray* is a VARIANT of VT_ARRAY | VT_I4

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Dim nChannelArray(0 To 1) As Integer
Dim nRangeArray(0 To 1) As Integer
Dim nRefGndArray(0 To 1) As Integer

nChannelArray(0) = 5
nRangeArray(0) = AD_B_10_V
nRefGndArray(0) = AI_RSE
nChannelArray(1) = 8
nRangeArray(1) = AD_B_2_5_V
nRefGndArray(1) = AI_RSE

vChannel = nChannelArray
vRange = nRangeArray
vRefGnd = nRefGndArray

Daq2206.AI.MuxScanSetup vChannel, vRange, vRefGnd
```

AI.StartContAI Method

This method performs continuous A/D conversions on the specified a nalog input channels at a rate as close to the rate you specified. This method takes advantage of the DAQ-2000 channel-gain that can be set separately for each channel to perform multi-channel/gain analog input. This method will fire AiComplete or AiHalfRe ady event depends on AI.DoubleBufferMode property.

Syntax

Function object.**AI.StartContAI**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Remarks

You can use this method to start the DMA analog input function. If the AI.StreamToFile property is True then the DMA data will be written to the file specified by AI.FileName. Otherwise, the AI.FileName property will be ignored.

The data file is written in binary format, with the lower byte first (little endian). Data type is “Binary codes with miscellaneous data”. DAQBench provides a convenient tool DAQCvt to convert the binary file to the file format read easily. See DAQBench User’s Guide for the usage of the utility. If you want to handle the data by yourself, please refer to Appendix D Data File Format for the file structure.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Dim nChannelArray(0 To 1) As Integer
Dim nRangeArray(0 To 1) As Integer
Dim nRefGndArray(0 To 1) As Integer

nChannelArray(0) = 5
```

```
    nRangeArray(0) = AD_B_10_V
    nRefGndArray(0) = AI_RSE
    nChannelArray(1) = 8
    nRangeArray(1) = AD_B_2_5_V
    nRefGndArray(1) = AI_RSE

    vChannel = nChannelArray
    vRange = nRangeArray
    vRefGnd = nRefGndArray

   Daq2206.AI.MuxScanSetup vChannel, vRange, vRefGnd

   Daq2206.AI.StartContAI

Private Sub Daq2206_AiComplete(ScaledData As Variant, BinaryCodes As Variant)
    ' Get Data in    ScaledData
End Sub

Private Sub Daq2206_AiHalfReady(ScaledData As Variant, BinaryCodes As Variant)
     ' Get Data in    ScaledData
End Sub
```

AI.StopContAI Method

You can use this method to force stop DMA analog input.

Syntax

Function object**.AI.StopContAI** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AI.ReadChannels Method

This method performs a software triggered A/D conversion (analog input) on analog input channels and returns the values converted.

Syntax

Function object**.AI.ReadChannels**(Buffer As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Buffer As Variant

An integer array to contain the acquired data. Please refer to Appendix C AD Data Format for the data format in the Buffer.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
    Dim nChannelArray(0 To 1) As Integer
```

```
    Dim nRangeArray(0 To 1) As Integer
    Dim nRefGndArray(0 To 1) As Integer

    nChannelArray(0) = 5
    nRangeArray(0) = AD_B_10_V
    nRefGndArray(0) = AI_RSE
    nChannelArray(1) = 8
    nRangeArray(1) = AD_B_2_5_V
    nRefGndArray(1) = AI_RSE

    vChannel = nChannelArray
    vRange = nRangeArray
    vRefGnd = nRefGndArray

   Daq2206.AI.MuxScanSetup vChannel, vRange, vRefGnd

Private Sub Timer1_Timer()
    Dim vBuffer As Variant
    Daq2206.AI.ReadChannels vBuffer
    ' Get Data in    vBuffer
End Sub
```

AO.StartContAO Method

This method performs continuous D/A conversions on the specified analog output channel at a rate as close to the rate you specified.
This method will fire AoComplete or AoBufferReady event depends on AO.DoubleBufferMode property.

Syntax

Function object**.AO.StartContAO**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
    Dim buffer1(0 To 4095) As Integer
    Dim buffer2(0 To 4095) As Integer

    Dim i As Double

     For i = 0 To 4095
          buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
     Next i

     For i = 0 To 4095
          If i < 2048 Then
```

```
            buffer2(i) = (&H800 + i Mod 2048) And &HFFF
        Else
            buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        End If
    Next i

    Daq2206.AO.Channels(0).buffer1 = buffer1
    Daq2206.AO.Channels(0).buffer2 = buffer2
    Daq2206.AO.DoubleBufferMode = True
    Daq2206.AO.Channels(0).Enable = True
    Daq2206.AO.StartContAO
```

AO.StopContAO Method

You can use this method to force stop DMA analog output.

Syntax

Function object**.AO.StopContAO** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AO.WriteChannel Method

Accepts a voltage value (or a current value), scales it to the proper binary value and writes a binary value to the specified analog output channel.

Syntax

Function object**.AO.WriteChannel**(Channel as Integer, Voltage as Single) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Channel as Integer

The analog output channel number.
Range: 0 or 1 for DAQ -2206

Voltage as Single

The value to be scaled and written to the analog output channel. The range of voltages depends on the type of device, on the output polarity,
and on the voltage reference (external or internal).

CALIBRATION.AutoCalibration Method

Uses this method to calibrate your DAQ -2000 device. When the method is called, the device goes into a self-calibration cycle. The method does not return until the self -calibration is completed.

Syntax

Function object**.CALIBRATION.AutoCalibration** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

CALIBRATION.DisplayErrors Method

Uses this method to fire AcquireADError and AcquireDAError events. Through those events user can identification DAQ-2000 device current status.

Syntax

Function object**.CALIBRATION.DisplayErrors** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Example

Daq2206.CALIBRATION.DisplayErrors

```
Private Sub Daq2206_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "BiPolar"
    End If
    strMsg = "AD channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub

Private Sub Daq2206_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub
```

CALIBRATION.Load Method

Load calibration constants from the specified bank of EEPROM.

Syntax

Function object**.CALIBRATION.Load**(BankOfEEPROM As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

CALIBRATION.Save Method

Save calibration constants to the specified bank of EEPROM.

Syntax

Function object**.CALIBRATION.Save**(BankOfEEPROM As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

SSI.ClearAll Method

Disconnects all of the device signals from the SSI bus trigger lines.

Syntax

Function object**.ClearAll** As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DAQError Event

Syntax

sub ControlName_DAQError (ErrString As String)

Arguments

ErrString As String
The string of error reasion

Remarks

This event will occur when some error occur in control

AiComplete Event

Syntax

sub ControlName_AiComplete(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant
The analog input data array that have been translated to the engineering data (voltage) according to AI.Channels(n).Range property.

BinaryCodes As Variant
The analog input data array with digital format.

Remarks

This event occurs when continuous analog input function is completed. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting.*"BinaryCodes without channel only".*
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AiHalfReady Event

Syntax

sub ControlName_AiHalfReady(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant
The analog input data array that have been translated to the engineering data (voltage) according to AIRange property.

BinaryCodes As Variant
The analog input data array with digital format.

Remarks

This event occurs when one half-buffer of the circular buffer is full at continuous analog input operation. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting. "*BinaryCodes without channel only*"
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AoComplete Event

Syntax
sub ControlName _AoComplete( )

Arguments
None

Remarks
This event occurs when continuous a nalog output function is completed.

AoBufferReady Event

Syntax
Sub ControlName_AoBufferReady(BufferIndex as Integer)

Arguments
BufferIndex as Integer, The index of the buffer just output.

Remarks
This event occurs when enable the AO.DoubleBufferMode, If User want to dynamic change the output pattern, process it when receive AoBufferReady event.

Example
```
'This sample code show that how to output four buffers (pattern) in one channel.
'You can modify this code for dynamic changes pattern.

Dim varArray(0 To 3) As Variant
Dim nCounter As Integer

Private Sub AO_Click()

    Dim buffer0(0 To 4095) As Integer
    Dim buffer1(0 To 4095) As Integer
    Dim buffer2(0 To 4095) As Integer
    Dim buffer3(0 To 4095) As Integer

    Dim i As Double

    For i = 0 To 4095
        buffer0(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer1(i) = (&H800 + i Mod 2048) And &HFFF
        Else
            buffer1(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        End If
    Next i
    For i = 0 To 4095
        buffer2(i) = (Cos(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer3(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        Else
            buffer3(i) = (&H800 + i Mod 2048) And &HFFF
        End If
    Next i

    varArray(0) = buffer0
    varArray(1) = buffer1
```

```
    varArray(2) = buffer2
    varArray(3) = buffer3

    Daq2206.AO.Channels(0).buffer1 = varArray(0)
    Daq2206.AO.Channels(0).buffer2 = varArray(1)
    nCounter = 1
    Daq2206.AO.Channels(0).Enable = True

    Daq2206.AO.DoubleBufferMode = True
    Daq2206.AO.StartContAO
End Sub

Private Sub Daq2206_AoBufferReady(ByVal BufferIndex As Integer)

nCounter = nCounter + 1
nCounter = nCounter Mod 4

If BufferIndex = 1 Then
    ' It can change buffer1 in here
    Daq2206.AO.Channels(0).buffer1 = varArray(nCounter)
End If

If BufferIndex = 2 Then
    ' It can change buffer2 in here
    Daq2206.AO.Channels(0).buffer2 = varArray(nCounter)
End If

End Sub
```

Acquire22XXADError Event

Acquires the offset and gain errors.

Syntax

Sub ***ControlName*** _Acquire22XXADError(ByVal gain_err As Double, ByVal bioffset_err As Double, ByVal unioffset_err As Double, ByVal hg_bios_err As Double)

Arguments

gain_err As Double ,
The gain error of the ADC.

bioffset_err As Double,
The offset error of the ADC in bipolar mode.

unioffset_err As Double
The offset error of the ADC in unipolar mode.

hg_bios_err As Double
The high-gain offset error of the ADC in bipolar mode.

Example

Daq2206.CALIBRATION.DisplayErrors

```
Private Sub Daq2206_Acquire22XXADError(ByVal gain_err As Double, ByVal bioffset_err As Double,
ByVal unioffset_err As Double, ByVal hg_bios_err As Double)
    strMsg = "AD Gain error:" & Format(gain_err, "#0.#####")
    List1.AddItem (strMsg)
    strMsg = "Bio offset error:" & Format(bioffset_err, "#0.#####")
    List1.AddItem (strMsg)
    strMsg = "Uni offset error:" & Format(unioffset_err, "#0.#####")
    List1.AddItem (strMsg)
    strMsg = "Hign Gain bipolar Offset error:" & Format(hg_bios_err, "#0.#####")
```

```
    List1.AddItem (strMsg)
End Sub
```

AcquireDAError Event

Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

Syntax

sub *ControlName*_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)

Arguments

***channel as Integer*,**

Indicate channel id.

polarity *as Integer,*

Indicate polarity

Value	Description
0	Unipolar
1	Bipolar

gain_err as Double

Indicate gain error

offset_err as Double

Indicate offset error

Example

Daq2206.CALIBRATION.DisplayErrors

```
Private Sub Daq2206_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####") & " Offset error:" & Format(offset_err, "#0.#####")
  List1.AddItem (strMsg)
  List1.Refresh
  End Sub
```

Daq2501 ActiveX Control

The Daq2501 ActiveX control is a software component that provides the interface for users to control PCI-2501 card.

Daq2501 ActiveX Control Overview			
Type(Group)	**Property**	**Method**	**Event**
General	DASKCardType	Open	DAQError
	CardNumber	AboutBox	
	OpenMode	ShowPropertyPages	
	DaskCardID		
DIO	P1Adir	ReadBackDOLine	
	P1Bdir	ReadBackDOPort	
	P1ClowerDir	ReadDILine	
	P1CupperDir	ReadDIPort	
		WriteDOLine	
		WriteDOPort	
GPTC	Mode	Start	
	ClockSource	Stop	
	ClockPolarity	Reset	
	GateSource	ReadStatus	
	GatePolarity		
	UpDownSource		
	UpDownPolarity		
	OutputPolarity		
	IntGATE		
	IntUpDnCTR		
	DelayCount		
	DurationCount		
	OutputValue		
AI	NumOfScan	StartContAI	AiComplete
	Channels(n).Enable	StopContAI	AiHalfReady
	ClockSource	ReadChannels	
	ScanInterval		
	SamplingInterval		
	ConversionSource		
	TriggerMode		
	TriggerSource		
	DelaySource		
	ReTriggerModeEnable		
	McounterEnable		
	PostTriggerCount		

Daq2501 ActiveX Control Overview			
	DelayCount		
	Mcount		
	ReTriggerCount		
	ExtTrigPolarity		
	ReturnType		
	DoubleBufferMode		
	StreamToFile		
	FileName		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
	Range		
	DelayCounterSource		
AO	CHUI	StartContGroup	AoComplete
	DAWRSource	StopContGroup	AoBufferReady
	TriggerSource	WriteChannel	
	TriggerMode		
	Delay1Source		
	Delay2Source		
	ExtTrigPolarity		
	ReTriggerCount		
	Delay1Count		
	Delay2Count		
	ClockSource		
	ReTriggerModeEnable		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
	DelayCounterSource		
	BreakDelayCounterSc		
	DoubleBufferMode		
	Iterations		
	Definite		
	StopMode		
	GroupA.Mode		
	GroupA.Channels(n).Enable		
	GroupA.Channels(n).IntOrExtref		
	GroupA.Channels(n).OutputPolarity		

Daq2501 ActiveX Control Overview			
	GroupA.Channels(n).RefVoltage		
	GroupA.Channels(n).Buffer1		
	GroupA.Channels(n).Buffer2		
CALIBRATION	BankTemperature	AutoCalibration	AcquireADError
	BankDate	Load	AcquireDAError
	CurrentTemperature	Save	
	CurrentDate	DisplayErrors	
SSI	TIMEBASE	ClearAll	
	ADCONV		
	DAWR		
	ADTRIG		
	DATRIG		

DAQ_2501 Properties

DASKCardType Property

Return a value that determines the card type. It is always DAQ_2501 in DAQ-2501 device.

.

Syntax

[Integer] =object.**CardType**

Remarks

Always return DAQ_2501 (for DAQ -2501 Device)

Settings

Value	Constant	Description
7	DAQ_2501	For DAQ-2501 Device

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Data Type

Integer

CardNumber Property

The sequence number of the card with ***the same card type*** plugged in the PCI slot.
The card sequence number setting is according t o the PCI slot sequence in the mainboard.
The first card (in the most prior slot) is with CardNumber=0. For example,
If there are two DAQ -2501 cards plugged on your PC, the DAQ -2501 card in the prior slot should be registered with CardNumber =0, and the o ther one with CardNumber =1.

.

