

## ActiveX Control (OCX) for PCI-Bus multifunction boards

---

### User's Manual

[Version 1.0]

PCI1800X OCX is for PCI1800L/ PCI1800H

PCI1800X OCX is for PCI1802L/ PCI1802H

PCI1602X OCX is for PCI1602 / PCI1602F

PCI1202X OCX is for PCI1202L/PCI1202H

PCI1002X OCX is for PCI1002L/PCI1002H

## Table of Contents

1. Introduction.....	5
2. Data Type.....	6
3. Interface of multifunction board .....	7
3.1 General Properties .....	7
3.1.1 ErrorCode.....	8
3.1.2 ErrorString.....	8
3.1.3 Table of ErrorCode and ErrorString.....	9
3.2 General Methods.....	10
3.2.1 DriverInit ().....	10
3.2.2 GetDLLVersion () .....	11
3.2.3 GetDriverVersion () .....	11
3.2.4 DelayUs() .....	12
3.2.5 DriverClose () .....	13
3.3 General Properties .....	14
3.3.1 ActiveBoard .....	14
3.4 Digital Input/Output Methods .....	15
3.4.1 DigitalIn () .....	15
3.4.2 DigitalOut () .....	15
3.5 Analog Output Methods.....	16
3.5.1 AnalogOut ().....	16
3.6 Analog input methods.....	17
3.6.1 SetChannelConfig().....	17
3.6.2 ADPolling() .....	18
3.6.3 ADMultiPacer() .....	19
3.6.4 ADMultiPolling ().....	20
3.6.5 ADPollingHex().....	21

---

3.6.6	CONFIGURATION CODE .....	22
3.7	MagicScan Methods .....	24
3.7.1	AddToMagicScan ().....	25
3.7.2	ClearMagicScan ().....	26
3.7.3	ReadMagicScanStatus ().....	27
3.7.4	SaveMagicScan () .....	28
3.7.5	StartMagicScan ().....	29
3.7.6	StartMagicScanPostTrg() .....	30
3.7.7	StartMagicScanPreTrg().....	31
3.7.8	StartMagicScanMiddleTrg().....	32
3.7.9	StopMagicScan().....	33
3.8	M_Function Methods .....	34
3.8.1	SetMFun1Parameter ().....	35
3.8.2	StartMFun1 () .....	36
3.8.3	SetMFun2Parameter ().....	37
3.8.4	SetMFun2DaVal().....	38
3.8.5	StartMFun2 () .....	39
3.8.6	GetMFun2Buf ().....	40
3.8.7	SetMFun3Parameter ().....	41
3.8.8	SetMFun3ChannelConfig().....	42
3.8.9	StartMFun3 () .....	43
3.9	Special TwoMA / OneMA Methods.....	44
3.9.1	SetOneMAChannelConfig().....	45
3.9.2	SetOneMAParameter ().....	46
3.9.3	OneMAStart() .....	47
3.9.4	OneMAReadStatus() .....	48
3.9.5	OneMAStop() .....	49
3.9.6	SetTwoMAChannelConfig ().....	50
3.9.7	SetTwoMAParameter ().....	51
3.9.8	TwoMAStart() .....	52
3.9.9	TwoMAReadStatus() .....	53

- 3.9.10 TwoMAStop() .....54
- 3.10 The Continuous Capture Methods.....55
  - 3.10.1 SetFirstCardChannelConfig () .....56
  - 3.10.2 FirstCardStartScan () .....57
  - 3.10.3 GetFirstCardChannelData ().....58
  - 3.10.4 FirstCardStop ().....59
  - 3.10.5 SetSecondCardChannelConfig () .....60
  - 3.10.6 SecondCardStartScan () .....61
  - 3.10.7 GetSecondCardChannelData () .....62
  - 3.10.8 SecondCardStop ().....63
- 3.11 Interrupt Methods .....64
  - 3.11.1 GetIrqNo() .....65
  - 3.11.2 GetIrqCount.....65
  - 3.11.3 ADIrqStop() .....66
  - 3.11.4 ADIrqStart() .....67
  - 3.11.5 InstallIrq() .....68
  - 3.11.6 GetBuffer().....69
  - 3.11.7 GetFloatBuffer().....70
- 3.12 General Events.....71
  - 3.12.1 OnError () .....71

## 1. Introduction

The PCI1800X(PCI1602X,PCI1202X, PCI1002X) is an ActiveX Control component (OCX) for PCI-Bus multifunction Boards. It enables you to develop programs in a quick and easy way. Before using this driver, users need to install the PCI1800X(PCI1602X, PCI1202X, PCI1002X) OCX drivers into the system firstly and then insert the OCX component into the selected software development tools. Finally, you can make use of this OCX just like the general ActiveX Control components (OCX) included in your development tools.

Please also refer to "[ActiveX Control \(OCX\) Installation Manual](#)". It contains the following topics:

1. Installing the software into your system.
2. Installing/Uninstalling the ActiveX Control into/from Visual Basic 5.0
3. Installing/Uninstalling the ActiveX Control into/from Delphi 5.0
4. Installing/Uninstalling the ActiveX Control into/from Borland C++ Builder 3.0

## 2. Data Type

ActiveX Control Data Type	Data Size	BCB	Delphi	VB
short	2 Bytes	Short	SmallInt	Integer
long	4 Bytes	Long	LongInt	Long
float	4 Bytes	Float	Single	Single
LPCTSTR		Wchar_t *	String	String
BSTR		AnsiString	String	String

### 3. Interface of multifunction board

The interface of multifunction boards is for PCI1800X, PCI1602X, PCI1202X and PCI1002X.

#### 3.1 General Properties

Property Name	Data Type	Data Size	Access mode	Run-time Only	Description
ErrorCode	long	4 Bytes	Read/Write	No	Get/set the Error Code
ErrorString	BSTR		Read only	No	Get the Error Message

Note: In the following examples, please replace the "Control" word by your Control-Object name. For example:

- (1)PCI1800X1 or PCI1800X2... is for PCI1800X OCX,
- (2)PCI1602X1 or PCI1602X2 ...is for PCI1602X OCX,
- (3)PCI1202X1 or PCI1202X2 ...is for PCI1202X OCX.
- (4) PCI1002X1 or PCI1002X2 ...is for PCI1002X OCX.

### 3.1.1 ErrorCode

This property records any error code (includes 0: no-error) produced by software function driver after you use any method or other properties. Users should check on this property to make sure no error occurred after using any function.

**Return:** (long)

**Example:**

```
If Control.ErrorCode <>0 then
' Error occurs
' Do something
' Exit sub
End if
```

**Comment :** Please refer to the ErrorString property to understand the meaning of error message.

### 3.1.2 ErrorString

This property provides message information according to error code when an error has been occurred. That is, users can get the meaning of error message from the ErrorString property.