Syntax

object.**CardNumber** [= short]

Remarks

This property will be used when Initializes the hardware states of a DAQ -2K data acquisition

card.

Data Type
Integer

OpenMode Property
Return/Set a value that determines the mode of opening device.

Syntax
object.**OpenMode** [= short]

Settings

Value	Display	Description
0	Automatic	D2K-OCX will initialize by itself. Automatically open device when the control was created
1	Manual	Call object.**Open**() method to initialized D2K-OCX(open device)
2	Demonstration	D2K-OCX will provide software DAQ data to simulating hardware drivers.

Data Type
Integer

DaskCardID Property
Returns a value that determines the D2K_Register_Card() returns value,
the DaskCardID is a numeric card ID that will be used by other D2K -DASK library functions.

Syntax
[short] =object.**DaskCardID**

Remarks
The range of card id is between 0 and 31.
This property will be used when combine D2K-DASK and D2K-OCX two module in one program.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example:
‘This sample will demonstration If user want to check AI completed by itself.

```
' OCX to starting continuous AI
Dim nDaskID as Integer
Daq2501.Open(TRUE)
nDaskID = Daq2501.DaskCardID
```

```
    Daq2501.AI.StartContAI

    ' Check AI Completed by DASK    API
  Dim Stopped    As Byte
  Dim AccessCnt As Long

  Do While True
      D2K_AI_AsyncCheck nDaskID, Stopped, AccessCnt
      If Stopped = 1 Then
          Exit Do
      End If
  Loop
  MsgBox "AI Complete"
```

DIO.P1Adir Property

Return/Set a value that determines P1A port direction.

Syntax

object.**DIO.P1ADir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1Bdir Property

Return/Set a value that determines P1B port direction.

Syntax

object.**DIO.P1BDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

nteger.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic

Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CLowerdir Property

Return/Set a value that determines P1C lower port direction.

Syntax

object.**DIO.P1CLowerDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CUpperdir Property

Return/Set a value that determines P1C upper port direction.

Syntax

object.**DIO.P1CUpperDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.Mode Property

Return/Set a value that determines the Timer/Counter mode.

Syntax

object.**GPTC.Counter0.Mode** [= Integer]

Settings

Value	Constant	Description

Value	Constant	Description
1	SimpleGatedEventCNT	
2	SinglePeriodMSR	
3	SinglePulseWidthMSR	
4	SingleGatedPulseGen	
5	SingleTrigPulseGen	
6	RetrigSinglePulseGen	
7	SingleTrigContPulseGen	
8	ContGatedPulseGen	

Remarks

Please refer to the hardware manual for the mode descrip tion.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockSource Property
GPTC.Counter1.ClockSource Property

Return/Set a value that determines the Timer/Counter Source.

Syntax

object.**GPTC.Counter0.ClockSource** [= Integer]
object.**GPTC.Counter1.ClockSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKSRC_INT	internal time base
2	GPTC_CLKSRC_EXT	external time base from GPTC0_SRC or GPTC1 _SRC pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockPolarity Property
GPTC.Counter1.ClockPolarity Property

Return/Set a value that determines the Timer/Counter clock polarity

Syntax

object.**GPTC.Counter0.ClockPolarity** [= Integer]
object.**GPTC.Counter1.ClockPolarity** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKEN_LACTIVE	Low active
2	GPTC_CLKEN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GateSource Property
GPTC.Counter1.GateSource Property

Return/Set a value that determines the Timer/Counter gate source

Syntax

object.**GPTC.Counter0.GateSource** [= Integer]
object.**GPTC.Counter1.GateSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_GATESRC_INT	gate is controlled by software
4	GPTC_GATESRC_EXT	gate is controlled by GPTC0_GATE or GPTC1_GATE pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GatePolarity Property
GPTC.Counter1.GatePolarity Property

Return/Set a value that determines the Timer/Counter gate polarity

Syntax

object.**GPTC.Counter0.GatePolarity** [= Integer]
object.**GPTC.Counter1.GatePolarity** [= Integer]

Settings

Value	Constant	Description
2	GPTC_GATE_LACTIVE	Low active
0	GPTC_GATE_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownSource Property
GPTC.Counter1.UpDownSource Property

Return/Set a value that determines the Timer/Counter UpDown Source

Syntax

object.**GPTC.Counter0.UpDownSource** [= Integer]
object.**GPTC.Counter1.UpDownSource** [= Integer]

Settings

Value	Constant	Description
0	GPTC_UPDOWN_SEL_INT	Up/Down controlled by software
16	GPTC_UPDOWN_SEL_EXT	Up/Down controlled by GPTC0_UPDOWN orGPTC1_UPDOWN pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownPolarity Property
GPTC.Counter1.UpDownPolarity Property

Return/Set a value that determines the Timer/Counter UpDown Polarity

Syntax

object.**GPTC.Counter0.UpDownPolarity** [= Integer]
object.**GPTC.Counter1.UpDownPolarity** [= Integer]

Settings

Value	Constant	Description

Value	Constant	Description
4	GPTC_UPDOWN_LACTIVE	Low active
0	GPTC_UPDOWN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.OutputPolarity Property
GPTC.Counter1.OutputPolarity Property

Return/Set a value that determines the Timer/Counter Output Polarity

Syntax
object.**GPTC.Counter0.OutputPolarity** [= Integer]
object.**GPTC.Counter1.OutputPolarity** [= Integer]

Settings

Value	Constant	Description
8	GPTC_OUTPUT_LACTIVE	Low active
0	GPTC_OUTPUT_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.IntGATE Property
GPTC.Counter1.IntGATE Property

Return/Set a value that determines the Timer/Counter Internal gate initialized status

Syntax
object.**GPTC.Counter0.IntGATE** [= Boolean]
object.**GPTC.Counter1.IntGATE** [= Boolean]

Settings

Value	Constant	Description

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type
Boolean

GPTC.Counter0.IntUpDnCTR Property
GPTC.Counter1.IntUpDnCTR Property

Return/Set a value that determines the Timer/Counter internal updown counter initialized status.

Syntax
object.**GPTC.Counter0.IntUpDnCTR** [= Boolean]
object.**GPTC.Counter1.IntUpDnCTR** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type
Boolean

GPTC.Counter0.DelayCount Property
GPTC.Counter1.DelayCount Property

Return/Set a value that determines the Timer/Counter internal initial count of the GPTC or pulse delay. The counter value of load register 1 of timer/counter. The meaning for the value depends on the mode the timer /counter performs. For mode 1 to mode 3, the value is the initial count of the GPTC. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse delay.

Syntax
object.**GPTC.Counter0.DelayCount** [= Integer]
object.**GPTC.Counter1.DelayCount** [= Integer]

Data Type
Integer

GPTC.Counter0.DurationCount Property
GPTC.Counter1.DurationCount Property

Return/Set a value that determines the Timer/Counter pulse width when mode 4 to mode 8.
The counter value of load register 2 of timer/counter. For mode 1 to mode 3, the value is not used. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse width.

Syntax
object.**GPTC.Counter0.DurationCount** [= Integer]
object.**GPTC.Counter1.DurationCount** [= Integer]

Data Type
Integer

GPTC.Counter0.OutputValue Property
GPTC.Counter1.OutputValue Property

Returns the counter value of the specified general -purpose timer/counter.
Range: 0 through 65535

Syntax
[Integer=] object**.GPTC.Counter0.OutputValue**
[Integer=] object**.GPTC.Counter1.OutputValue**

Data Type
Integer

GPTC.CALIBRATION.BankTemperature Property

Returns a value that since user's last calibrated temperature in the EEPRON Bank .

Syntax
object**.CALIBRATION.BankTemperature**(*[BankOfEEPROM as Integer]*) As Single

Data Type
Single

GPTC.CALIBRATION.BankDate Property

Returns a value that since user's last calibrated date in the EEPRON Bank .

Syntax
object**.CALIBRATION.BankDate**(*[BankOfEEPROM as Integer]*) As String

Data Type
String

GPTC.CALIBRATION.CurrentTemperat ure Property

Returns a value that determines the current temperature on card.

Syntax
[Single=] object**.CALIBRATION.CurrentTemperature**

Data Type
Single

GPTC.CALIBRATION.CurrentDate Property

Returns a value that determines the current date.

Syntax
[String=] object**.CALIBRATION.CurrentDate**

Data Type
String

AI.NumOfScan Property

If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, NumOf *Scan* is the size (in samples) allocated for each channel in the circular buffer. This value must be a multiple of 2.

Syntax
object**.AI.NumOfScan** [=Long]

Remarks
Non-double-buffer mode
This value multiply the total number of scan channels is the total number of A/D conversions to be performed.
Double-buffer-mode
This value multiply the total number of scan channels is the size (in sample) of the circular buffer.

Data Type
Long

AI.ClockSource Property

The clock source (Time Base) the device selected.

Syntax
object**.AI.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ScanInterval Property

The length of the scan interval (that is, the counter value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.
If the timer base is from *external*, the valid range of the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of the value is 100 through 16777215

Syntax
object.**AI.ScanInterval** [=Long]

Data Type
Long

AI.SamplingInterval Property
The length of the sample interval (that is, the counter value between each A/D conversion within a scan sequence). The A/D conversion rate will be *TimeBase/SampIntrv*. The value of *TimeBase(AI.ClockSource)* depends on the card type.
If the timer base is from *external*, the valid range of the value is 2 through 65535.
If the timer base is *Internal timer*, the valid range of the value is as follows:
DAQ-2501 : 100 through 16777215

Syntax
object.**AI.SamplingInterval** [=Long]

Data Type
Long

AI.ConversionSource Property
The A/D Conversion Source Selection.

Syntax
object.**AI.ConversionSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_ADCONVSRC_Int	Internal timer
4	DAQ2K_AI_ADCONVSRC_AFI0	From AFI0 pin
8	DAQ2K_AI_ADCONVSRC_SSI	From SSI source
12	DAQ2K_AI_ADCONVSRC_AFI1	From AFI1 pin

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerMode Property
The Trigger Mode Selection.

Syntax
object.**AI.TriggerMode** [=Short]

Settings

Value	Constant	Description

Value	Constant	Description
0	DAQ2K_AI_TRGMOD_POST	Post Trigger Mode
8	DAQ2K_AI_TRGMOD_DELAY	Delay Trigger Mode
16	DAQ2K_AI_TRGMOD_PRE	Pre-Trigger Mode
24	DAQ2K_AI_TRGMOD_MIDL	Middle-Trigger Mode

Remarks
Please refer to the hardware manual for the trigger mode description.

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerSource Property
The Trigger Source Selection.

Syntax
object.**AI.TriggerSource** [=Short]

Settings
Value	Constant	Description
0	DAQ2K_AI_TRGSRC_SOFT	Software
1	DAQ2K_AI_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_AI_TRGSRC_ExtD:	From external digital trigger pin
3	DAQ2K_AI_TRSRC_SSI	From SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.DelaySource Property
The delay source selection.

Syntax
object.**AI. DelaySource** [=Short]

Settings
Value	Constant	Description

Value	Constant	Description
256	DAQ2K_AI_Dly1InSamples	Delay in samples

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReTriggerModeEnable Property
Re-trigger in an acquisition is enabled

Syntax
object.**AI.ReTriggerModeEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	ReTriggerMode is disabled.
1	True	ReTriggerMode is enabled.

Data Type
Boolean

AI.MCounterEnable Property
This constant is only valid for Pre -trigger and Middle trigger mode .
Mcounter is enabled and then the trigger signal is ignore before M terminal count is reached.

Syntax
object.**AI.MCounterEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	MCounter is disabled.
1	True	Mcounter is enabled.

Data Type
Boolean

AI.PostTriggerCount Property
This constant is only valid for Middle trigger mode,
The PostTriggerCount indicates the number of data will be accessed after a specific trigger event.

Syntax
object.**AI.PostTriggerCount** [=Long]

Data Type
Long

AI.DelayCount Property

This constant is only valid for Delay trigger mode,
The DelayCount indicates the number of data or timer ticks will be ignored after a specifi c trigger event.

Syntax
object.**AI.DelayCount** [=Long]

Data Type
Long

AI.MCount Property

The counter value of MCounter . This argument is only valid for Pretrigger and Middle trigger mode.

Syntax
object.**AI.MCount** [=Long]

Data Type
Long

AI.ReTriggerCount Property

The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax
object.**AI.ReTriggerCount** [=Long]

Data Type
Long

AI.ExtTrigPolarity Property

External Digital Trigger Polari ty.

Syntax
object.**AI.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_AI_TrgPositive	Trigger positive edge active
4096	DAQ2K_AI_TrgNegative	Trigger negative edge active

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReturnType Property

Return/Set a value that determines the return data type of analog input when AiComplete or AiHalfReady event would occur.

Syntax

object**.AI.ReturnType** [=Integer]

Settings

Value	Constant	Description
0		Scaled data only
1		Binary codes without channel only
2		Scaled data and binary codes without channel
3		No data return

Data Type

Integer

AI.DoubleBufferMode Property

Enables or disables double -buffered data acquisition mode.

Syntax

object**.AI.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type

Boolean

AI.StreamToFile Property

Return/Set a value that determines if the control is enabled the function of streaming data to disk file. This argument is only valid for Delay trigger

Syntax

object**.AI.StreamToFile** [=Boolean]

Settings

Value	Constant	Description
0	False	Disable the function of streaming data to disk file.
1	True	Enable the function of streaming data to disk file

Data Type

Boolean

AI.FileName Property

FileName specified the file name of streaming data to disk. This argument is only valid for AI.StreamToFile is Enable.

Syntax
object.**AI.FileName** [=String]

Data Type
String

AI.AIOAnalogTrigCtrl Property
Return/Set a value that determines the t rigger source selection

Syntax
object.**AI.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	ADCATRIG	The first AI channel in the channel-gain queue
1	EXTATRIG	From external analog trigger pin

Data Type
Integer

AI.AIOTrigCondition Property
Return/Set a value that determines the t rigger condition selection

Syntax
object.**AI.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above-High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks
Please refer to the hardware manual for the more description.

Data Type
Integer

AI.AIOHLevel Property
The High value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax
object.**AI.AIOHLevel** [=Long]

Remarks
***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	Trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AI.AIOLLevel Property

The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax

object.**AI.AIOLLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AI.Channels(0).Enable Property
AI.Channels(1).Enable Property
AI.Channels(2).Enable Property
AI.Channels(3).Enable Property
AI.Channels(4).Enable Property
AI.Channels(5). Enable Property
AI.Channels(6).Enable Property
AI.Channels(7).Enable Property

DAQ-2000 channel-gain that can be set separately for each channel to perform multi-channel/gain analog input, The parameter 0~7 is channel id.

Syntax

object.**AI.Channels**(0).Enable [=Boolean]

Data Type
Boolean

AI.Range Property

This property can setting same range for all channels.