**Return:** (BSTR)

**Example:**

```
strError = Control.ErrorString 'Get the error message.'
```

### 3.1.3 Table of ErrorCode and ErrorString

Error Code	Error ID	Error String	Comment
0	NoError	OK ! No Error!	
1	DriverHandleError	Device driver opened error	
2	DriverCallError	Got the error while calling the driver functions	
3	ADControllerError	Embedded controller handshake error	
4	ConfigCodeError	Refer to "Section 3.6.9 Configuration Code"	
5	ADPollingTimeOut	Hardware timeout error	
6	FindBoardError	Can't find the multifunction board on the system	
7	ADChannelError	The valid range is 0 to 31	
8	DAChannelError	The valid channel number must be 0 or 1	
9	InvalidateDelay	The valid range of dwDelayUs is <= 8191	
10	DelayTimeOut	Timeout	
11	InvalidateData	Invalid Data	
12	TimeOut	Timeout	
13	ExceedBoardNumber	Invalidate board number (Valid range: 0 to TotalBoards -1)	
14	NotFoundBoard	Can't detect any multifunction board on the system	
15	OpenError	(Not Used)	
16	FindTwoBoardError	Can't find out two multifunction boards	
17	AllocateMemoryError	Fail to allocate the memory buffer	
101	M_FunExecError	Fail to execute the M_Functions	not for 1002
102	FrequencyComputeError	(Not Used)	not for 1002
103	HighAlarm	Analog input value > High Alarm value	not for 1002
104	LowAlarm	Analog input value < Low Alarm value	not for 1002
105	AlarmTypeError	The valid range is: 0 to 4	not for 1002
106	FifoOverflow	FIFO overflow	not for 1002
107	ThreadCreateError	Fail to create thread	not for 1002
108	StopError	Stop Error	not for 1002
109	DriverNoOpen	Driver not open	for 1002 only
110	GetIntCountError	Get counter of interrput error	for 1002 only
111	InstallIrqError	Install IRQ Error	for 1002 only
112	RemoveIrqError	Remove IRQ Error	for 1002 only
113	ClearIntCountError	Clear counter value Error	for 1002 only

## 3.2 General Methods

Method Name	Data type of returned value	Data size	Descriptions
<b>DriverInit</b>	short	2 Bytes	Open the driver and allocate the resource for the device, and get the total boards.
<b>GetDLLVersion</b>	short	2 Bytes	Get the DLL file version.
<b>GetDriverVersion</b>	short	2 Bytes	Get the driver version in the system
<b>DelayUs</b>	void		Set the delay time
<b>DriverClose</b>	void		Close the Driver and release the resource from the device.

### 3.2.1 DriverInit ()

Start the PCI Multifunction board's driver and allocate the computer resource for the device. This method must be called once before calling or using control methods or other properties

**Prototype:** short DriverInit()

**Return:** Total boards of defined PCI Multifunction board.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
boardsControl.ActiveBoard=0 'Set the Active Board to 0
Print Control.GetDllVersion 'Print the DLL Version
```

### 3.2.2 GetDLLVersion ()

Get the version information of multifunction board's drivers.

**Prototype:** short GetDLLVersion()

**Return:** DLL version of multifunction board.

**Example:**

*For example: If it returns 0x250, then the version is 2.50.*

```
wVer = Control.GetDLLVersion
```

### 3.2.3 GetDriverVersion ()

This method is used to obtain the driver version information of multifunction board from .Vxd driver of window 9X or .Sys driver of windows NT/2000.

**Prototype:** short GetDriverVersion()

**Return:** Driver version from operation system.

**Example:**

*For example: If it returns 0x200, then the version is 2.00.*

```
wVer = Control.GetDriverVersion
```

### 3.2.4 DelayUs()

This function provides a machine independent timer. It can be used to delay the setting time or used as a general purpose machine independent timer. It need to work with the current active multifunction board. Please use the method **ActiveBoard** to select the active board.

**Prototype:** void DelayUs(short nDelaytime)

**Parameters:** nDelaytime: Number of us to delay, 8191 Max

nDelaytime =1 → delay 1 us

nDelaytime =1000 → delay 1000 us = 1 ms

nDelaytime =8191 → delay 8191 us = 8.191 ms (maximum delay)

nDelaytime =8192 → invalidate delay (will return error)

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard=0 'Set the first board to active.
Control.SetChannelConfig 1, 0 ' +/- 5V range
Control.DelayUs 23 ' delay 23 us settling time
V0 = Control.ADPolling 'V0 is ADPolling value
```

**Comments:**This function is not for PCI-1002.

### 3.2.5 DriverClose ()

Stop and close the OCX Driver of PCI multifunction board and release the device resource from computer device resource. This method must be called once before exiting the user's application program.

**Prototype:** void DriverClose ()

**Example:**

```
TotalBoards = Control.DriverInit 'initial driver  
Control.ActiveBoard = 0 'select action board=0  
...  
...  
Control.DriverClose 'release the device resource
```

### 3.3 General Properties

Properties Name	Data type of returned value	Data size	Descriptions
<b>ActiveBoard</b>	short	2 Bytes	Get or select active multifunction boards installed in the system.

#### 3.3.1 ActiveBoard

This property is used to activate one of the multifunction boards installed in the system. This function must be called once before the DigitalIn, DigitalOut, AnalogIn, AnalogOut functions are used. It allow user to get or select the **ActiveBoard**.

**Return:** (short) Active board number.

**Default:** 0 (ActiveBoard=0)

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard=0 'Set the Active Board to 0
Board=Control.ActiveBoard 'Get the Active Board value
```

### 3.4 Digital Input/Output Methods

Method Name	Data type of returned value	Data size	Descriptions
<b>DigitalIn</b>	short	2 Bytes	Get digital input value.
<b>DigitalOut</b>	void		Output digital value.

#### 3.4.1 DigitalIn ()

This method is used to obtain the digital input value from active multifunction board.

**Prototype:** short DigitalIn ()

**Return:** 16 bits data from Digital input port

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard=0 'Set the Active Board to 0
Print Control.DigitalIn 'Print the digital input value
```

#### 3.4.2 DigitalOut ()

This method is used to output the digital value throught active multifunction board.

**Prototype:** void DigitalOut (long wDo)

**Parameters:** wDo: Digital output value by Hex format.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard=0 'Set the Active Board to 0
Control.DigitalOut 5 'Digital output value
```

### 3.5 Analog Output Methods

Method Name	Data type of returned value	Data size	Descriptions
AnalogOut	void		This method is used to output analog value.

#### 3.5.1 AnalogOut ()

This method is used to output 12 bits DA data to DA port through current active PCI multifunction board. Firstly, users need to call **Control.ActiveBoard** method to activate selected PCI multifunction board and then use this function to output analog value. The analog output value is in Hex format.

**Prototype:** void AnalogOut (short nChannel, short nDaVal)

**Parameters:** nChannel : The Channel Number of card: 0 to 1  
 nDaVal : Analog output value by hex format

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard=0 'Set the Active Board to 0
Control.AnalogOut 0, A000 'Channel Number:0 Analog output value:A000(hex)
```

**Comment :**This function is not for PCI-1002.

### 3.6 Analog input methods

Method Name	Data type of returned value	Data size	Descriptions
<b>SetChannelConfig</b>	Void		Set the AD channel's configuration code.
<b>ADPolling</b>	Float	4 Bytes	Get Analog input value by float format.
<b>ADMultiPacer</b>	Void		Perform multiple AD conversions by pacer trigger.
<b>ADMultiPolling</b>	Void		Perform multiple AD conversions by polling.
<b>ADPollingHex</b>	short	2 Bytes	Get Analog input value by Hex format.