Syntax
object**.AI.Range** [=Integer]

Settings

Value	Constant	Description
1	AD_B_10_V	Bipolar -10V to +10V
15	AD_U_10_V	Unipolar 0 to +10V

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.DelayCounterSource Property

A/D Delay Counter Source Selection

Syntax
object**.AI.DelayCounterSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_AI_DTSRC_Int	Internal timer
16	DAQ2K_AI_DTSRC_AFI1	From AFI1 pin
32	DAQ2K_AI_DTSRC_GPTC0	From GPTC0_OUT
48	DAQ2K_AI_DTSRC_GPTC1	From GPTC1_OUT

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.CHUI Property

The length of the Channel Update interval (that is, the counter value between the initiation of each update sequence).
If the timer base is from *external*, the valid range of the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of the value is as follows:

Range: 40 through 16777215

Syntax
object.**AO.CHUI** [=Long]

Data Type
Long

AO.DAWRSource Property
Return/Set a value that deter mines the D/A R/W Source Selection

Syntax
object.**AO.DAWRSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_WRSRC_Int	Internal timer
1	DAQ2K_DA_WRSRC_AFI0	From AFI0 pin
2	DAQ2K_DA_WRSRC_SSI	From SSI source

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerSource Property
Return/Set a value that determines the trigger source selection

Syntax
object.**AO.TriggerSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGSRC_SOFT	Software
1	DAQ2K_DA_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_DA_TRGSRC_ExtD	From external digital trigger pin
3	DAQ2K_DA_TRSRC_SSI	From SSI source

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerMode Property

Return/Set a value that determines the trigger mode selection

Syntax

object.**AO.TriggerMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGMOD_POST	Post Trigger Mode
1	DAQ2K_DA_TRGMOD_DELAY	Delay Trigger Mode

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay1Source Property

Return/Set a value that determines the delay1 source selection

Syntax

object.**AO.Delay1Source** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_Dly1InTimebase	Delay in time base

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay2Source Property

Return/Set a value that determines the delay2 source selection

Syntax

object.**AO.Delay2Source** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_Dly2InTimebase	Delay in time base

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ExtTrigPolarity Property

Return/Set a value that determines the e xternal digital trigg er polarity selection

Syntax

object.**AO.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TrgPositive	Trigger positive edge active
512	DAQ2K_DA_TrgNegative	Trigger negative edge active

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerCount Property

The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax

object.**AO.ReTriggerCount** [=Long]

Data Type

Long

AO.Delay1Count Property

The counter value of DLY1 Counter (the delay time after the trigger signal to the start of the waveform generation) . This argument is only valid for Delay trigger mode.

Syntax
object.**AO.Delay1Count** [=Long]

Data Type
Long

AO.Delay2Count Property

The counter value of DLY2 Counter (the Delay between two consecutive waveform generations).

Syntax
object.**AO.Delay2Count** [=Long]

Data Type
Long

AO.ClockSource Property

The clock source (Time Base) the device selected.

Syntax
object.**AO.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerModeEnable Property

Re-trigger in an acquisition is enabled

Syntax
object.**AO.ClockSource** [=Boolean]

Settings

Value	Constant	Description
0	False	ReTriggerMode is disabled
1	True	ReTriggerMode is enabled

Data Type
Boolean

AO.AIOAnalogTrigCtrl Property

Return/Set a value that determines the trigger source selection

Syntax
object.**AO.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	ADCATRIG	The first AI channel in the channel-gain queue
1	EXTATRIG	From external analog trigger pin

Data Type
Integer

AO.AIOTrigCondition Property

Return/Set a value that determines the trigger condition selection

Syntax
object.**AO.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above-High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks
Please refer to the hardware manual for the more description.

Data Type
Integer

AO.AIOHLevel Property

The High value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax
object.**AO.AIOHLevel** [=Long]

Remarks
***For example*:**
If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	Trigger voltage
0xFF	+9.92V

Trigger Level digital setting	Trigger voltage
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AO.AIOLLevel Property
The Low value setting of Trigger level. Pl ease refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax
object**.AO.AIOLLevel** [=Long]

Remarks
***For example*:**
If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	Trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AO.DelayCounterSource Property
D/A Trigger delay Counter Source Selection

Syntax
object**.AO.DelayCounterSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TDSRC_Int	Internal timer
16	DAQ2K_DA_TDSRC_AFI0	From AFI1 pin
32	DAQ2K_DA_TDSRC_GPTC0	From GPTC0_OUT
48	DAQ2K_DA_TDSRC_GPTC1	From GPTC1_OUT

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.BreakDelayCounterSc Property

D/A Break delay Counter Source Selection

Syntax

object.**AO.BreakDelayCounterSc** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_BDSRC_Int	Internal timer
16	DAQ2K_DA_BDSRC_AFI0	From AFI1 pin
32	DAQ2K_DA_BDSRC_GPTC0	From GPTC0_OUT
48	DAQ2K_DA_BDSRC_GPTC1	From GPTC1_OUT

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.DoubleBufferMode Property

Enables or disables double -buffered data acquisition mode.

Syntax

object.**AO.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type
Boolean

AO.Iterations Property

The times of number of the data in the buffer to output to the port. a value of zero is not allowed.

Syntax

object.**AO.Iterations** [=Long]

Data Type
Long

AO.Definite Property

Waveform generation proceeds definite or indefinitely. If double -buffered mode is enabled, this parameter is of no use.

Syntax

object.**AO.Definite** [=Boolean]

Settings

Value	Constant	Description
0	False	Indefinitely
1	True	Definite

Data Type

Boolean

AO.StopMode Property

Return/Set a value that determines DA transfer termination mode selected.

Syntax

object.**AO. StopMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TerminateImmediate	Software terminate the DA continuous operation immediately
1	DAQ2K_DA_TerminateUC	Software terminate the DA continuous operation on next update counter terminal count
2	DAQ2K_DA_TerminateIC	Software terminate the DA continuous operation on iteration count

Remarks

Please refer to the hardware manual for the more description.

Data Type

Integer

AO.GroupA.Mode Property

Return/Set a value that determines GroupA DA transfer mode selected.

Syntax

object.**AO.GroupA.Mode** [=Integer]

Settings

Value	Constant	Description
0	GROUP_MODE_DMA	Using DMA to transfer data
1	GROUP_MODE_FIFO	The data will be loaded into on-board DA FIFO(8K samples)

Remarks

Please refer to the hardware manual for the more description.

Data Type

Integer

AO.GroupA.Channels(0).Enable Pro perty
AO.GroupA.Channels(1).Enable Property

AO.GroupA.Channels(2).Enable Property
AO.GroupA.Channels(3).Enable Property

DAQ-2000 output channel that can be set separately for each channel to perform multi-channel analog input, The parameter 0~3 is channel id.

Syntax

object.**AO.GroupA.Channels**(0).Enable [=Boolean]

Data Type

Boolean

AO.GroupA.Channels(0).OutputPolarity Property
AO.GroupA.Channels(1).OutputPolarity Property
AO.GroupA.Channels(2).OutputPolarity Property
AO.GroupA.Channels(3).OutputPolari ty Property

Return/Set a value that determines polarity (unipolar or bipolar) of the output channel.

Syntax

object.**AO.GroupA.Channels**(0).OutputPolarity [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_UniPolar	Unipolar
1	DAQ2K_DA_BiPolar	Bipolar

Data Type

Integer

AO.GroupA.Channels(0).IntOrExtref Property
AO.GroupA.Channels(1).IntOrExtref Property
AO.GroupA.Channels(2).IntOrExtref Property
AO.GroupA.Channels(3).IntOrExtref Property

Return/Set a value that determines DA reference voltage source of the output channel.

Syntax

object.**AO.GroupA.Channels**(0). IntOrExtref [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_Int_REF	internal reference
1	DAQ2K_DA_Ext_REF	external reference

Data Type
Integer

AO.GroupA.Channels(0).RefVoltage Property
AO.GroupA.Channels(1).RefVoltage Property
AO.GroupA.Channels(2).RefVoltage Property
AO.GroupA.Channels(3).RefVoltage Property

If the D/A reference voltage source your device use is internal reference, the valid values is 10.
If the D/A reference voltage source your device use is external reference, the valid range is -10 to +10.

Syntax
object.**AO.GroupA.Channels**(0).RefVoltage [=Single]

Data Type
Single

AO.GroupA.Channels(0).Buffer1 Property
AO.GroupA.Channels(1).Buffer1 Property
AO.GroupA.Channels(2).Buffer1 Property
AO.GroupA.Channels(3).Buffer1 Property

This property set up the buffer for continuous analog output operation.
A buffer data or a array of buffer data, data type is integer.

Syntax
object.**AO.GroupA.Channels**(0).Buffer1 [=Variant]

Remarks
You must assign this property before call AO.StartContGroup(DA_Group_A) method.
This property will be used when single or double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type
Variant (Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i
 Daq2501.AO.GroupA.Channels(0).buffer1 = buffer1
 Daq2501.AO.GroupA.Channels(0).Enable = True
```

Daq2501.AO.StartContGroup(DA_Group_A_)

AO.GroupA.Channels(0).Buffer2 Property
AO.GroupA.Channels(1).Buffer2 Property
AO.GroupA.Channels(2).Buffer2 Property
AO.GroupA.Channels(3).Buffer2 Property

This property set up the buffer for continuous analog output operation. A buffer data or a array of buffer data, data type is integer. This property only available when double buffer mode.

Syntax

object.**AO.GroupA.Channels**(0).Buffer2 [=Variant]

Remarks

You must assign this property before call AO.StartContGroup(DA_Group_A) method.
This property will be used when double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type

Variant(Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i

 For i = 0 To 4095
      If i < 2048 Then
           buffer2(i) = (&H800 + i Mod 2048) And &HFFF
      Else
           buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
      End If
 Next i

 Daq2501.AO.GroupA.Channels(0).buffer1 = buffer1
 Daq2501.AO.GroupA.Channels(0).buffer2 = buffer2
 Daq2501.AO.GroupA.Channels(0).Enable = True
 Daq2501.AO.DoubleBufferMode = True
 Daq2501.AO.StartContGroup(DA_Group_A)
```

SSI.ADCONV Property

Connect / Disconnect a SSI_ADCONV device signal to the specified SSI bus trigger line.

Syntax

object.**SSI.ADCONV** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.ADTRIG Property

Connect / Disconnect a SSI_ADTRIG device signal to the specified SSI bus trigger line.

Syntax

object**.SSI. ADTRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.DATRIG Property

Connect / Disconnect a SSI_DATRIG device signal to the specified SSI bus trigger line.

Syntax

object**.SSI.DATRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.DAWR Property

Connect / Disconnect a SSI_DAWR device signal to the specified SSI bus trigger line.

Syntax

object**.SSI.DAWR** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type

Boolean

SSI.TIMEBASE Property

Connect / Disconnect a SSI_TIMEBASE device signal to the specified SSI bus trigger line.

Syntax
object.**SSI.TIMEBASE** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

DAQ_2501 Methods

Open Method

Syntax
Function object.**Open** ([ErrMsgBox As Variant]) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
[ErrMsgBox As Variant]

Boolean type.
True: It will popup error message dialog box when operation error.
False: It will fire DAQError event instead of popping up dialog when operation error.

Remarks
This method will be used when the OpenMode property is Manual.

Note
In VC++, *ErrMsgBox* is a VARIANT of VT_I2.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

ShowPropertyPages Method

This method will show propertypages of ActiveX Control.

Syntax
Function object.***ShowPropertyPages***() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

AboutBox Method

This method will show About ADLINK dialog box.

Syntax
Function object.**AboutBox**( ) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DIO.ReadDIPort Method

Syntax

Function object.**DIO.ReadDIPort** (port As Integer, value as Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Returns the digital data read from the specified port. The returned value is 8 -bit data.

Remarks

You can read data from the digital input port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.ReadDILine Method

Syntax

Function object.**DIO.ReadDILine** (port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

The digital line to be read. The valid value is 0 through 7

value As Variant

Returns the digital logic state, 0 or 1, of the specified line.

Remarks

You can read data from the digital input port.

Note

In VC++, value is a VARIANT of VT_I2.

DIO.WriteDOPort Method

Syntax

Function object.**DIO.WriteDOPort** (port As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value as Variant

8-bit data that will be written to the digital output port.

Remarks

Users can write data to the digital output port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.WriteDOLine Method

Syntax

Function object.**DIO.WriteDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

Selects one line number from the indicated port: from 0 to 7 (8-bit port).

value As Variant

Sets 0 or 1 to the indicated line.

Note

In VC++, value is a VARIANT of VT_I2.

DIO.ReadBackDOPort Method

Reads back data from the indicated digital output port.

Syntax

Function object.**DIO.ReadBackDOPort** (port As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant
Data that is read back from the indicated port.

Note
In VC++, value is a VARIANT of VT_I4.

DIO.ReadBackDOLine Method
Reads back data from the indicated digital output line of the indicated digital output port.

Syntax
Function object**.DIO.ReadBackDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
port As Integer
Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer
Selects one line number from the indicated port: from 0 to 7 (8-bit port).
value As Variant
Data that is read back from the indicated line.

Note
In VC++, value is a VARIANT of VT_I2.

GPTC.Counter0.Start Method
GPTC.Counter1.Start Method
Start counter operation with the specified mode.

Syntax
Function object**.GPTC.Counter0.Start**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Remarks
You can start the indicated counter to operate in the specified mode.

GPTC.Counter0.Stop Method
GPTC.Counter1.Stop Method
Stop counter operation.

Syntax
Function object.**GPTC.Counter0.Stop**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

GPTC.Counter0.Reset Method
GPTC.Counter1.Reset Method

Halts the specified general -purpose timer/counter operation and rel oad the initial value of the timer/counter.

Syntax
Function object.**GPTC.Counter0.Reset**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

GPTC.Counter0.ReadStatus Method
GPTC.Counter1.ReadStatus Method

Reads the latched GPTC status of the general -purpose counter from the GPTC status register.

Syntax
Function object.**GPTC.Counter0.ReadStatus**(Clock *As Integer, Output As Integer, Gate As Integer, Updown As Integer,*) As Boolean

Return Value
True if the function is successful; otherwise False.

AI.StartContAI Method

This method performs continuous A/D conversions on the specified analog input channels at a rate as close to the rate you specified. This method takes advantage of the DAQ-2000 channel-gain that can be set separately for each channel to perform multi-channel/gain analog input. This method will fire AiComplete or AiHalfReady event depends on AI.DoubleBufferMode property.

Syntax
Function object.**AI.StartContAI**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Remarks
You can use this method to start the DMA analog input function. If the AI.StreamToFile property is True then the DMA data will be written to the file specified by AI.FileName. Otherwise, the AI.FileName property will be ignored. The data file is written in binary format, with the lower byte first (little endian). Data type is “Binary codes with miscellaneous data”. DAQBench provides a convenient tool DAQCvt to convert the binary file to the file format read easily. See DAQBench User’s Guide for the usage of the utility. If you want to handle the data by yourself, please refer to Appendix D Data File Format for the file structure.