#### 3.6.1 SetChannelConfig()

This method provides a function to setup AD channel configuration code. This subroutine can be used to setup the active AD channel of the **ADPolling**, **ADMultiPolling** and **ADMultiPacer** function for the active multifunction board. In order to enable this function, users need to call the **ActiveBoard** method to activate selected PCI multifunction board.

**Prototype:** void SetChannelConfig (short nChannel, short nConfig)

**Parameters:** nChannel: Set channel number

nConfig : Configuration code.

Refer to “Section 3.6.9 Configuration Table” for details.

**Example:**

```

wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard=0           'Set the first Board to active
Control.SetChannelConfig 0, 0   ' Channel :0 +/- 5V range
Control.DelayUs 23              ' delay 23 us settling time
V0=Control.ADPolling            'V0:Channel 0 analog value
    
```

### 3.6.2 ADPolling()

This method performs the AD conversion by polling method. Function **Control.SetChannelConfig** can be used to set up or change analog input channel and configuration code. And then **Control.ADPolling** bases on this configuration to obtain analog input value from defined channel by the float format according to active multifunction board. In first, users need to call **Control.ActiveBoard** method to activate selected PCI multifunction board.

**Prototype:** float ADPolling ()

**Return:** Analog input value in float format.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard=0 "Set the first Board to active
Control.SetChannelConfig 0, 0 ' Channel :0 +/- 5V range
Control.DelayUs 23 ' delay 23 us settling time
V0=Control. ADPolling 'V0:Channel 0 analog value
```

### 3.6.3 ADMultiPacer()

This method performs multiple AD conversions by pacer trigger. Function **SetChannelConfig** can be used to change channel or configuration code. Then, according to this setting, **ADMultiPacer** can get Analog input data by hardware pacer trigger periodically. These AD data can be used to reconstruct the waveform from analog input channel. However, software polling signal triggers the function **ADMultiPolling** and AD conversion operation is interrupted by system OS. It is recommended to use **ADMultiPacer** if the input waveform reconstruction is needed.

**Prototype:** void ADMultiPacer (float FAR\* fAdsBuf, long wNum, short wSample)

**Parameters:** fAdsBuf : The starting address of AD data buffer (16 bits), these data will be automatically computed based on the setting of the SetChannelConfig function.

wNum : Number of AD conversions will be performed.

wSample : **AD sampling rate=8M/wSample**

wSample=80      Sampling rate=8M/80=100K

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard=0 "Set the first Board to active
Control.SetChannelConfig 3, 0 'Setting channel:3 Configuration code:0 .
Control.DelayUs 23 ' delay 23 us settling time
Control.ADMultiPacer fAdBuf(0), 100, 80
' Start the ADMultiPacer function and save data into fAdBuf
```

### 3.6.4 ADMultiPolling ()

This method performs multiple AD conversions by software polling trigger during one batch time. Function **SetChannelConfig** can be used to change channel or configuration code. And the **ADMultiPolling** function bases on the setting of **SetChannelConfig** to get AD data. This function also refer to the current active multifunction board by using the function **Control.ActiveBoard**.

**Prototype:** void ADMultiPolling (float FAR\* fAdBuf, long wNum)

**Parameters:** fAdsBuf :Starting address of AD data buffer (16 bits), these data will be automatically computed based on the setting of the SetChannelConfig function.

wNum :Number of AD conversions will be performed.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard=0 "Set the first Board to active
Control.SetChannelConfig 3, 0 ' +/- 2.5V range
Control.DelayUs 23 ' delay 23 us settling time
Control. ADMultiPolling fAdBuf(0), 100
'Start the ADMultiPolling function and save data into fAdBuf
```

### 3.6.5 ADPollingHex()

This method performs the AD conversion by polling method. Function **Control.SetChannelConfig** can be used to set up or change analog input channel and configuration code. And then **Control.ADPollingHex** bases on those configuration to obtain analog input value from defined channel according to active multifunction board. In first, users need to call **Control.ActiveBoard** method to activate selected PCI multifunction board.

**Prototype:** short ADPollingHex ()

**Return:** Analog input value which is automatically computed based on the setting of `Control.SetChannelConfig()`.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard=0 "Set the first Board to active
Control.SetChannelConfig 0, 0 ' Channel :0 +/- 5V range
V0=Control. ADPollingHex 'V0:Channel 0 analog value
```

### 3.6.6 CONFIGURATION CODE

**PCI1202L/PCI-1800L/PCI1802L Configuration Code Table**

Bipolar/Unipolar	Input Signal Range	Gain	Settling Time	Configuration Code
Bipolar	+/- 5V	1	3 us	0x00
Bipolar	+/- 2.5V	2	3 us	0x01
Bipolar	+/- 1.25V	4	3 us	0x02
Bipolar	+/- 0.625V	8	3 us	0x03
Bipolar	+/- 10V	0.5	3 us	0x04
Bipolar	+/- 5V	1	3 us	0x05
Bipolar	+/- 2.5V	2	3 us	0x06
Bipolar	+/- 1.25V	4	3 us	0x07
Unipolar	0V ~ 10V	1	3 us	0x08
Unipolar	0V ~ 5V	2	3 us	0x09
Unipolar	0V ~ 2.5V	4	3 us	0x0A
Unipolar	0V ~ 1.25V	8	3 us	0x0B

**PCI-1602 Configuration Code Table**

Bipolar/Unipolar	Input Signal Range	Gain	Settling Time	Configuration Code
Bipolar	+/- 10V	1	10 us	0
Bipolar	+/- 5V	2	10 us	1
Bipolar	+/- 2.5V	4	10us	2
Bipolar	+/- 1.25V	8	10 us	3

**PCI-1602F Configuration Code Table**

Bipolar/Unipolar	Input Signal Range	Gain	Settling Time	Configuration Code
Bipolar	+/- 10V	1	5 us	0
Bipolar	+/- 5V	2	5 us	1
Bipolar	+/- 2.5V	4	5 us	2
Bipolar	+/- 1.25V	8	5 us	3

**PCI1202H/PCI-1800H/PCI1802H Configuration Code Table**

Bipolar/Unipolar	Input Signal Range	Gain	Settling Time	Configuration Code
Bipolar	+/- 5V	1	23 us	0x10
Bipolar	+/- 0.5V	10	28 us	0x11
Bipolar	+/- 0.05V	100	140 us	0x12
Bipolar	+/- 0.005V	1000	1300 us	0x13
Bipolar	+/- 10V	0.5	23 us	0x14
Bipolar	+/- 1V	5	28 us	0x15
Bipolar	+/- 0.1V	50	140 us	0x16
Bipolar	+/- 0.01V	500	1300 us	0x17
Unipolar	0V ~ 10V	1	23 us	0x18
Unipolar	0V ~ 1V	10	28 us	0x19
Unipolar	0V ~ 0.1V	100	140 us	0x1A
Unipolar	0V ~ 0.01V	1000	1300 us	0x1B

**PCI-1002L Configuration Code Table**

Bipolar/Unipolar	Input Signal Range	Gain	Max. Switching Frequency	Configuration Code
Bipolar	+/- 5V	1	110K/s	0x00
Bipolar	+/- 2.5V	2	110K/s	0x01
Bipolar	+/- 1.25V	4	110K/s	0x02
Bipolar	+/- 0.625	8	110K/s	0x03

**PCI-1002H Configuration Code Table**

Bipolar/Unipolar	Input Signal Range	Gain	Max. Switching Frequency	Configuration Code
Bipolar	+/- 5V	1	44K/s	0x10
Bipolar	+/- 0.5V	10	36K/s	0x21
Bipolar	+/- 0.05V	100	7K/s	0x32
Bipolar	+/- 0.005	1000	0.8K/s	0x43

### 3.7 MagicScan Methods

These methods cannot be used with PCI-1002 series multifunction boards.