Microsoft C/C++ and Borland C++

Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Daq2501.AI.Channels(0).Enable = True
Daq2501.AI.Channels(2).Enable = True
Daq2501.AI.Range = AD_B_10_V

Daq2501.AI.StartContAI

Private Sub Daq2501_AiComplete(ScaledData As Variant, BinaryCodes As Variant)
    ' Get Data in    ScaledData
End Sub

Private Sub Daq2501_AiHalfReady(ScaledData As Variant, BinaryCodes As Variant)
    ' Get Data in    ScaledData
End Sub
```

AI.StopContAI Method

You can use this method to force stop DMA analog input.

Syntax

Function object**.AI.StopContAI** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AI.ReadChannels Method

This method performs a software triggered A/D conversion (analog input) on analog input channels and returns the values converted.

Syntax

Function object**.AI.ReadChannels**(Buffer As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Buffer As Variant
An integer array to contain the acquired data. Please refer to Appendix C AD Data Format for the data format in the Buffer.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Daq2501.AI.Channels(0).Enable = True
Daq2501.AI.Channels(2).Enable = True
Daq2501.AI.Range = AD_B_10_V

Private Sub Timer1_Timer()
    Dim vBuffer As Variant
    Daq2501.AI.ReadChannels vBuffer
    ' Get Data in    vBuffer
End Sub
```

AO.StartContGroup Method

This method performs continuous D/A conversions on t he specified analog output channel at a rate as close to the rate you specified. This method will fire AoComplete or AoBufferReady event depends on AO.DoubleBufferMode property.

Syntax

Function object**.AO.StartContGroup(** GroupID As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

GroupID As Integer
Always set to DA_Group_A

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double

  For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
  Next i

  For i = 0 To 4095
      If i < 2048 Then
          buffer2(i) = (&H800 + i Mod 2048) And &HFFF
      Else
          buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
      End If
  Next i

  Daq2501.AO.GroupA.Channels(0).buffer1 = buffer1
  Daq2501.AO.GroupA.Channels(0).buffer2 = buffer2
  Daq2501.AO.GroupA.Channels(0).Enable = True
  Daq2501.AO.DoubleBufferMode = True
```

Daq2501.AO.StartContGroup(DA_Group_A_)

AO.StopContGroup Method

You can use this method to force stop analog output.

Syntax

Function object.**AO.StopContGroup**(GroupID As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

GroupID As Integer

Always set to DA_Group_A

AO.WriteChannel Method

Accepts a voltage value (or a current value), scales it to the proper binary valu e and writes a binary value to the specified analog output channel.

Syntax

Function object.**AO.WriteChannel**(Channel as Integer, Voltage as Single) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Channel as Integer

The analog output channel number.

Voltage as Single

The value to be scaled and written to the analog output channel. The range of voltages depends on the type of device, on the output polarity, and on the voltage reference (external or int ernal).

CALIBRATION.AutoCalibration Method

Uses this method to calibrate your DAQ -2000 device. When the method is called, the device goes into a self-calibration cycle. The method does not return until the self -calibration is completed.

Syntax

Function object.**CALIBRATION.AutoCalibration** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

CALIBRATION.DisplayErrors Method

Uses this method to fire AcquireADError and AcquireDAError events. Through those events use r can identification DAQ-2000 device current status.

Syntax

Function object.**CALIBRATION.DisplayErrors** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Example

Daq2501.CALIBRATION.DisplayErrors

```
Private Sub Daq2501_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "BiPolar"
    End If
    strMsg = "AD channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub

Private Sub Daq2501_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub
```

CALIBRATION.Load Method

Load calibration constants from the specified bank of EEPROM.

Syntax

Function object**.CALIBRATION.Load**(BankOfEEPROM As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

CALIBRATION.Save Method

Save cali bration constants to the specified bank of EEPROM.

Syntax

Function object**.CALIBRATION.Save**(BankOfEEPROM As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

BankOfEEPROM As Integer

The storage bank on EEPR OM. The valid range of the value of bank is 0 through 3.

SSI.ClearAll Method

Disconnects all of the device signals from the SSI bus trigger lines.

Syntax

Function object**.ClearAll** As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DAQError Event

Syntax

sub ControlName_DAQError (ErrString As String)

Arguments

ErrString As String
The string of error reasion

Remarks

This event will occur when some error occur in control

AiComplete Event

Syntax

sub ControlName_AiComplete(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant
The analog input data array that have been translated to the engineering data (voltage) according to AI.Channels(n).Range property.

BinaryCodes As Variant
The analog input data array with digital format.

Remarks

This event occurs when continuous analog input function is completed. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting.*"BinaryCodes without channel only"*.
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AiHalfReady Event

Syntax

sub ControlName_AiHalfReady(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant
The analog input data array that have been translated to the engineering data (voltage) according to AIRange property.

BinaryCodes As Variant
The analog input data array with digital format.

Remarks

This event occurs when one half-buffer of the circular buffer is full at continuous analog input operation. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting. "*BinaryCodes without channel only*"
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AoComplete Event

Syntax
sub ControlName _AoComplete( )

Arguments
None

Remarks
This event occurs when continuous analog output function is completed.

AoBufferReady Event

Syntax
sub ControlName_AoBufferReady(BufferIndex as Integer)

Arguments
BufferIndex as Integer,
The index of the buffer just output.

Remarks
This event occurs when enable the AO.DoubleBufferMode, If User want to dynamic change the output pattern, process it when receive AoBufferReady event.

Example
```
'This sample code show that how to output four buffers (pattern) in one channel.
'You can modify this code for dynamic changes pattern.

Dim varArray(0 To 3) As Variant
Dim nCounter As Integer

Private Sub AO_Click()

    Dim buffer0(0 To 4095) As Integer
    Dim buffer1(0 To 4095) As Integer
    Dim buffer2(0 To 4095) As Integer
    Dim buffer3(0 To 4095) As Integer

    Dim i As Double

    For i = 0 To 4095
        buffer0(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer1(i) = (&H800 + i Mod 2048) And &HFFF
        Else
            buffer1(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        End If
    Next i
    For i = 0 To 4095
        buffer2(i) = (Cos(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer3(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        Else
            buffer3(i) = (&H800 + i Mod 2048) And &HFFF
        End If
    Next i

    varArray(0) = buffer0
```

```
    varArray(1) = buffer1
    varArray(2) = buffer2
    varArray(3) = buffer3

    Daq2501.AO.GroupA.Mode = GROUP_MODE_DMA
    Daq2501.AO.GroupA.Channels(0).buffer1 = varArray(0)
    Daq2501.AO.GroupA.Channels(0).buffer2 = varArray(1)
    nCounter = 1
    Daq2501.AO.GroupA.Channels(0).Enable = True

    Daq2501.AO.DoubleBufferMode = True
    Daq2501.AO.StartContGroup(DA_Group_A_)
End Sub

Private Sub Daq2501_AoBufferReady(ByVal BufferIndex As Integer)

nCounter = nCounter + 1
nCounter = nCounter Mod 4

If BufferIndex = 1 Then
    ' It can change buffer1 in here
    Daq2501.AO.GroupA.Channels(0).buffer1 = varArray(nCounter)
End If

If BufferIndex = 2 Then
    ' It can change buffer2 in here
    Daq2501.AO.GroupA.Channels(0).buffer2 = varArray(nCounter)
End If

End Sub
```

AcquireADError Event

Acquires the offset and gain errors of the specified AI channel in the specified polarity mode.

Syntax

sub *ControlName*_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)

Arguments

***channel as Integer*,**

Indicate channel id.

polarity *as Integer,*

Indicate polarity

Value	Description
0	Unipolar
1	Bipolar

gain_err as Double

Indicate gain error

offset_err as Double

Indicate offset error

Example

```
Daq2501.CALIBRATION.DisplayErrors

Private Sub Daq2501_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
```

```
    End If
    strMsg = "AD channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub
```

AcquireDAError Event

Acquires the offset and gain err ors of the specified DA channel in the specified polarity mode.

Syntax

sub *ControlName*_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)

Arguments

***channel as Integer*,**

Indicate channel id.

polarity *as Integer,*

Indicate polarity

Value	**Description**
0	Unipolar
1	Bipolar

gain_err as Double

Indicate gain error

offset_err as Double

Indicate offset error

Example

Daq2501.CALIBRATION.DisplayErrors

```
Private Sub Daq2501_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
  List1.AddItem (strMsg)
  List1.Refresh
  End Sub
```

Daq2502 ActiveX Control

The Daq2502 ActiveX control is a software component that provides the interface for users to control PCI-2502 card.

Daq2502 ActiveX Control Overview			
Type (Group)	**Property**	**Method**	**Event**
General	DASKCardType	Open	DAQError
	CardNumber	AboutBox	
	OpenMode	ShowPropertyPages	
	DaskCardID		
DIO	P1ADir	ReadBackDOLine	
	P1BDir	ReadBackDOPort	
	P1CLowerDir	ReadDILine	
	P1CUpperDir	ReadDIPort	
		WriteDOLine	
		WriteDOPort	
GPTC	Mode	Start	
	ClockSource	Stop	
	ClockPolarity	Reset	
	GateSource	ReadStatus	
	GatePolarity		
	UpDownSource		
	UpDownPolarity		
	OutputPolarity		
	IntGATE		
	IntUpDnCTR		
	DelayCount		
	DurationCount		
	OutputValue		
AI	NumOfScan	StartContAI	AiComplete
	Channels(n).Enable	StopContAI	AiHalfReady
	ClockSource	ReadChannels	
	ScanInterval		
	SamplingInterval		
	ConversionSource		
	TriggerMode		
	TriggerSource		
	DelaySource		
	ReTriggerModeEnable		
	MCounterEnable		
	PostTriggerCount		

Daq2502 ActiveX Control Overview			
	DelayCount		
	MCount		
	ReTriggerCount		
	ExtTrigPolarity		
	ReturnType		
	DoubleBufferMode		
	StreamToFile		
	FileName		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
	Range		
	DelayCounterSource		
AO	CHUI	StartContGroup	AoComplete
	DAWRSource	StopContGroup	AoBufferReady
	TriggerSource	WriteChannel	
	TriggerMode		
	Delay1Source		
	Delay2Source		
	ExtTrigPolarity		
	ReTriggerCount		
	Delay1Count		
	Delay2Count		
	ClockSource		
	ReTriggerModeEnable		
	AIOAnalogTrigCtrl		
	AIOTrigCondition		
	AIOHLevel		
	AIOLLevel		
	DelayCounterSource		
	BreakDelayCounterSc		
	DoubleBufferMode		
	Iterations		
	Definite		
	StopMode		
	GroupA.Mode		
	GroupA.Channels(n).Enable		
	GroupA.Channels(n).IntOrExtref		
	GroupA.Channels(n).OutputPolarity		

Daq2502 ActiveX Control Overview			
	GroupA.Channels(n).RefVoltage		
	GroupA.Channels(n).Buffer1		
	GroupA.Channels(n).Buffer2		
	GroupB.Mode		
	GroupB.Channels(n).Enable		
	GroupB.Channels(n).IntOrExtref		
	GroupB.Channels(n).OutputPolarity		
	GroupB.Channels(n).RefVoltage		
	GroupB.Channels(n).Buffer1		
	GroupB.Channels(n).Buffer2		
CALIBRATION	BankTemperature	AutoCalibration	AcquireADError
	BankDate	Load	AcquireDAError
	CurrentTemperature	Save	
	CurrentDate	DisplayErrors	
SSI	TIMEBASE	ClearAll	
	ADCONV		
	DAWR		
	ADTRIG		
	DATRIG		

DASKCardType Property

Return a value that determines the card type. It is always DAQ_2502 in DAQ-2502 device.

.

Syntax

[Integer] =object**.CardType**

Remarks

Always return DAQ_2502 (for DAQ -2502 Device)

Settings

Value	Constant	Description
8	DAQ_2502	For DAQ-2502 Device

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Data Type

Integer

CardNumber Property

The sequence number of the card with ***the same card type*** plugged in the PCI slot.
The card sequence number setting is according to the PCI slot sequence in the mainboard.
The first card (in the most prior slot) is with CardNumber=0. For example,
If there are two DAQ -2502 cards plugged on your PC, the DAQ -2502 card in the prior slot should be registered with CardNumber =0, and the other one with CardNumber =1.

.

Syntax

object**.CardNumber** [= short]

Remarks

This property will be used when Initializes the hardware states of a DAQ -2K data acquisition card.

Data Type

Integer

OpenMode Property

Return/Set a value that determines the mode of opening device.

Syntax

object**.OpenMode** [= short]

Settings

Value	display	Description
0	Automatic	D2K-OCX will initialize by itself. Automatically open device when the control was created
1	Manual	Call object**.Open**() method to initialized D2K-OCX(open device)

Value	display	Description
2	Demonstration	D2K-OCX will provide software DAQ data to simulating hardware drivers.

Data Type
Integer

DaskCardID Property
Returns a value that determines the D2K_Register_Card() returns value, the DaskCardID is a numeric card ID that will be used by other D2K -DASK library functions.

Syntax
[short] =object**.DaskCardID**

Remarks
The range of card id is between 0 and 31.
This property will be used when combine D2K-DASK and D2K-OCX two module in one program.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example
This sample will demonstration If user want to check AI completed by itself.

```
' OCX to starting continuous AI
  Dim nDaskID as Integer
  Daq2502.Open(TRUE)
  nDaskID = Daq2502.DaskCardID

 Daq2502.AI.StartContAI

' Check AI Completed by DASK    API
Dim Stopped    As Byte
Dim AccessCnt As Long

Do While True
    D2K_AI_AsyncCheck nDaskID, Stopped, AccessCnt
    If Stopped = 1 Then
        Exit Do
    End If
Loop
MsgBox "AI Complete"
```

DIO.P1Adir Property
Return/Set a value that determines P1A port direction.

Syntax
object.**DIO.P1ADir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1Bdir Property
Return/Set a value that determines P1B port direction.

Syntax
object.**DIO.P1BDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CLowerdir Property
Return/Set a value that determines P1C lower port direction.

Syntax
object.**DIO.P1CLowerDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

DIO.P1CUpperdir Property

Return/Set a value that determines P1C upper port direction.

Syntax

object.**DIO.P1CUpperDir** [= Direction]

Settings

Value	Constant	Description
1	INPUT_PORT	Direction: input port.
2	OUTPUT_PORT	Direction: output port.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.Mode Property

Return/Set a value that determines the Timer/Counter mode.

Syntax

object.**GPTC.Counter0.Mode** [= Integer]

Settings

Value	Constant	Description
1	SimpleGatedEventCNT	
2	SinglePeriodMSR	
3	SinglePulseWidthMSR	
4	SingleGatedPulseGen	
5	SingleTrigPulseGen	
6	RetrigSinglePulseGen	
7	SingleTrigContPulseGen	
8	ContGatedPulseGen	

Remarks

Please refer to the hardware man ual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic

Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockSource Property
GPTC.Counter1.ClockSource Property

Return/Set a val ue that determines the Timer/Counter Source.

Syntax

object.**GPTC.Counter0.ClockSource** [= Integer]
object.**GPTC.Counter1.ClockSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKSRC_INT	internal time base
2	GPTC_CLKSRC_EXT	external time base from GPTC0_SRC or GPTC1_SRC pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.ClockP olarity Property
GPTC.Counter1.ClockPolarity Property

Return/Set a value that determines the Timer/Counter clock polarity

Syntax

object.**GPTC.Counter0.ClockPolarity** [= Integer]
object.**GPTC.Counter1.ClockPolarity** [= Integer]

Settings

Value	Constant	Description
1	GPTC_CLKEN_LACTIVE	Low active
2	GPTC_CLKEN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GateSource Property
GPTC.Counter1.GateSource Property

Return/Set a value that determines the Timer/Counter gate source

Syntax

object.**GPTC.Counter0.GateSource** [= Integer]
object.**GPTC.Counter1.GateSource** [= Integer]

Settings

Value	Constant	Description
1	GPTC_GATESRC_INT	gate is controlled by software
4	GPTC_GATESRC_EXT	gate is controlled by GPTC0_GATE or GPTC1_GATE pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.GatePolarity Property
GPTC.Counter1.GatePolarity Property

Return/Set a value that determines the Timer/Counter gate polarity

Syntax

object.**GPTC.Counter0.GatePolarity** [= Integer]
object.**GPTC.Counter1.GatePolarity** [= Integer]

Settings

Value	Constant	Description
2	GPTC_GATE_LACTIVE	Low active
0	GPTC_GATE_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownSource Property
GPTC.Counter1.UpDownSource Property

Return/Set a value that determines the Timer/ Counter UpDown Source

Syntax

object.**GPTC.Counter0.UpDownSource** [= Integer]
object.**GPTC.Counter1.UpDownSource** [= Integer]

Settings

Value	Constant	Description
0	GPTC_UPDOWN_SEL_INT	Up/Down controlled by software
16	GPTC_UPDOWN_SEL_EXT	Up/Down controlled by GPTC0_UPDOWN orGPTC1_UPDOWN pin

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.UpDownPolarity Property
GPTC.Counter1.UpDownPolarity Property

Return/Set a value that determines the Timer/Counter UpDown Polarity

Syntax

object.**GPTC.Counter0.UpDownPolarity** [= Integer]
object.**GPTC.Counter1.UpDownPolarity** [= Integer]

Settings

Value	Constant	Description
4	GPTC_UPDOWN_LACTIVE	Low active
0	GPTC_UPDOWN_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.OutputPolarity Property
GPTC.Counter1.OutputPolarity Property

Return/Set a value that determines the Timer/Counter Output Polarity

Syntax

object.**GPTC.Counter0.OutputPolarity** [= Integer]
object.**GPTC.Counter1.OutputPolarity** [= Integer]

Settings

Value	Constant	Description
8	GPTC_OUTPUT_LACTIVE	Low active
0	GPTC_OUTPUT_HACTIVE	High active

Please refer to the hardware manual for the mode description.

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

GPTC.Counter0.IntGATE Property
GPTC.Counter1.IntGATE Property

Return/Set a value that determines the Timer/Counter Internal gate initialized status

Syntax

object.**GPTC.Counter0.IntGATE** [= Boolean]
object.**GPTC.Counter1.IntGATE** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type

Boolean

GPTC.Counter0.IntUpDnCTR Property
GPTC.Counter1.IntUpDnCTR Property

Return/Set a value that determines the Timer/Counter internal updown counter initialized status.

Syntax

object.**GPTC.Counter0.IntUpDnCTR** [= Boolean]
object.**GPTC.Counter1.IntUpDnCTR** [= Boolean]

Settings

Value	Constant	Description
0	FALSE	
1	TRUE	

Data Type

Boolean

GPTC.Counter0.DelayCount Property
GPTC.Counter1.DelayCount Property

Return/Set a value that determines the Timer/Counter internal initial count of the GPTC or pulse delay. The counter value of load register 1 of timer/counter. The meaning for the value depends on the mode the timer /counter performs. For mode 1 to mode 3, the value is the initial count of the GPTC. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse delay.

Syntax

object.**GPTC.Counter0.DelayCount** [= Integer]
object.**GPTC.Counter1.DelayCount** [= Integer]

Data Type

Integer

GPTC.Counter0.DurationCount Property
GPTC.Counter1.DurationCount Property

Return/Set a value that determines the Timer/Counter pulse width when mode 4 to mode 8.
The counter value of load register 2 of timer/counter. For mode 1 to mode 3, the value is not used. For mode 4 to mode 8 (the pulse generation modes), the value is configures as the pulse width.

Syntax

object.**GPTC.Counter0.DurationCount** [= Integer]
object.**GPTC.Counter1.DurationCount** [= Integer]

Data Type

Integer

GPTC.Counter0.OutputValue Property
GPTC.Counter1.OutputValue Property

Returns the counter value of the specified general -purpose timer/counter.
Range: 0 through 65535

Syntax

[Integer=] object.**GPTC.Counter0.OutputValue**
[Integer=] object.**GPTC.Counter1.OutputValue**

Data Type

Integer

GPTC.CALIBRATION.BankTemperature Property

Returns a value that since user's last calibrated temperature in the EEPRON Bank .

Syntax

object**.CALIBRATION.BankTemperature**(*[BankOfEEPROM as Integer]*) As Single

Data Type

Single

GPTC.CALIBRATION.BankDate Property

Returns a value that since user's last calibrated date in the EEPRON Bank .

Syntax

object**.CALIBRATION.BankDate**(*[BankOfEEPROM as Integer]*) As String

Data Type

String

GPTC.CALIBRATION.CurrentTemperature Property

Returns a value that determines the current temperature on card.

Syntax

[Single=] object**.CALIBRATION.CurrentTemperature**

Data Type

Single

GPTC.CALIBRATION.CurrentDate Property

Returns a value that determines the current date.

Syntax

[String=] object**.CALIBRATION.CurrentDate**

Data Type

String

AI.NumOfScan Property

If double-buffered mode is disabled, the total number of scans to be performed. For double-buffered acquisition, NumOf *Scan* is the size (in samples) allocated for each channel in the circular buffer. This value must be a multiple of 2.

Syntax

object**.AI.NumOfScan** [=Long]

Remarks

Non-double-buffer mode

This value multiply the total number of scan channels is the total number of A/D conversions to be performed.

Double-buffer-mode

This value multiply the total number of scan channels is the size (in sample) of the circular buffer.

Data Type
Long

AI.ClockSource Property
The clock source (Time Base) the device selected.

Syntax
object**.AI.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ScanInterval Property
The length of the scan interval (that is, the counter value between the initiation of each scan sequence). The scan rate will be *TimeBase/ScanIntrv*. The value of *TimeBase* depends on the card type.
If the timer base is from *external*, the valid range of the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of the value is 100 through 16777215

Syntax
object**.AI.ScanInterval** [=Long]

Data Type
Long

AI.SamplingInterval Property
The length of the sample interval (that is, the counter value between each A/D conversion within a scan sequence). The A/D conversion rate will be *TimeBase/SampIntrv*. The value of *TimeBase(AI.ClockSource)* depends on the card type.
If the timer base is from *external*, the valid range of the value is 2 through 65535.
If the timer base is *Internal timer*, the valid range of the value is as follows:
DAQ-2502 : 100 through 16777215

Syntax
object**.AI.SamplingInterval** [=Long]

Data Type
Long

AI.ConversionSource Property
The A/D Conversion Source Selection.

Syntax
object.**AI.ConversionSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_ADCONVSRC_Int	Internal timer
4	DAQ2K_AI_ADCONVSRC_AFI0	From AFI0 pin
8	DAQ2K_AI_ADCONVSRC_SSI	From SSI source
12	DAQ2K_AI_ADCONVSRC_AFI1	From AFI1 pin

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerMode Property
The Trigger Mode Selection.

Syntax
object.**AI.TriggerMode** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_TRGMOD_POST	Post Trigger Mode
8	DAQ2K_AI_TRGMOD_DELAY	Delay Trigger Mode
16	DAQ2K_AI_TRGMOD_PRE	Pre-Trigger Mode
24	DAQ2K_AI_TRGMOD_MIDL	Middle-Trigger Mode

Remarks
Please refer to the hardware manual for the trigger mode description.

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.TriggerSource Property

The Trigger Source Selection.

Syntax

object.**AI.TriggerSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_AI_TRGSRC_SOFT	Software
1	DAQ2K_AI_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_AI_TRGSRC_ExtD:	From external digital trigger pin
3	DAQ2K_AI_TRSRC_SSI	From SSI source

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.DelaySource Property

The delay source sele ction.

Syntax

object.**AI. DelaySource** [=Short]

Settings

Value	Constant	Description
256	DAQ2K_AI_Dly1InSamples	Delay in samples

Data Type

Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReTriggerModeEnable Property

Re-trigger in an acquisition is enabled

Syntax

object.**AI.ReTriggerModeEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	ReTriggerMode is disabled.
1	True	ReTriggerMode is enabled.

Data Type

Boolean

AI.MCounterEnable Property

This constant is only valid for Pre -trigger and Middle trigger mode
Mcounter is enabled and then the trigger signal is ignore before M terminal count is reached.

Syntax

object.**AI.MCounterEnable** [=Boolean]

Settings

Value	Constant	Description
0	False	MCounter is disabled.
1	True	Mcounter is enabled.

Data Type

Boolean

AI.PostTriggerCount Property

This constant is only valid for Middle trigger mode, The PostTriggerCount indicates the number of data will be accessed after a specific trigger event.

Syntax

object.**AI.PostTriggerCount** [=Long]

Data Type

Long

AI.DelayCount Property

This constant is only valid for Delay trigger mode,
The DelayCount indicates the number of data or timer ticks will be ignor ed after a specific trigger event.

Syntax

object.**AI.DelayCount** [=Long]

Data Type

Long

AI.MCount Property

The counter value of MCounter . This argument is only valid for Pretrigger and Middle trigger mode.

Syntax
object.**AI.MCount** [=Long]

Data Type
Long

AI.ReTriggerCount Property
The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax
object.**AI.ReTriggerCount** [=Long]

Data Type
Long

AI.ExtTrigPolarity Property
External Digital Trigger Polarity.

Syntax
object.**AI.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_AI_TrgPositive	Trigger positive edge active
4096	DAQ2K_AI_TrgNegative	Trigger negative edge active

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.ReturnType Property
Return/Set a value that determines the return data type of analog input when AiComplete or AiHalfReady event would occur.

Syntax
object.**AI.ReturnType** [=Integer]

Settings

Value	Constant	Description
0		Scaled data only
1		Binary codes without channel only
2		Scaled data and binary codes without channel
3		No data return

Data Type
Integer

AI.DoubleBufferMode Property

Enables or disables double-buffered data acquisition mode.

Syntax
object.**AI.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type
Boolean

AI.StreamToFile Property

Return/Set a value that determines if the control is enabled the function of streaming data to disk file. This argument is only valid for Delay trigger

Syntax
object.**AI.StreamToFile** [=Boolean]

Settings

Value	Constant	Description
0	False	Disable the function of streaming data to disk file.
1	True	Enable the function of streaming data to disk file

Data Type
Boolean

AI.FileName Property

FileName specified the file name of streaming data to disk. This argument is only valid for AI.StreamToFile is Enable.

Syntax
object.**AI.FileName** [=String]

Data Type
String

AI.AIOAnalogTrigCtrl Property

Return/Set a value that determines the trigger source selection

Syntax
object.**AI.AIOAnalogTrigCtrl** [=Integer]

Settings

Value	Constant	Description
0	ADCATRIG	The first AI channel in the channel-gain queue
1	EXTATRIG	From external analog trigger pin

Data Type
Integer

AI.AIOTrigCondition Property

Return/Set a value that determines the t rigger condition selection

Syntax
object.**AI.AIOTrigCondition** [=Integer]

Settings

Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above -High-Level Triggering
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks
Please refer to the hardware manual for the more description.

Data Type
Integer

AI.AIOHLevel Property

The High value setting of Trigger level. Please refer to the hardware manual for the relations hip between the value of *TrgLevel* and trigger voltage.

Syntax
object.**AI.AIOHLevel** [=Long]

Remarks
***For example*:**
If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AI.AIOLLevel Property

The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax
object.**AI.AIOLLevel** [=Long]

Remarks

***For example*:**

If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type

Long

AI.Channels(0).Enable Property
AI.Channels(1).Enable Property
AI.Channels(2).Enable Property
AI.Channels(3).Enable Property

DAQ-2000 channel-gain that can be set separately for each channel to perform multi-channel/gain analog input, The parameter 0~3 is channel id.

Syntax

object.**AI.Channels**(0).Enable [=Boolean]

Data Type

Boolean

AI.Range Property

This property can setting same range for al l channels.

Syntax

object.**AI.Range** [=Integer]

Settings

Value	Constant	Description
1	AD_B_10_V	Bipolar -10V to +10V
15	AD_U_10_V	Unipolar 0 to +10V

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AI.DelayCounterSource Property

A/D Delay Counter Source Selection

Syntax

object.**AI.DelayCounterSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_AI_DTSRC_Int	Internal timer
16	DAQ2K_AI_DTSRC_AFI1	From AFI1 pin
32	DAQ2K_AI_DTSRC_GPTC0	From GPTC0_OUT
48	DAQ2K_AI_DTSRC_GPTC1	From GPTC1_OUT

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.CHUI Property

The length of the Channel Update interval (that is, the counter value between the initiation of each update sequence).
If the timer base is from *external*, the valid range of the value is 2 through 16777215.
If the timer base is *Internal timer*, the valid range of th e value is as follows:
Range: 40 through 16777215

Syntax

object.**AO.CHUI** [=Long]

Data Type

Long

AO.DAWRSource Property

Return/Set a value that determines the D/A R/W Source Selection

Syntax

object.**AO.DAWRSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_WRSRC_Int	Internal timer
1	DAQ2K_DA_WRSRC_AFI0	From AFI0 pin
2	DAQ2K_DA_WRSRC_SSI	From SSI source

Data Type

Integer

Microsoft C/C++ and Borland C++

Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerSource Property

Return/Set a value that determines the trigger source selection

Syntax

object**.AO.TriggerSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGSRC_SOFT	Software
1	DAQ2K_DA_TRGSRC_ANA	From analog trigger pin
2	DAQ2K_DA_TRGSRC_ExtD	From external digital trigger pin
3	DAQ2K_DA_TRSRC_SSI	From SSI source

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.TriggerMode Property

Return/Set a value that determines the trigger mode selection

Syntax

object**.AO.TriggerMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TRGMOD_POST	Post Trigger Mode
1	DAQ2K_DA_TRGMOD_DELAY	Delay Trigger Mode

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay1Source Property

Return/Set a value that determines the delay1 source selection

Syntax

object.**AO.Delay1Source** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_Dly1InTimebase	Delay in time base

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.Delay2Source Property

Return/Set a value that determines the delay2 source selection

Syntax

object.**AO.Delay2Source** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_Dly2InTimebase	Delay in time base

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ExtTrigPolarity Property

Return/Set a value that determines the e xternal digital trigger polarity selection

Syntax

object.**AO.ExtTrigPolarity** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TrgPositive	Trigger positive edge active
512	DAQ2K_DA_TrgNegative	Trigger negative edge active

Data Type
Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerCount Property

The accepted trigger times in an acquisition. This argument is only valid for Delay trigger and Post trigger mode.

Syntax
object.**AO.ReTriggerCount** [=Long]

Data Type
Long

AO.Delay1Count Property

The counter value of DLY1 Counter (the delay time after the trigger signal to the start of the waveform generation) . This argument is only valid for Delay trigger mode.

Syntax
object.**AO.Delay1Count** [=Long]

Data Type
Long

AO.Delay2Count Property

The counter value of DLY2 Counter (the Delay between two consecutive waveform generations).

Syntax
object.**AO.Delay2Count** [=Long]

Data Type
Long

AO.ClockSource Property

The clock source (Time Base) the device selected.

Syntax
object.**AO.ClockSource** [=Short]

Settings

Value	Constant	Description
0	DAQ2K_IntTimeBase	Internal timer as the time base
1	DAQ2K_ExtTimeBase	External timer as the time base

Value	Constant	Description
2	DAQ2K_SSITimeBase	The timer based on the SSI source

Data Type
Short

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.ReTriggerModeEnable Property
Re-trigger in an acquisition is enabled

Syntax
object**.AO.ClockSource** [=Boolean]

Settings
Value	Constant	Description
0	False	ReTriggerMode is disabled
1	True	ReTriggerMode is enabled

Data Type
Boolean

AO.AIOAnalogTrigCtrl Property
Return/Set a value that determines the trigger source selection

Syntax
object**.AO.AIOAnalogTrigCtrl** [=Integer]

Settings
Value	Constant	Description
0	ADCATRIG	The first AI channel in the channel-gain queue
1	EXTATRIG	From external analog trigger pin

Data Type
Integer

AO.AIOTrigCondition Property
Return/Set a value that determines the trigger condition selection

Syntax
object**.AO.AIOTrigCondition** [=Integer]

Settings
Value	Constant	Description
0	Below_Low_level	Below-Low-Level Triggering
256	Above_High_Level	AI channel 1Above-High-Level Triggering

Value	Constant	Description
512	Inside_Region	Inside Region Triggering
768	High_Hysteresis	High Hysteresis Triggering
1024	Low_Hysteresis	Low Hysteresis Triggering

Remarks
Please refer to the hardware manual for the more description.

Data Type
Integer

AO.AIOHLevel Property
The High value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltage.

Syntax
object.**AO.AIOHLevel** [=Long]

Remarks
***For example*:**
If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AO.AIOLLevel Property
The Low value setting of Trigger level. Please refer to the hardware manual for the relationship between the value of *TrgLevel* and trigger voltag e.

Syntax
object.**AO.AIOLLevel** [=Long]

Remarks
***For example*:**
If the trigger voltage is ±10V, the relationship between the value of *TrgLevel* and trigger voltage is as the following table:

Trigger Level digital setting	trigger voltage
0xFF	+9.92V
0x81	+0.08V
0x80	0
0x7F	-0.08V
0x01	-10V

Data Type
Long

AO.DelayCounterSource Property

D/A Trigger delay Counter Source Selection

Syntax

object.**AO.DelayCounterSource** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TDSRC_Int	Internal timer
16	DAQ2K_DA_TDSRC_AFI0	From AFI1 pin
32	DAQ2K_DA_TDSRC_GPTC0	From GPTC0_OUT
48	DAQ2K_DA_TDSRC_GPTC1	From GPTC1_OUT

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.BreakDelayCounterSc Property

D/A Break delay Counter Source Selection

Syntax

object.**AO.BreakDelayCounterSc** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_BDSRC_Int	Internal timer
16	DAQ2K_DA_BDSRC_AFI0	From AFI1 pin
32	DAQ2K_DA_BDSRC_GPTC0	From GPTC0_OUT
48	DAQ2K_DA_BDSRC_GPTC1	From GPTC1_OUT

Data Type

Integer

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

AO.DoubleBufferMode Property

Enables or disables double-buffered data acquisition mode.

Syntax

object.**AO.DoubleBufferMode** [=Boolean]

Settings

Value	Constant	Description
0	False	double-buffered mode is disabled.
1	True	double-buffered mode is enabled.

Data Type

Boolean

AO.Iterations Property

The times of number of the data in the buffer to output to the port. a value of zero is not allowed.

Syntax

object.**AO.Iterations** [=Long]

Data Type

Long

AO.Definite Property

Waveform generation proceeds definite or indefinitely. If double -buffered mode is enabled, this parameter is of no use.

Syntax

object.**AO.Definite** [=Boolean]

Settings

Value	Constant	Description
0	False	Indefinitely
1	True	Definite

Data Type

Boolean

AO.StopMode Property

Return/Set a value that determines DA tra nsfer termination mode selected.

Syntax

object.**AO. StopMode** [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_TerminateImmediate	Software terminate the DA continuous operation immediately
1	DAQ2K_DA_TerminateUC	Software terminate the DA continuous operation on next update counter terminal count
2	DAQ2K_DA_TerminateIC	Software terminate the DA continuous operation on iteration count

Remarks

Please refer to the hardware manual for the more description.

Data Type
Integer

AO.GroupA.Mode Property

Return/Set a value that determines GroupA DA transfer mode selected.

Syntax
object.**AO.GroupA.Mode** [=Integer]

Settings

Value	Constant	Description
0	GROUP_MODE_DMA	Using DMA to transfer data
1	GROUP_MODE_FIFO	The data will be loaded into on-board DA FIFO(8K samples)

Remarks
Please refer to the hardware manual for the more description.

Data Type
Integer

AO.GroupA.Channels(0).Enable Property
AO.GroupA.Channels(1).Enable Property
AO.GroupA.Channels(2).Enable Property
AO.GroupA.Channels(3).Enable Property

DAQ-2000 output channel that can be set separately for each channel to perform multi-channel analog input, The parameter 0~3 is channel id.

Syntax
object.**AO.GroupA.Channels**(0).Enable [=Boolean]

Data Type
Boolean

AO.GroupA.Channels(0).OutputPolarity Property
AO.GroupA.Channels(1).OutputPolarity Property
AO.GroupA.Channels(2).OutputPolarity Property
AO.GroupA.Channels(3).OutputPolarity Property

Return/Set a value that determines polarity (unipolar or bipolar) of the output channel.

Syntax
object.**AO.GroupA.Channels**(0).OutputPolarity [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_UniPolar	Unipolar
1	DAQ2K_DA_BiPolar	Bipolar

Data Type

Integer

AO.GroupA.Channels(0).IntOrExtref Property
AO.GroupA.Channels(1).IntOrExtre f Property
AO.GroupA.Channels(2).IntOrExtref Property
AO.GroupA.Channels(3).IntOrExtref Property

Return/Set a value that determines DA reference voltage source of the output channel.

Syntax

object.**AO.GroupA.Channels**(0). IntOrExtref [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_Int_REF	internal reference
1	DAQ2K_DA_Ext_REF	external reference

Data Type

Integer

AO.GroupA.Channels(0).RefVoltage Property
AO.GroupA.Channels(1).RefVoltage Property
AO.GroupA.Channels(2).RefVoltage Property
AO.GroupA.Channels(3).RefVoltage Property

If the D/A reference voltage source your device use is internal reference, the valid values is 10.
If the D/A reference voltage source your device use is external reference, the valid range is -10 to +10.

Syntax

object.**AO.GroupA.Channels**(0).RefVoltage [=Single]

Data Type

Single

AO.GroupA.Channels(0).Buffer1 Property
AO.GroupA.Channels(1).Buffer1 Property
AO.GroupA.Channels(2).Buffer1 Property
AO.GroupA.Channels(3).Buffer1 Property

This property set up the buffer for continuous analog output operation.
A buffer data or a array of buffer data, data type is integer.

Syntax

object.**AO.GroupA.Channels**(0).Buffer1 [=Variant]

Remarks

You must assign this property before call AO.StartContGroup method.
This property will be used when single or double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type

Variant(Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
       buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i
 Daq2502.AO.GroupA.Channels(0).buffer1 = buffer1
 Daq2502.AO.GroupA.Channels(0).Enable = True
 Daq2502.AO.StartContGroup (DA_Group_A)
```