Method Name	Data type of returned value	Data size	Descriptions
<b>AddToMagicScan</b>	Void		Add one channel to the MagicScan circular queue.
<b>ClearMagicScan</b>	Void		Initialize the MagicScan controller to the Initial state.
<b>ReadMagicScanStatus</b>	Short	2 Bytes	Get MagicScan Status value
<b>SaveMagicScan</b>	Void		Specify the starting address of AD databuffer for MagicScan.
<b>StartMagicScan</b>	Void		Start the MagicScan operation
<b>StartMagicScanPostTrg</b>	Void		Start the MagicScanPostTrg operation
<b>StartMagicScanPreTrg</b>	Void		Start the MagicScanPretTrg operation
<b>StartMagicScanMiddleTrg</b>	Void		Start the MagicScanMiddleTrg operation
<b>StopMagicScan</b>	Void		Stop MagicScan

### 3.7.1 AddToMagicScan ()

This method adds one channel to the **MagicScan circular queue** for current active PCI multifunction board. Before enabling this function, users need to call function **Control.ActiveBoard** to activate selected PCI multifunction board.

**Prototype:** void AddToMagicScan (short nAdChannel, short nConfig, short nAverage, short nLowAlarm, short nHighAlarm, short nAlarmType)

**Parameters:**

- nAdChannel : Set channel number
- nConfig : Configuration code, refer to "Section 3.6.9 Configuration Table" for details.
- nAverage : The factor of digital average filter
- nLowAlarm : 12 bits low alarm data
- nHighAlarm : 12 bits high alarm data
- nAlarmType : 0=no alarm, 1=high alarm, 2=low alarm, 3=in-alarm, 4=out-alarm

**Example:**

```

wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board=0
Control. ClearMagicScan 'Clear the scan data
Control. AddToMagicScan 0, 0, 1, 0, 0, 0 ' Add CH:0 to scan in Magic Scan
Control. SaveMagicScan 0, wV0(0) 'Save data in place 0 of Magic scan buffer
Control. AddToMagicScan 2, 0, 1, 0, 0, 0 ' Add CH:2 to scan in Magic Scan
Control. SaveMagicScan 1, wV1(0) 'Save data in place 1 of Magic scan buffer
Control. StartMagicScan 24, 2000, 0
'Set parameters of magic scan and start scan
    
```

**Comments:** Please refer to the demo11 and it is not for PCI-1002.

### 3.7.2 ClearMagicScan ()

This method initializes the **MagicScan** controller of current active PCI multifunction board into the Initial state. Users need to call **Control.ActiveBoard** method to activate selected PCI multifunction board in advance.

**Prototype:** void ClearMagicScan ()

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board=0
Control. ClearMagicScan 'Clear the scan data
Control. AddToMagicScan 0, 0, 1, 0, 0, 0 ' Add CH:0 to scan in Magic Scan
Control. SaveMagicScan 0, wV0(0) 'Save data in place 0 of Magic scan buffer
Control. AddToMagicScan 2, 0, 1, 0, 0, 0 ' Add CH:2 to scan in Magic Scan
Control. SaveMagicScan 1, wV1(0) 'Save data in place 1 of Magic scan buffer
Control. StartMagicScan 24, 2000, 0
'Set parameters of magic scan and start scan
While True 'Do while
    Status = Control. ReadMagicScanStatus "Get the Magic Scan Status
    If Status > 1 Then ' If the status >1
        GoTo Over ' call the sub "Over"
    End If 'End if
    Sleep (10)
Wend
Over:
Control.StopMagicScan 'Stop magic scan
```

**Comments:** Please refer to the demo 11 and it is not for PCI-1002.

### 3.7.3 ReadMagicScanStatus ()

This method obtains the status of the **MagicScan** operation for current active multifunction board.

**Prototype:** short ReadMagicScanStatus ()

**Return:** MagicScan status

Status	Description
0x00	MagicScan initial condition (idle state)
0x01	MagicScan is working.
0x02	MagicScan stage 1 controller timeout
0x04	MagicScan stage 2 controller timeout
0x08	MagicScan FIFO overflow
0x80	MagicScan function OK

**Example:**

```

wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board=0
Control. ClearMagicScan 'Clear the scan data
Control. AddToMagicScan 0, 0, 1, 0, 0, 0 ' Add CH:0 to scan in Magic Scan
Control. SaveMagicScan 0, wV0(0) 'Save data in place 0 of Magic scan buffer
Control. AddToMagicScan 2, 0, 1, 0, 0, 0 ' Add CH:2 to scan in Magic Scan
Control. SaveMagicScan 1, wV1(0) 'Save data in place 1 of Magic scan buffer
Control. StartMagicScan 24, 2000, 0
'Set parameters of magic scan and start scan
While True 'Do while
    Status = Control. ReadMagicScanStatus ""Get the Magic Scan Status
    If Status > 1 Then ' If the status >1
        GoTo Over ' call the sub "Over"
    End If 'End if
    Sleep (10)
Wend
Over:
    Control.StopMagicScan 'Stop magic scan
    
```

**Comments:** Please also refer to the demo 11 and it is not for PCI-1002.

### 3.7.4 SaveMagicScan ()

This method specifies the scan sequence into the scan queue and the starting address of AD data buffer of current active multifunction board for **MagicScan** method.

**Prototype:** void SaveMagicScan (short wAdChannel, short FAR\* wAdBuf)

**Parameters:** wAdChannel: Scan sequence in the scan queue.

(Note: not the AD channel number.)

wAdBuf : Buffer to store the AD data of Magic Scan function

**Example:**

```

wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board=0
Control. ClearMagicScan 'Clear the scan data
Control. AddToMagicScan 0, 0, 1, 0, 0, 0 'Add CH:0 to scan in Magic Scan
Control. SaveMagicScan 0, wV0(0) 'Save data in place 0 of Magic scan buffer
Control. AddToMagicScan 2, 0, 1, 0, 0, 0 ' Add CH:2 to scan in Magic Scan
Control. SaveMagicScan 1, wV1(0) 'Save data in place 1 of Magic scan buffer
Control. StartMagicScan 24, 2000, 0
'Set parameters of magic scan and start scan
While True 'Do while
    Status = Control. ReadMagicScanStatus "Get the Magic Scan Status
    If Status > 1 Then ' If the status >1
        GoTo Over ' call the sub "Over"
    End If 'End if
    Sleep (10)
Wend
Over:
Control.StopMagicScan 'Stop magic scan
    