AO.GroupA.Channels(0).Buffer2 Property
AO.GroupA.Channels(1).Buffer2 Property
AO.GroupA.Channels(2).Buffer2 Property
AO.GroupA.Channels(3).Buffer2 Property

This property set up the buffer for continuous analog output operation.
A buffer data or a array of buffer data, data type is integer.
This property only available when double buffer mode.

Syntax

object.**AO.GroupA.Channels**(0).Buffer2 [=Variant]

Remarks

You must assign this property before call AO.StartContGroup method.
This property will be used when double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type

Variant(Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i

 For i = 0 To 4095
      If i < 2048 Then
           buffer2(i) = (&H800 + i Mod 2048) And &HFFF
      Else
           buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
      End If
 Next i

 Daq2502.AO.GroupA.Channels(0).buffer1 = buffer1
 Daq2502.AO.GroupA.Channels(0).buffer2 = buffer2
 Daq2502.AO.GroupA.Channels(0).Enable = True
 Daq2502.AO.DoubleBufferMode = True
 Daq2502.AO.StartContGroup(DA_Group_A)
```

AO.GroupB.Mode Property

Return/Set a value that determines GroupB DA trans fer mode selected.

Syntax

object.**AO.GroupB.Mode** [=Integer]

Settings

Value	Constant	Description
0	GROUP_MODE_DMA	Using DMA to transfer data
1	GROUP_MODE_FIFO	The data will be loaded into on-board DA FIFO(8K samples)

Remarks

Please refer to the har dware manual for the more description.

Data Type

Integer

AO.GroupB.Channels(4).Enable Property
AO.GroupB.Channels(5).Enable Property
AO.GroupB.Channels(6).Enable Property
AO.GroupB.Channels(7).Enable Property

DAQ-2000 output channel that can be set se parately for each channel to perform multi-channel analog input, The parameter 4~7 is channel id.

Syntax

object.**AO.GroupB.Channels**(4).Enable [=Boolean]

Data Type
Boolean

AO.GroupB.Channels(4).OutputPolarity Property
AO.GroupB.Channels(5).OutputPola rity Property
AO.GroupB.Channels(6).OutputPolarity Property
AO.GroupB.Channels(7).OutputPolarity Property

Return/Set a value that determines polarity (unipolar or bipolar) of the output channel.

Syntax
object.**AO.GroupB.Channels**(4).OutputPolarity [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_UniPolar	Unipolar
1	DAQ2K_DA_BiPolar	Bipolar

Data Type
Integer

AO.GroupB.Channels(4).IntOrExtref Property
AO.GroupB.Channels(5).IntOrExtref Property
AO.GroupB.Channels(6).IntOrExtref Property
AO.GroupB.Channels(7).IntOrExtref Property

Return/Set a value that determines DA reference voltage source of the output channel.

Syntax
object.**AO.GroupB.Channels**(4). IntOrExtref [=Integer]

Settings

Value	Constant	Description
0	DAQ2K_DA_Int_REF	internal reference
1	DAQ2K_DA_Ext_REF	external reference

Data Type
Integer

AO.GroupB.Channels(4).RefVoltage Property
AO.GroupB.Channels(5).RefVoltage Property
AO.GroupB.Channels(6).RefVoltage Property
AO.GroupB.Channels(7).RefVoltage Property

If the D/A refer ence voltage source your device use is internal reference, the valid values is 10.

If the D/A reference voltage source your device use is external reference, the valid range is -10 to +10.

Syntax

object.**AO.GroupB.Channels**(4).RefVoltage [=Single]

Data Type

Single

AO.GroupB.Channels(4).Buffer1 Property
AO.GroupB.Channels(5).Buffer1 Property
AO.GroupB.Channels(6).Buffer1 Property
AO.GroupB.Channels(7).Buffer1 Property

This property set up the buffer for continuous analog output operation.
A buffer data or a array of buffer data, data type is integer.

Syntax

object.**AO.GroupB.Channels**(4).Buffer1 [=Variant]

Remarks

You must assign this property before call AO.StartContGroup method. This property will be used when single or double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type

Variant (Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i
 Daq2502.AO.GroupB.Channels(4).buffer1 = buffer1
 Daq2502.AO.GroupB.Channels(4).Enable = True
 Daq2502.AO.StartContGroup(DA_Group_B)
```

AO.GroupB.Channels(4).Buffer2 Property
AO.GroupB.Channels(5).Buffer2 Property
AO.GroupB.Channels(6).Buffer2 Property
AO.GroupB.Channels(7).Buffer2 Property

This property set up the buffer for continuous analog output operation.
A buffer data or a array of buffer data, data type is integer.
This property only available when double buffer mode.

Syntax
object.**AO.GroupB.Channels**(4).Buffer2 [=Variant]

Remarks
You must assign this property before call AO.StartContGroup method.
This property will be used when double buffer mode.
in VC++, Buffer is a VT_ARRAY | VT_I4.

Data Type
Variant(Integer array)

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double

 For i = 0 To 4095
      buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
 Next i

 For i = 0 To 4095
      If i < 2048 Then
           buffer2(i) = (&H800 + i Mod 2048) And &HFFF
      Else
           buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
      End If
 Next i

 Daq2502.AO.GroupB.Channels(4).buffer1 = buffer1
 Daq2502.AO.GroupB.Channels(4).buffer2 = buffer2
 Daq2502.AO.GroupB.Channels(4).Enable = True
 Daq2502.AO.DoubleBufferMode = True
 Daq2502.AO.StartContGroup(DA_Group_B)
```

SSI.ADCONV Property
Connect / Disconnect a SSI_ADCONV device signal to the specified SSI bus trigger line.

Syntax
object.**SSI.ADCONV** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.ADTRIG Property
Connect / Disconnect a SSI_ADTRIG device signal to the specified SSI bus trigger line.

Syntax
object.**SSI. ADTRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.DATRIG Property
Connect / Disconnect a SSI_DATRIG device signal to the specified SSI bus trigger line.

Syntax
object.**SSI.DATRIG** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.DAWR Property
Connect / Disconnect a SSI_DAWR device signal to the specified SSI bus trigger line.

Syntax
object.**SSI.DAWR** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

SSI.TIMEBASE Property
Connect / Disconnect a SSI_TIMEBASE device signal to the specified SSI bus trigger line.

Syntax
object.**SSI.TIMEBASE** [=Boolean]

Settings

Value	Constant	Description
0	False	Disconnect to the specified SSI bus trigger line
1	True	Connect to the specified SSI bus trigger line

Data Type
Boolean

Open Method

Syntax

Function object**.Open** ([ErrMsgBox As Variant]) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

[ErrMsgBox As Variant]

Boolean type.
True: It will popup error message dialog box when operation error.
False: It will fire DAQError event instead of popping up dialog when operation error.

Remarks

This method will be used when the OpenMode property is Manual.

Note

In VC++, *ErrMsgBox* is a VARIANT of VT_I2.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

ShowPropertyPages Method

This method will show propertypages of ActiveX Control.

Syntax

Function object***.ShowPropertyPages***() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AboutBox Method

This method will show About ADLINK dialog box.

Syntax

Function object**.AboutBox**( ) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DIO.ReadDIPort Method

Syntax

Function object**.DIO.ReadDIPort** (port As Integer, value as Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Returns the digital data read from the specified port. The returned value is 8 -bit data.

Remarks

You can read data from the digital input port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.ReadDILine Method

Syntax

Function object**.DIO.ReadDILine** (port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

The digital lin e to be read. The valid value is 0 through 7

value As Variant

Returns the digital logic state, 0 or 1, of the specified line.

Remarks

You can read data from the digital input port.

Note

In VC++, value is a VARIANT of VT_I2.

DIO.WriteDOPort Method

Syntax

Function object**.DIO.WriteDOPort** (port As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for inputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port

Value	Constant	Description
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value as Variant

8-bit data that will be written to the digital output port.

Remarks

Users can write data to the digital output port.

Note

In VC++, value is a VARIANT of VT_I4.

DIO.WriteDOLine Method

Syntax

Function object.**DIO.WriteDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

Selects one line number from the indicated port: from 0 to 7 (8-bit port).

value As Variant

Sets 0 or 1 to the indicated line.

Note

In VC++, value is a VARIANT of VT_I2.

DIO.ReadBackDOPort Method

Reads back data from the indicated digital output port.

Syntax

Function object.**DIO.ReadBackDOPort** (port As Integer, value As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

value As Variant

Data that is read back from the indicated port.

Note
In VC++, value is a VARIANT of VT_I4.

DIO.ReadBackDOLine Method
Reads back data from the indicated digital output line of the indicated digital output port.

Syntax
Function object.**DIO.ReadBackDOLine**(port As Integer, line As Integer, value As Variant) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
port As Integer

Selects one port for outputting digital data.

Value	Constant	Description
0	Channel_P1A	P1A Port
1	Channel_P1B	P1B Port
2	Channel_P1C	P1C Port
3	Channel_P1CL	P1C Lower Port
4	Channel_P1CH	P1C Higher Port

line As Integer

Selects one line number from the indicated port: from 0 to 7 (8-bit port).

value As Variant

Data that is read back from the indicated line.

Note
In VC++, value is a VARIANT of VT_I2.

GPTC.Counter0.Start Method
GPTC.Counter1.Start Method
Start counter ope ration with the specified mode.

Syntax
Function object.**GPTC.Counter0.Start**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Remarks
You can start the indicated counter to operate in the specified mode.

GPTC.Counter0.Stop Method
GPTC.Counter1.Stop Method
Stop counter operation.

Syntax
Function object.**GPTC.Counter0.Stop**() As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

GPTC.Counter0.Reset Method
GPTC.Counter1.Reset Method

Halts the specified general -purpose timer/counter operation and reload the initial value of the timer/counter.

Syntax

Function object.**GPTC.Counter0.Reset**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

GPTC.Counter0.ReadStatus Method
GPTC.Counter1.ReadStatus Method

Reads the latched GPTC status of the general -purpose counter from the GPTC status register.

Syntax

Function object.**GPTC.Counter0.ReadStatus**(Clock *As Integer, Output As Integer, Gate As Integer, Updown As Integer,*) As Boolean

Return Value

True if the function is successful; otherwise False.

AI.StartContAI Method

This method performs continuous A/D conversions on the specified analog input channels at a rate as close to the rate you specifie d. This method takes advantage of the DAQ-2000 channel-gain that can be set separately for each channel to perform multi-channel/gain analog input. This method will fire AiComplete or AiHalfReady event depends on AI.DoubleBufferMode property.

Syntax

Function object.**AI.StartContAI**() As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

Remarks

You can use this method to start the DMA analog input function. If the AI.StreamToFile property is True then the DMA data will be written to the file specified by AI.FileName. Otherwise, the AI.FileName property will be ignored. The data file is written in binary format, with the lower byte first (little endian). Data type is “Binary codes with miscellaneous data”. DAQBench provides a convenient tool DAQCvt to convert the binary file to the file format read easily. See DAQBench User’s Guide for the usage of the utility. If you want to handle the data by yourself, please refer to Appendix D Data File Format for the file structure.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Daq2502.AI.Channels(0).Enable = True
Daq2502.AI.Channels(2).Enable = True
Daq2502.AI.Range = AD_B_10_V

Daq2502.AI.StartContAI

Private Sub Daq2502_AiComplete(ScaledData As Variant, BinaryCodes As Variant)
    ' Get Data in    ScaledData
End Sub

Private Sub Daq2502_AiHalfReady(ScaledData As Variant, BinaryCodes As Variant)
    ' Get Data in    ScaledData
End Sub
```

AI.StopContAI Method

You can use this method to force stop analog input.

Syntax

Function object**.AI.StopContAI** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

AI.ReadChannels Method

This method performs a sof tware triggered A/D conversion (analog input) on analog input channels and returns the values converted.

Syntax

Function object**.AI.ReadChannels**(Buffer As Variant) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Buffer As Variant

An integer array to contain the acquired data. Please refer to Appendix C AD Data Format for the data format in the Buffer.

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Daq2502.AI.Channels(0).Enable = True
Daq2502.AI.Channels(2).Enable = True
Daq2502.AI.Range = AD_B_10_V
```

```
Private Sub Timer1_Timer()
    Dim vBuffer As Variant
    Daq2502.AI.ReadChannels vBuffer
    ' Get Data in    vBuffer
End Sub
```

AO.StartContGroup Method

This method performs continuous D/A conversions on the specified analog output channel at a rate as close to the rate you specified.
This method will fire AoComplete or AoBufferReady event depends on AO.DoubleBufferMode property.

Syntax

Function object**.AO.StartContGroup(** GroupID As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

GroupID As Integer
The group of analog output channels.

Value	Constant	Description
0	DA_Group_A	0-3 channels belong to Group A
4	DA_Group_B	4-7 channels belong to Group A
8	DA_Group_AB	0-7 channels belong to Group A B

Microsoft C/C++ and Borland C++
Header: D2kDask.h

Visual Basic
Module: D2kDask.bas

Borland Delphi
Unit: D2kDask.pas

Example

```
Dim buffer1(0 To 4095) As Integer
Dim buffer2(0 To 4095) As Integer

Dim i As Double

  For i = 0 To 4095
        buffer1(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
  Next i

  For i = 0 To 4095
        If i < 2048 Then
              buffer2(i) = (&H800 + i Mod 2048) And &HFFF
        Else
              buffer2(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        End If
  Next i

  Daq2502.AO.GroupA.Channels(0).buffer1 = buffer1
  Daq2502.AO.GroupA.Channels(0).buffer2 = buffer2
  Daq2502.AO.GroupA.Channels(0).Enable = True
  Daq2502.AO.DoubleBufferMode = True

  Daq2502.AO.StartContGroup(DA_Group_A)
```

AO.StopContGroup Method

You can use this method to force stop analog output.

Syntax

Function object.**AO.StopContGroup**(GroupID As Integer) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

GroupID As Integer

The group of analog output channels .

Value	Constant	Description
0	DA_Group_A	0-3 channels belong to Group A
4	DA_Group_B	4-7 channels belong to Group A
8	DA_Group_AB	0-7 channels belong to Group A B

AO.WriteChannel Method

Accepts a voltage value (or a current value), scales it to the proper binary value and writes a binary value to the specified analog output channel.

Syntax

Function object.**AO.WriteChannel**(Channel as Integer, Voltage as Single) As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

Channel as Integer

The analog output channel number.

Voltage as Single

The value to be scaled and written to the analog output channel. The range of voltages depends on the type of device, on the output polarity, and on the voltage reference (external or internal).

CALIBRATION.AutoCalibration Method

Uses this method to calibrate your DAQ -2000 device. When the method is called, the device goes into a self -calibration cycle. The method does not return until the self -calibration is completed.

Syntax

Function object.**CALIBRATION.AutoCalibration** () As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

CALIBRATION.DisplayErrors Method

Uses this method to fire AcquireADError and AcquireDAError events. Through those events user can identification DAQ-2000 device current status.

Syntax

Function object.**CALIBRATION.DisplayErrors** () As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
None

Example
Daq2502.CALIBRATION.DisplayErrors

```
Private Sub Daq2502_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "BiPolar"
    End If
    strMsg = "AD channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub

Private Sub Daq2502_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
    List1.AddItem (strMsg)
    List1.Refresh
End Sub
```

CALIBRATION.Load Method
Load calibration constants from the specified bank of EEPROM.

Syntax
Function object**.CALIBRATION.Load**(BankOfEEPROM As Integer) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

CALIBRATION.Save Method
Save calibration constants to the specified bank of EEPROM.

Syntax
Function object**.CALIBRATION.Save**(BankOfEEPROM As Integer) As Boolean

Return Value
True if the function is successful; otherwise False.

Arguments
BankOfEEPROM As Integer

The storage bank on EEPROM. The valid range of the value of bank is 0 through 3.

SSI.ClearAll Method

Disconnects all of the device signals from the SSI bus trigger lines.

Syntax

Function object**.ClearAll** As Boolean

Return Value

True if the function is successful; otherwise False.

Arguments

None

DAQError Event

Syntax

sub ControlName_DAQError (ErrString As String)

Arguments

ErrString As String
The string of error reasion

Remarks

This event will occur when some error occur in control

AiComplete Event

Syntax

sub ControlName_AiComplete(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant
The analog input data array that have been translated to the engineering data (voltage) according to AI.Channels(n).Range property.

BinaryCodes As Variant
The analog input data array with digital format.

Remarks

This event occurs when continuous analog input function is completed. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting.*"BinaryCodes without channel only".*
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AiHalfReady Event

Syntax

sub ControlName_AiHalfReady(ScaledData as Variant, BinaryCodes As Variant)

Arguments

ScaledData as Variant
The analog input data array that have been translated to the engineering data (voltage) according to AIRange property.

BinaryCodes As Variant
The analog input data array with digital format.

Remarks

This event occurs when one half-buffer of the circular buffer is full at continuous analog input operation. Whether ScaledData or BinaryCodes contains data depends on AI.ReturnType property setting. "*BinaryCodes without channel only*"
Format: Please refer to Appendix C AD Data Format for the data format in the Buffer.

Note

In VC++, ScaledData is a VARIANT of VT_ARRAY | VT_R4, BinaryCodes is a VARIANT of VT_ARRAY | VT_I4 (without channel).

AoComplete Event

Syntax

sub ControlName _AoComplete( )

Arguments

None

Remarks

This event occurs when contin uous analog output function is completed.

AoBufferReady Event

Syntax

sub ControlName_AoBufferReady(BufferIndex as Integer)

Arguments

BufferIndex as Integer, The index of the buffer just output.

Remarks

This event occurs when enable the AO.DoubleBufferMode, If User want to dynamic change the output pattern, process it when receive AoBufferReady event.

Example

```
'This sample code show that how to output four buffers (pattern) in one channel.
'You can modify this code for dynamic changes pattern.

Dim varArray(0 To 3) As Variant
Dim nCounter As Integer

Private Sub AO_Click()

    Dim buffer0(0 To 4095) As Integer
    Dim buffer1(0 To 4095) As Integer
    Dim buffer2(0 To 4095) As Integer
    Dim buffer3(0 To 4095) As Integer

    Dim i As Double

    For i = 0 To 4095
        buffer0(i) = (Sin(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer1(i) = (&H800 + i Mod 2048) And &HFFF
        Else
            buffer1(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        End If
    Next i
    For i = 0 To 4095
        buffer2(i) = (Cos(i / 512 * 3.14159) * &H7FF) + &H800
    Next i
    For i = 0 To 4095
        If i < 2048 Then
            buffer3(i) = (&H800 + 2047 - (i Mod 2048)) And &HFFF
        Else
            buffer3(i) = (&H800 + i Mod 2048) And &HFFF
        End If
    Next i

    varArray(0) = buffer0
    varArray(1) = buffer1
    varArray(2) = buffer2
    varArray(3) = buffer3
```

```
    Daq2502.AO.GroupA.Mode = GROUP_MODE_DMA
    Daq2502.AO.GroupA.Channels(0).buffer1 = varArray(0)
    Daq2502.AO.GroupA.Channels(0).buffer2 = varArray(1)
    nCounter = 1
    Daq2502.AO.GroupA.Channels(0).Enable = True

    Daq2502.AO.DoubleBufferMode = True
    Daq2502.AO.StartContGroup(DA_Group_A)
End Sub

Private Sub Daq2502_AoBufferReady(ByVal BufferIndex As Integer)

nCounter = nCounter + 1
nCounter = nCounter Mod 4

If BufferIndex = 1 Then
    ' It can change buffer1 in here
    Daq2502.AO.GroupA.Channels(0).buffer1 = varArray(nCounter)
End If

If BufferIndex = 2 Then
    ' It can change buffer2 in here
    Daq2502.AO.GroupA.Channels(0).buffer2 = varArray(nCounter)
End If

End Sub
```

AcquireADError Event

Acquires the offset and gain errors of the specified AI channel in the specified polarity mode.

Syntax

sub *ControlName*_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)

Arguments

***channel as Integer*,**

Indicate channel id.

polarity *as Integer,*

Indicate polarity

Value	Description
0	Unipolar
1	Bipolar

gain_err as Double

Indicate gain error

offset_err as Double

Indicate offset error

Example

```
Daq2502.CALIBRATION.DisplayErrors

Private Sub Daq2502_AcquireADError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err
As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If
    strMsg = "AD channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####")
& " Offset error:" & Format(offset_err, "#0.#####")
```

```
    List1.AddItem (strMsg)
    List1.Refresh
End Sub
```

AcquireDAError Event

Acquires the offset and gain errors of the specified DA channel in the specified polarity mode.

Syntax

sub *ControlName*_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)

Arguments

***channel as Integer*,**

Indicate channel id.

polarity *as Integer,*

Indicate polarity

Value	Description
0	Unipolar
1	Bipolar

gain_err as Double

Indicate gain error

offset_err as Double

Indicate offset error

Example

Daq2502.CALIBRATION.DisplayErrors