```

**Comments:** Please also refer to the demo 11 and it is not for PCI-1002.

### 3.7.5 StartMagicScan ()

This method starts the **MagicScan** operation of current active multifunction board. Users can utilize **Control. ReadMagicScanStatus** to check the operation state of **MagicScan** function.

**Prototype:** void StartMagicScan (short nSampleRateDiv, long dwNum, short nPriority)

**Parameters:** nSampleRateDiv: AD sampling rate = 8M/ nSampleRateDiv.

nSampleRateDiv =24 → sampling rate=8M/24=330K

dwNum : Number of **MagicScan cycle** performed

nPriority: Be used to adjust the priority of the thread. For example, to prevent the lock of CPU, users can try to use the lower priority.

Value	ID (Description)
-2	THREAD_PRIORITY_LOWEST
-1	THREAD_PRIORITY_BELOW_NORMAL
0	THREAD_PRIORITY_NORMAL
1	THREAD_PRIORITY_ABOVE_NORMAL
2	THREAD_PRIORITY_HIGHEST
15	THREAD_PRIORITY_TIME_CRITICAL
others	THREAD_PRIORITY_NORMAL

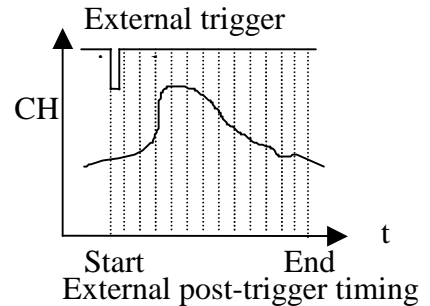
**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board=0
Control. ClearMagicScan 'Clear the scan data
Control. AddToMagicScan 0, 0, 1, 0, 0, 0 ' Add CH:0 to scan in Magic Scan
Control. SaveMagicScan 0, wV0(0) 'Save data in place 0 of Magic scan buffer
Control. AddToMagicScan 2, 0, 1, 0, 0, 0 ' Add CH:2 to scan in Magic Scan
Control. SaveMagicScan 1, wV1(0) 'Save data into Magic scan buffer
Control. StartMagicScan 24, 2000, 0
'Set parameters of Magic scan and start scan
```

**Comments:** Please also refer to the demo11 and it is not for PCI-1002.

### 3.7.6 StartMagicScanPostTrg()

This method starts the post-trigger operation of MagicScan for current active multifunction board. Users can utilize the method, **ReadMagicScanStatus**, to check the operation state of **StartMagicScanPostTrg** function.



**Prototype:** void StartMagicScanPostTrigger(short wSampleRateDiv, long dwNum, short nPriority);

**Parameters:** wSampleRateDiv: AD sampling rate = 8M/ wSampleRateDiv.

wSampleRate=24 → sampling rate=8M/24=330K

dwNum : Number of **MagicScan cycle** performed

nPriority: Be used to adjust the priority of the thread. For example, to prevent the lock of CPU, users can try to use the lower priority.

Value	ID (Description)
-2	THREAD_PRIORITY_LOWEST
-1	THREAD_PRIORITY_BELOW_NORMAL
0	THREAD_PRIORITY_NORMAL
1	THREAD_PRIORITY_ABOVE_NORMAL
2	THREAD_PRIORITY_HIGHEST
15	THREAD_PRIORITY_TIME_CRITICAL
others	THREAD_PRIORITY_NORMAL

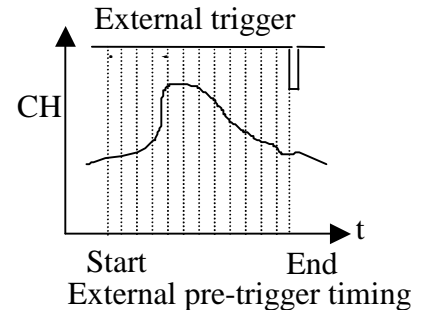
**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board=0
Control.ClearMagicScan 'Clear the scan data
Control.AddToMagicScan 0, 0, 1, 0, 0, 0 ' Add CH:0 to scan in Magic Scan
Control.SaveMagicScan 0, wV0(0) 'Save data in place 0 of Magic scan buffer
Control.AddToMagicScan 2, 0, 1, 0, 0, 0 ' Add CH:2 to scan in Magic Scan
Control.SaveMagicScan 1, wV1(0) 'Save data in place 1 of Magic scan buffer
Control.StartMagicScanPostTrigger 24, 100, 0
'Set parameters of Post-Trigger Magic scan and start scan
```

**Comments:** Please refer to the demo23 and it is not for PCI-1002.

### 3.7.7 StartMagicScanPreTrg()

This method will start the pre-trigger operation of MagicScan for current active multifunction board. Users can use the method **ReadMagicScanStatus** to check the operation state of **StartMagicScanPreTrg** function.



**Prototype:** void StartMagicScanPreTrigger(short wSampleRateDiv, long dwNum, short nPriority);

**Parameters:** wSampleRateDiv : AD sampling rate = 8M/ wSampleRateDiv.

wSampleRate=24 → sampling rate=8M/24=330K

dwNum : Number of **MagicScan cycle** performed

nPriority: Be used to adjust the priority of the thread. For example, to prevent the lock of CPU, users can try to use the lower priority.

Value	ID (Description)
-2	THREAD_PRIORITY_LOWEST
-1	THREAD_PRIORITY_BELOW_NORMAL
0	THREAD_PRIORITY_NORMAL
1	THREAD_PRIORITY_ABOVE_NORMAL
2	THREAD_PRIORITY_HIGHEST
15	THREAD_PRIORITY_TIME_CRITICAL
others	THREAD_PRIORITY_NORMAL

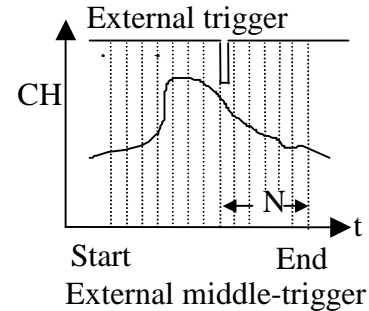
**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board=0
Control. ClearMagicScan 'Clear the scan data
Control. AddToMagicScan 0, 0, 1, 0, 0, 0 ' Add CH:0 to scan in Magic Scan
Control. SaveMagicScan 0, wV0(0) 'Save data in place 0 of Magic scan buffer
Control. AddToMagicScan 2, 0, 1, 0, 0, 0 ' Add CH:2 to scan in Magic Scan
Control. SaveMagicScan 1, wV1(0) 'Save data in place 1 of Magic scan buffer
CI1602X1.StartMagicScanPreTrigger 24, 100, 0
'Set parameters of Pre-Trigger Magic scan and start scan
```

**Comments:** Please also refer to the demo24 and it is not for PCI-1002.

### 3.7.8 StartMagicScanMiddleTrg()

This method will start the middle-trigger operation of MagicScan for current active multifunction board. Users can use the method **ReadMagicScanStatus** to check the operation state of **StartMagicScanMiddleTrg** function.



**Prototype:** void StartMagicScanMiddleTrigger(short wSampleRateDiv, long dwN1, long dwN2, short nPriority);

**Parameters:** wSampleRateDiv: AD sampling rate =  $8M / wSampleRateDiv$ .  
 wSampleRate=24 → sampling rate= $8M/24=330K$   
 dwN1 : Number of **MagicScan cycle** performed  
 dwN2 : Number of **MagicScan cycle** performed  
 nPriority : Be used to adjust the priority of the thread. For example, to prevent the lock of CPU, users can try to use the lower priority.