```
Private Sub Daq2502_AcquireDAError(ByVal channel As Integer, ByVal polarity As Integer, ByVal gain_err As Double, ByVal offset_err As Double)
    If polarity = 0 Then
        strPolarity = "Unipolar"
    Else
        strPolarity = "Bipolar"
    End If

    strMsg = "DA channel:" & channel & " " & strPolarity & " Gain error:" & Format(gain_err, "#0.#####") & " Offset error:" & Format(offset_err, "#0.#####")
  List1.AddItem (strMsg)
  List1.Refresh
  End
Sub
```

Appendix A Status Codes

This appendix lists the status codes returned by D2K-DASK, including the name and description.

Each D2K-DASK function returns a status code that indicates whether the function was performed successfully. When a D2K-DASK function returns a negative number, it means that an error occurred while executing the function.

Status Code	Status Name	Description
0	NoError	No error occurred
-1	ErrorUnknownCardType	The C*ardType* argument is not valid
-2	ErrorInvalidCardNumber	The CardNumber argument is out of range (larger than 31).
-3	ErrorTooManyCardRegistered	There have been 32 cards that were registered.
-4	ErrorCardNotRegistered	No card registered as id *CardNumber*.
-5	ErrorFuncNotSupport	The function called is not supported by this type of card..
-6	ErrorInvalidIoChannel	The specified *Channel* or *Port* argument is out of range..
-7	ErrorInvalidAdRange	The specified analog input range is invalid.
-8	ErrorContIoNotAllowed	The specified continuous IO operation is not supported by this type of card.
-9	ErrorDiffRangeNotSupport	All the analog input ranges must be the same for multi-channel analog input.
-10	ErrorLastChannelNotZero	The channels for multi-channel analog input must be ended with or started from zero.
-11	ErrorChannelNotDescending	The channels for multi-channel analog input must be contiguous and in descending order.
-12	ErrorChannelNotAscending	The channels for multi-channel analog input must be contiguous and in ascending order.
-13	ErrorOpenDriverFailed	Failed to open the device driver.
-14	ErrorOpenEventFailed	Open event failed in device driver.
-15	ErrorTransferCountTooLarge	The size of transfer is larger than the size of Initially allocated memory in driver.
-16	ErrorNotDoubleBufferMode	Double buffer mode is disabled.
-17	ErrorInvalidSampleRate	The specified sampling rate is out of range.
-18	ErrorInvalidCounterMode	The value of the *Mode* argument is invalid.
-19	ErrorInvalidCounter	The value of the *Ctr* argument is out of range.
-20	ErrorInvalidCounterState	The value of the *State* argument is out of range.
-21	ErrorInvalidBinBcdParam	The value of the *BinBcd* argument is invalid.
-22	ErrorBadCardType	The value of Card Type argument is invalid
-23	ErrorInvalidDaRefVoltage	The value of DA reference voltage argument is invalid

-24	ErrorAdTimeOut	Time out for AD operation
-25	ErrorNoAsyncAI	Continuous Analog Input is not set as Asynchronous mode
-26	ErrorNoAsyncAO	Continuous Analog Output is not set as Asynchronous mode
-27	ErrorNoAsyncDI	Continuous Digital Input is not set as Asynchronous mode
-28	ErrorNoAsyncDO	Continuous Digital Output is not set as Asynchronous mode
-29	ErrorNotInputPort	The value of AI/DI port argument is invalid
-30	ErrorNotOutputPort	The value of AO/DO argument is invalid
-31	ErrorInvalidDioPort	The value of DI/O port argument is invalid
-32	ErrorInvalidDioLine	The value of DI/O line argument is invalid
-33	ErrorContIoActive	Continuous IO operation is not active
-34	ErrorDblBufModeNotAllowed	Double Buffer mode is not allowed
-35	ErrorConfigFailed	The specified function configuration is failed
-36	ErrorInvalidPortDirection	The value of DIO port direction argument is invalid
-37	ErrorBeginThreadError	Failed to create thread
-38	ErrorInvalidPortWidth	The port width setting is not allowed
-39	ErrorInvalidCtrSource	The clock source setting is invalid
-40	ErrorOpenFile	Failed to Open file
-41	ErrorAllocateMemory	The memory allocation is failed
-42	ErrorDaVoltageOutOfRange	The value of DA voltage argument is out of range
-43	ErrorInvalidSyncMode	The sync. mode of operation is invalid
-44	ErrorInvalidBufferID	The buffer id selected is invalid
-45	ErrorInvalidCNTInterval	The counter value is invalid
-46	ErrorReTrigModeNotAllowed	The Re-Trigger mode of operation is invalid
-47	ErrorResetBufferNotAllowed	The buffer is not allowed to be reset
-48	ErrorAnaTriggerLevel	The value of analog trigger level is invalid
-201	ErrorConfigIoctl	The configuration API is failed
-202	ErrorAsyncSetIoctl	The async. mode API is failed
-203	ErrorDBSetIoctl	The double-buffer setting API is failed
-204	ErrorDBHalfReadyIoctl	The half-ready API is failed
-205	ErrorContOPIoctl	The continuous data acquisition API is failed
-206	ErrorContStatusIoctl	The continuous data acquisition status API setting is failed
-207	ErrorPIOIoctl	The polling data API is failed
-208	ErrorDIntSetIoctl	The dual interrupt setting API is failed
-209	ErrorWaitEvtIoctl	The wait event API is failed
-210	ErrorOpenEvtIoctl	The open event API is failed
-211	ErrorCOSIntSetIoctl	The COS interrupt setting API is failed
-212	ErrorMemMapIoctl	The memory mapping API is failed
-213	ErrorMemUMapSetIoctl	The memory Un-mapping API is failed
-214	ErrorCTRIoctl	The counter API is failed
-215	ErrorGetResIoctl	The resource getting API is failed

Appendix B AI Range Codes

The ***Analog Input Range*** of DAQ-2000 Cards

AD_B_10_V	Bipolar -10V to +10V
AD_B_5_V	Bipolar -5V to +5V
AD_B_2_5_V	Bipolar -2.5V to +2.5V
AD_B_1_25_V	Bipolar -1.25V to +1.25V
AD_B_0_625_V	Bipolar -0.625V to +0.625V
AD_B_0_3125_V	Bipolar -0.3125V to +0.3125V
AD_B_0_5_V	Bipolar -0.5V to +0.5V
AD_B_0_05_V	Bipolar -0.05V to +0.05V
AD_B_0_005_V	Bipolar -0.005V to +0.005V
AD_B_1_V	Bipolar -1V to +1V
AD_B_0_1_V	Bipolar -0.1V to +0.1V
AD_B_0_01_V	Bipolar -0.01V to +0.01V
AD_B_0_001_V	Bipolar -0.01V to +0.001V
AD_U_20_V	Unipolar 0 to +20V
AD_U_10_V	Unipolar 0 to +10V
AD_U_5_V	Unipolar 0 to +5V
AD_U_2_5_V	Unipolar 0 to +2.5V
AD_U_1_25_V	Unipolar 0 to +1.25V
AD_U_1_V	Unipolar 0 to +1V
AD_U_0_1_V	Unipolar 0 to +0.1V
AD_U_0_01_V	Unipolar 0 to +0.01V
AD_U_0_001_V	Unipolar 0 to +0.001V
AD_B_2_V	Bipolar -2V to +2V
AD_B_0_25_V	Bipolar -0.25V to +0.25V
AD_B_0_2_V	Bipolar -0.2V to +0.2V
AD_U_4_V	Unipolar 0 to +4V
AD_U_2_V	Unipolar 0 to +2V
AD_U_0_5_V	Unipolar 0 to +0.5V
AD_U_0_4_V	Unipolar 0 to +0.4V

Valid values for each card:

DAQ-2010
DAQ-2205
DAQ-2206
DAQ-2005
DAQ-2006
: AD_B_10_V, AD_B_5_V, AD_B_2_5_V, AD_B_1_25_V AD_U_10_V, AD_U_5_V, AD_U_2_5_V, AD_U_1_25_V

DAQ-2204
: AD_B_10_V, AD_B_5_V, AD_B_2_5_V, AD_B_2_V, AD_B_1_25_V, AD_B_1_V, AD_B_0_5_V, AD_B_0_25_V, AD_B_0_2_V, AD_B_0_05_V, AD_U_10_V, AD_U_5_V, AD_U_4_V, AD_U_2_5_V, AD_U_2_V, AD_U_1_V, AD_U_0_5_V, AD_U_0_4_V, AD_U_0_1_V

DAQ-2501
DAQ-2502
: AD_B_10_V, AD_U_10_V

Appendix C AI DATA FORMAT

This appendix lists the AI data format for the cards performing analog input operation, as well as the calculation methods to retrieve the A/ D converted data and the channel where the data read from.

Card Type	**Data Format**	**Value calculation** ** channel no. (CH#)* ** A/D converted data (ND)* ** Value returned from AI function (OD)*
DAQ-2010	*Every 16-bit signed integer data:* *D13 D12 ..... D1D0 b1 b0* *where D13,D12, ... , D0 : A/D converted data* *b1, b0 : Simultaneous Digital Input data.*	ND = OD >>2 or ND = OD/4
DAQ-2005 DAQ-2006	*Every 16-bit unsigned integer data:* *D15 D14 D13..... D1D0* *where D15,D14, ... , D0 : A/D converted data*	ND = OD
DAQ-2204	*Every 16-bit signed integer data:* *D12 D11 ..... D1D0 b3 b2 b1 b0* *where D12,D11, ... , D0 : A/D converted data* *b1, b0 : Simultaneous Digital Input data.*	ND = OD >>4 or ND = OD/16
DAQ-2205 DAQ-2206	*Every 16-bit signed integer data:* *D15 D14 D13..... D1D0* *where D15,D14, ... , D0 : A/D converted data*	ND = OD
DAQ-2501 DAQ-2502	*Every 16-bit signed integer data:* *D13 D12 ..... D1D0 b1 b0* *where D13,D12, ... , D0 : A/D converted data* *b1, b0 : AI Auto-scan Channel.*	ND = OD >>2 or ND = OD/4

Appendix D DATA File FORMAT

This appendix describes the file format of the data files generated by the functions performing continuous data acquisition followed by storing the data to disk.

The data file includes three parts, Header, ChannelRange (optional) and Data block. The file structure is as the figure below:

Header
ChannelRange (Optional)
DAQ data

Header

The *header* part records the information related to the stored data and its total length is 60 bytes. The data structure of the file header is as follows:

Total Length: 60 bytes			**Header**
Elements	**Type**	**Size (bytes)**	**Comments**
ID	char	10	file ID *ex. ADLINKDAQ2*
card_type	short	2	card Type *ex. DAQ2010*
num_of_channel	short	2	number of scanned channels *ex. 1, 2*
channel_no	unsigned char	1	Channel number where the data read from (only available as the num_of_channel is 1) *ex. 0, 1*
num_of_scan	long	4	the number of scan for each channel (total count / num_of_channel)
data_width	short	2	the data width 0: 8 bits, 1: 16 bits, 2: 32 bits
channel_order	short	2	the channel scanned sequence 0: normal (ex. 0 -1-2-3) 1: reverse (ex. 3-2-1-0) 2: custom* (ex. 0, 1, 3)
ad_range	short	2	the AI range code Please refer to Appexdix B *ex. 0 (AD_B_10_V)*

scan_rate	double	8	The scanning rate of each channel (total sampling rate / num_of_channel)
num_of_channel_range	short	2	The number of ChannelRange* structure
start_date	char	8	The starting date of data acquisition *ex. 12/31/99*
start_time	char	8	The starting time of data acquisition *ex. 18:30:25*
start_millisec	char	3	The starting millisecond of data acquisition *ex. 360*
Reserved	char	6	not used

* If the *num_of_channel_range* is 0, the *ChannelRange* block won't be included in the data file.
* The *channel_order* is set to "custom" only when the card supports variant channel scanning order.

ChannelRange

The *ChannelRange* part records the channel number and data range information related to t he stored data. This part consists of several channel & range units. The length of each unit is 2 bytes. The total length depends on the value of *num_of_channel_range* (one element of the file header) and is calculated as the following formula:

Total Length = *2* * num_of_channel_range bytes

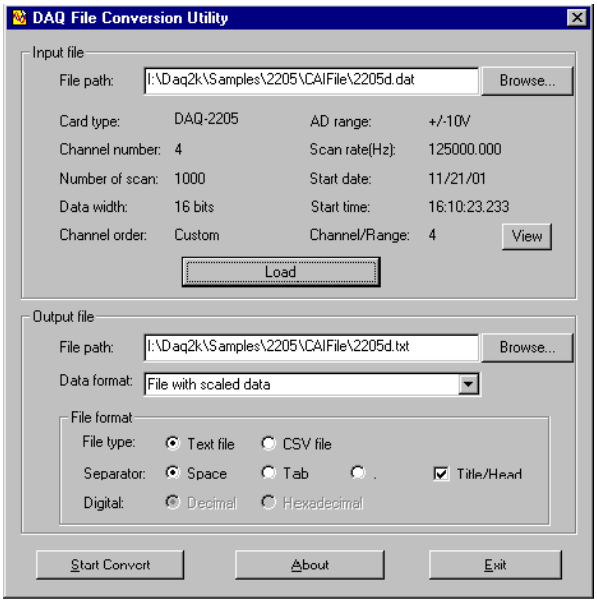
The data structure of each ChannelRange unit is as follows:

ChannelRange Unit *Length: 2 bytes*			
Elements	**Type**	**Size (bytes)**	**Comments**
Channel	char	1	scanned channel number *ex. 0, 1*
Range	char	1	the AI range code of *channel* Please refer to Appexdix B *ex. 0 (AD_B_10_V)*

Data Block

The last part is the data block. The data is written to file in 16 -bit binary format, with the lower byt e first (little endian). For example, the value 0x1234 is written to disk with 34 first followed by 12. The total length of the data block depends on the data width and the total data count.

The file is written in Binary format and can't be read in norma l text editor. You can use any binary file editor to view it or the functions used for reading files, e.g. fread, to get the file information and data value. D2K -DASK provides a useful utility *DAQCvt* for you to convert the binary file. The *DAQCvt* main window is as the figure below:



DAQCvt first translates the information stored in the header part and the ChannelRange part and then displays the corresponding information in the “Input File” frame of DAQCvt main window. After setting the properties (File Path, Format, …etc) of the converted file and push “*Start Convert*” button in the “Output File” frame, DAQCvt gets rid of header and ChannelRange parts and converts the data in data block according to the card type and the data width. Finally, DAQCvt writes the converted data to disk. You thus can use any text editor or Excel to view or analyze the accessed data.