Value	ID (Description)
-2	THREAD_PRIORITY_LOWEST
-1	THREAD_PRIORITY_BELOW_NORMAL
0	THREAD_PRIORITY_NORMAL
1	THREAD_PRIORITY_ABOVE_NORMAL
2	THREAD_PRIORITY_HIGHEST
15	THREAD_PRIORITY_TIME_CRITICAL
others	THREAD_PRIORITY_NORMAL

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board=0
Control.ClearMagicScan 'Clear the scan data
Control.AddToMagicScan 0, 0, 1, 0, 0, 0 ' Add CH:0 to scan in Magic Scan
Control.SaveMagicScan 0, wV0(0) 'Save data in place 0 of Magic scan buffer
Control.AddToMagicScan 2, 0, 1, 0, 0, 0 ' Add CH:2 to scan in Magic Scan
Control.SaveMagicScan 1, wV1(0) 'Save data in place 1 of Magic scan buffer
Control.StartMagicScanMiddleTrigger 24, 100, 250, 0
'Set parameters of Middle-Trg Magic scan and start scan
```

**Comments:** Please also refer to the demo25 and it is not for PCI-1002.

### 3.7.9 StopMagicScan()

This method is used to stop the **MagicScan** mechanism.

**Prototype:** void StopMagicScan ()

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board=0
Control. ClearMagicScan 'Clear the scan data
Control. AddToMagicScan 0, 0, 1, 0, 0, 0 ' Add CH:0 to scan in Magic Scan
Control. SaveMagicScan 0, wV0(0) 'Save data in place 0 of Magic scan buffer
Control. AddToMagicScan 2, 0, 1, 0, 0, 0 ' Add CH:2 to scan in Magic Scan
Control. SaveMagicScan 1, wV1(0) 'Save data in place 1 of Magic scan buffer
Control. StartMagicScan 24, 2000, 0
'Set parameters of magic scan and start scan
While True 'Do while
    Status = Control. ReadMagicScanStatus "Get the Magic Scan Status
    If Status > 1 Then ' If the status >1
        GoTo Over ' call the sub "Over"
    End If 'End if
    Sleep (10)
Wend
Over:
Control.StopMagicScan 'Stop magic scan
```

**Comments:** Please also refer to the demo11 and it is not for PCI-1002.

### 3.8 M\_Function Methods

Some real world applications have to send out the pre-defined pattern signal to the external device and measure the output responses of the device for analysis purpose. Maybe users need one arbitrary waveform generator and one high speed AD converter. The **MFunctions**, provided by PCI-1202/1602/1800/1802, can send out the user defined arbitrary wave form and perform the AD conversion at the same time. The detail functions list are shown in the following table. Note that the following methods can not be used with PCI-1002.

Method Name	Data type of returned value	Data size	Descriptions
<b>SetMFun1Parameter</b>	Void		Set the Parameter of MFun1 Function
<b>StartMFun1</b>	Void		Start MFun1 Function and get data
<b>SetMFun2DaVal</b>	Void		Set DA value and store into DA buffer
<b>SetMFun2Parameter</b>	Void		Set the Parameter of MFun2
<b>StartMFun2</b>	Void		Start MFun2 Function
<b>GetMFun2Buf</b>	Void		Get MFun2 buffer data
<b>SetMFun3Parameter</b>	Void		Set Parameter of Mfun3.
<b>SetMFun3ChannelConfig</b>	Void		Set channel's Parameter of Mfun3.
<b>StartMFUN3</b>	Void		Start MFun3 Function and get data

The procedures for how to use these methods of M-function are shown as following.

1. The MFun1 working procedure

MFun1: SetMFun1Parameter -> StartMFun1

2. The Mfun2 working procedure

MFun2: SetMFun2DaVal -> SetMFun2Parameter-> StartMFun2-> GetMFun2Buf

3. The Mfun3 working procedure

MFun3: SetMFun3ChannelConfig -> SetMFun3Parameter -> StartMFun3

Function name	DA	AD
MFun1	channel_0, sine wave	channel_0, ±10V
MFun2	channel_0, arbitrary wave form	channel_0, ±10V
MFun3	channel_0, sine wave	channel/gain programmable (32 channels max.)

### 3.8.1 SetMFun1Parameter ()

This method is used to configure the Parameters of MFun1 function to generate the waveform image automatically.

**Prototype:** void SetMFun1Parameter(short nDaFrequency, short nDaWave, float fDaAmplitude, short nAdClock, short nAdNumber, short nAdConfig, long fLowAlarm, float fHighAlarm)

**Parameters:** nDaFrequency : DA output frequency = 1.8M/wDaFrequency (Pentium 120)  
nDaWave : Number of DA waveform to be generated  
fDaAmplitude : Amplitude of DA output.

**NOTE : the hardware J1 must be selected as +/-10V**

nAdClock : **AD sampling clock (samples/sec)=8M/ nAdClock**  
nAdNumber : Number of AD data to be read  
nAdConfig : Configuration code of AD input range, Refer to "Section 3.6.5"  
fLowAlarm : Low alarm limit. if **value < fLowAlarm** → LowAlarm  
fHighAlarm : High alarm limit. if **value >fHighAlarm** → HighAlarm

#### Example:

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board 0
Control.DigitalOut Dodata 'Set the digital output value
Didata = Control.DigitalIn 'Get the digital input value
Control.SetMFun1Parameter 360, 10, 4.5, 80, 200, 0, -10, 10
'Set Parameters of MFun1 Function
Control.StartMFun1 AdBuf(0) 'Start MFun1 and get MFun1 data
```

**Comments:** Please refer to the DEMO5 and it is not for PCI-1002.

### 3.8.2 StartMFun1 ()

The function **StartMFun1** is used to start **Mfun1** to execute the default duty. (Refer to “multifunction boards Hardware Manual” chapter-5 for details). Here, the Analog input channel\_0 and analog output channel\_0 are used to produce designated function.

**Prototype:** void StartMFun1(float FAR\* fAdBuf)

**Parameters:** fAdBuf : The starting address of **fAdBuf** which store the AD data

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board 0
Control.DigitalOut Dodata 'Set the digital output value
Didata = Control.DigitalIn 'Get the digital input value
Control.SetMFun1Parameter DF, DW, DA, AC, 200, 0, -10, 10
'Set Parameters of MFun1 Function
Control.StartMFun1 AdBuf(0) 'Start MFun1 and get MFun1 data
```

**Comments:** Please refer to the demo5 and it is not for PCI-1002.

### 3.8.3 SetMFun2Parameter ()

This method provides a function to set up configuration code of function **Mfun2**. This subroutine can be used to set how much data of DA samples can be outputted during one waveform, how many DA waveforms can be outputted in one batch time, AD sampling rate, how many AD data can be read during one working cycle and the configuration code of AD input range for continuous waveform generation and capture of MFun2Function.

**Prototype:** void SetMFun2Parameter(short nDaNumber, short nDaWave, short nAdClock, short nAdNumber, short nAdConfig)

**Parameters:** nDaNumber: Number of DA samples in one waveform

nDaWave : Number of DA waveform to be output

nAdClock : AD sampling clock (samples/sec)=8M/ nAdClock

nAdNumber: Number of AD data to be read

nAdConfig : AD input range configuration code. Refer to "Section 3.6.5 "

#### Example:

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board is 0
Control.SetMFun2Parameter 320, 50, 80, 510, 0 'Set Mfun2's Parameter
For ii = 0 To 127 'produce arbitrary waveform by user, triangle waveform
  If ii < 64 Then
    CONTROL.SetMFun2DaVal ii, Str(&H800 - ii * 16) 'Set data No. and value
  Else
    CONTROL.SetMFun2DaVal (ii), Str(&H800 - (128 - ii) * 16)
    'Set data No. and value
  End If
Next ii
For ii = 128 To 319 'produce arbitrary waveform by user, square waveform
  If ii >= 128 And ii < 192 Then
    CONTROL.SetMFun2DaVal (ii), Str(&H800) 'Set data No. and value
  Elself (ii > 191 And ii < 256) Then
    CONTROL.SetMFun2DaVal (ii), Str(&H800 - 64 * 8)
    'Set data No. and value
  Else
    CONTROL.SetMFun2DaVal (ii), Str(&H800) 'Set data No. and value
  End If
Next ii
CONTROL.StartMFun2 'Start MFun2 Function
CONTROL.GetMFun2Buf wAd(0) 'Get MFun2 AD Buffer data
```

**Comments:** Please refer to the demo7 and it is not for PCI-1002.

### 3.8.4 SetMFun2DaVal()

This method is used to transfer the defined DA data order and its values into the **Mfun2** buffer. That means this function provides a way for user to generate arbitrary waveform.

**Prototype:** void SetMFun2DaVal (short DaNum, short DaVal)

**Parameters:** DaNum: Set DA data's number.

DaVal : Set Data's value

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board is 0
Control.SetMFun2Parameter 320, 50, 80, 510, 0 'Set Mfun2's Parameter
For ii = 0 To 127 'produce arbitrary waveform by user, triangle waveform
  If ii < 64 Then
    CONTROL.SetMFun2DaVal ii, Str(&H800 - ii * 16) 'Set data No. and value
  Else
    CONTROL.SetMFun2DaVal (ii), Str(&H800 - (128 - ii) * 16)
    'Set data No. and value
  End If
Next ii
For ii = 128 To 319 'produce arbitrary waveform by user, square waveform
  If ii >= 128 And ii < 192 Then
    CONTROL.SetMFun2DaVal (ii), Str(&H800) 'Set data No. and value
  ElseIf (ii > 191 And ii < 256) Then
    CONTROL.SetMFun2DaVal (ii), Str(&H800 - 64 * 8)
    'Set data No. and value
  Else
    CONTROL.SetMFun2DaVal (ii), Str(&H800) 'Set data No. and value
  End If
Next ii
CONTROL.StartMFun2 'Start MFun2 Function
CONTROL.GetMFun2Buf wAd(0) 'Get MFun2 AD Buffer data
```

**Comments:** Please refer to the demo7 and it is not for PCI-1002.

### 3.8.5 StartMFun2 ()

**StartMFun2** is used to start function **MFun2** to output the defined waveform to the device and obtain the defined data number of Analog input values into the buffer. (Refer to “multifunction boards Hardware Manual” chapter-5 for details) Here, the analog input channel\_0 and analog output channel\_0 are used to produce designated function.

**Prototype:** void StartMFun2()

**Example:**

```

wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board is 0
Control.SetMFun2Parameter 320, 50, 80, 510, 0 'Set Mfun2's Parameter
For ii = 0 To 127 'produce arbitrary waveform by user, triangle waveform
  If ii < 64 Then
    CONTROL.SetMFun2DaVal ii, Str(&H800 - ii * 16) 'Set data No. and value
  Else
    CONTROL.SetMFun2DaVal (ii), Str(&H800 - (128 - ii) * 16)
    'Set data No. and value
  End If
Next ii
For ii = 128 To 319 'produce arbitrary waveform by user, square waveform
  If ii >= 128 And ii < 192 Then
    CONTROL.SetMFun2DaVal (ii), Str(&H800) 'Set data No. and value
  ElseIf (ii > 191 And ii < 256) Then
    CONTROL.SetMFun2DaVal (ii), Str(&H800 - 64 * 8)
    'Set data No. and value
  Else
    CONTROL.SetMFun2DaVal (ii), Str(&H800) 'Set data No. and value
  End If
Next ii
CONTROL.StartMFun2 'Start MFun2 Function
CONTROL.GetMFun2Buf wAd(0) 'Get MFun2 AD Buffer data

```

**Comments:** Please refer to the demo7 and it is not for PCI-1002.

### 3.8.6 GetMFun2Buf ()

The method is used to get the Analog input data from data buffer of **MFun2** .

**Prototype:** void GetMFun2Buf(short FAR\* MFun2Buf)

**Parameters:** MFun2Buf : The beginning address of MFun2Buf which AD data is stored.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0 'Set the active board is 0
Control.SetMFun2Parameter 320, 50, 80, 510, 0 'Set Mfun2's Parameter
For ii = 0 To 127 'produce arbitrary waveform by user, triangle waveform
  If ii < 64 Then
    CONTROL.SetMFun2DaVal ii, Str(&H800 - ii * 16) 'Set data No. and value
  Else
    CONTROL.SetMFun2DaVal (ii), Str(&H800 - (128 - ii) * 16)
    'Set data No. and value
  End If
Next ii
For ii = 128 To 319 'produce arbitrary waveform by user, square waveform
  If ii >= 128 And ii < 192 Then
    CONTROL.SetMFun2DaVal (ii), Str(&H800) 'Set data No. and value
  ElseIf (ii > 191 And ii < 256) Then
    CONTROL.SetMFun2DaVal (ii), Str(&H800 - 64 * 8)
    'Set data No. and value
  Else
    CONTROL.SetMFun2DaVal (ii), Str(&H800) 'Set data No. and value
  End If
Next ii
CONTROL.StartMFun2 'Start MFun2 Function
CONTROL.GetMFun2Buf wAd(0) 'Get MFun2 AD Buffer data
```

**Comments:** Please refer to the demo7 and it is not for PCI-1002.

### 3.8.7 SetMFun3Parameter ()

This method provides a function to set up the configuration code of **Mfun3**. This subroutine can be used to set the frequency of DA output, how many DA waveforms can be generated during one working cycle, the Amplitude of DA output, AD sampling rate, how many AD data can be read during one working cycle, and the value of LowAlarm and HighAlarm for continuous waveform generation and capture of function **MFun3**.

**Prototype:** void SetMFun3Parameter(short nDaFrequency, short nDaWave, float fDaAmplitude, short nAdClock, short nAdNumber, float fLowAlarm, float fHighAlarm)

**Parameters:**

- nDaFrequency : DA output frequency = 1.8M/wDaFrequency (Pentium 120)
- nDaWave : How many number of DA wave form to be generated
- nDaAmplitude : Amplitude of DA output.

**Note: the hardware J1 must be selected as +/-10V**

- nAdClock : **AD sampling clock(samples/sec)=8M/ nAdClock**
- nAdNumber : How many number of AD data to be read
- nLowAlarm : Low alarm limit. if **value< fLowAlarm** → LowAlarm
- nHighAlarm : High alarm limit. if **value>fHighAlarm** → HighAlarm

**Example:**

```
Control.ActiveBoard = 0 'Set the active board
Control.SetMFun3ChannelConfig 0, 1, 0 ' Set Mfun3 channel configuration
Control.SetMFun3ChannelConfig 1, 1, 0 ' Set Mfun3 channel configuration
Control.SetMFun3ChannelConfig 2, 1, 0 ' Set Mfun3 channel configuration
' Set Mfun3 parameter
Control. SetMFun3Parameter 90, 10, 4.5, 24, 100, -10, 10
Control.StartMFUN3 AdBuf(0) 'Start MFun3 function and get the MFun3 buffer
```

**Comments:** Please refer to the demo9 and it is not for PCI-1002.

### 3.8.8 SetMFun3ChannelConfig()

This method is used to set up which AD channel is used to obtain the analog input data and its corresponding configuration code(input range) for MFun3.

**Prototype:** void SetMFun3ChannelConfig(short nChannel, short nChannelVal, short nConfig)

**Parameters:** nChannel : Define the analog input channel.  
                   nChannelVal : Define the analog input channel(the first parameter) is read or not.  
                   nConfig : Configuration code. Refer to "Section 3.6.5 Configuration Table"

**Example:**

```
Control.ActiveBoard = 0 'Set the active board
Control.SetMFun3ChannelConfig 0, 1, 0 ' Set Mfun3 channel configuration
Control.SetMFun3ChannelConfig 1, 1, 0 ' Set Mfun3 channel configuration
Control.SetMFun3ChannelConfig 2, 1, 0 ' Set Mfun3 channel configuration
Control. SetMFun3Parameter 90, 10, 4.5, 24, 100, -10, 10
' Set Mfun3 parameter
Control.StartMFun3 AdBuf(0) 'Start MFun3 function and get the MFun3 buffer
```

**Comments:** Please refer to the demo9 and it is not for PCI-1002.

### 3.8.9 StartMFun3 ()

This function is used to start the defined work of **MFun3** to generate the sine waveform image automatically and obtain analog input data from the defined AD channel. (Refer to “multifunction boards Hardware Manual” chapter-5 for details), where the analog output channel is channel\_0 and analog input channel and its corresponding gain is programmable.

**Prototype:** void StartMFun3(float FAR\* fAdBuf)

**Parameters:** fAdBuf: the beginning address of **fAdBuf** which AD data is stored

**Example:**

```
Control.ActiveBoard = 0 'Set the active board
Control.SetMFun3ChannelConfig 0, 1, 0 ' Set Mfun3 channel configuration
Control.SetMFun3ChannelConfig 1, 1, 0 ' Set Mfun3 channel configuration
Control.SetMFun3ChannelConfig 2, 1, 0 ' Set Mfun3 channel configuration
' Set Mfun3 parameter
Control. SetMFun3Parameter 90, 10, 4.5, 24, 100, -10, 10
Control.StartMFun3 AdBuf(0) 'Start MFun3 function and get the MFun3 buffer
```

**Comments:** Please refer to the demo9 and it is not for PCI-1002.

### 3.9 Special TwoMA / OneMA Methods

These functions provide a method to allow user to define how many data can be got and store into main memory. Due to the magicscan function is used, the following method is not suitable for PCI-1002 multifunction card.

**OneMA** is for One board operating.

**TwoMA** is for two boards operating simultaneously.

Method Name	Data type of returned value	Data size	Descriptions
<b>SetOneMAChannelConfig</b>	Void		Set channel configuration of OneMA.
<b>SetOneMAParameter</b>	Void		Set up the parameter of OneMA.
<b>OneMAStart</b>	Void		Start OneMA and store the data into buffer.
<b>OneMAReadStatus</b>	short	2 Bytes	Get the status of OneMA.
<b>OneMAStop</b>	Void		Stop the Function of OneMA.
<b>SetTwoMAChannelConfig</b>	Void		Set channel configuration of Card0 and Card1.
<b>SetTwoMAParameter</b>	Void		Set the parameter of TwoMA .
<b>TwoMAStart</b>	Void		Start TwoMA and store the data into buffer.
<b>TwoMAReadStatus</b>	short	2 Bytes	Get the status of TwoMA .
<b>TwoMAStop</b>	Void		Stop the Function of TwoMA .

The procedures for how to use these methods of OneMA / TwoMA -function are shown as following.

1. The operation procedure for OneMA:

OneMA: SetOneMAChannelConfig -> SetOneMAParameter -> OneMAStart  
 -> OneMAReadStatus -> OneMAStop

2. The operation procedure for TwoMA :

TwoMA : SetTwoMAChannelConfig -> SetTwoMAParameter-> TwoMAStart  
 -> TwoMAReadStatus -> TwoMAStop

### 3.9.1 SetOneMAChannelConfig()

This method is used to set up the AD channel to be scanned or not and configuration code of selected channel for the only one multifunction board.

**Prototype:** void SetOneMAChannelConfig (short nChannel, short nChannelVal, short nConfig)

**Parameters:**

- nChannel : Select the Card0 scan channel,
- nChannelVal : Set up scan(1) or no scan(0) for card0 selected channel,
- nConfig : Configuration code for card0 selected channel.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control.SetOneMAChannelConfig 0, 1, 0 'Set the OneMA channel0
configuration
Control.SetOneMAChannelConfig 1, 1, 0
'Set the OneMA channel1 configuration
Control.SetOneMAChannelConfig 2, 1, 0
'Set the OneMA channel2 configuration
Control.SetOneMAChannelConfig 7, 1, 0
'Set the OneMA channel7 configuration
Control.SetOneMAParameter 80, 2000000, 0 'Set the parameters of function
Control.OneMAStart Card0Buf0(0) 'Start OneMA and store the data into buffer.
RetVal = Control.OneMAReadStatus ' Get the status of OneMA
IF RetVal =128 Then
    Control.OneMAStop 'Stop OneMA function
End IF
```

**Comments:** Please refer to the demo21 and it is not for PCI-1002.

### 3.9.2 SetOneMAParameter ()

The method is used to set up the sampling rate, how many data can be read and priority of **OneMA** for the only one multifunction board. The priority setting is described in below.

**Prototype:** void SetOneMAParameter(short nClock0Div, long dwMaxCount0, short nPriority)

**Parameters:**

- nClock0Div : define Card0 AD sampling rate, which is 8M/nCard0ClockDiv
- dwMaxCount0 : Set how many analog input data can be read from the Card0.
- nPriority : Set the thread's priority. Here, the thread is used to get the AD data from this PCI multifunction board.

Priority Value	Priority ID
-2	THREAD_PRIORITY_LOWEST
-1	THREAD_PRIORITY_BELOW_NORMAL
0	THREAD_PRIORITY_NORMAL
1	THREAD_PRIORITY_ABOVE_NORMAL
2	THREAD_PRIORITY_HIGHEST
15	THREAD_PRIORITY_TIME_CRITICAL
others	THREAD_PRIORITY_NORMAL

**Example:**

```

wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control.SetOneMAChannelConfig 0, 1, 0
'Set the OneMA channel0 configuration
Control.SetOneMAChannelConfig 1, 1, 0
'Set the OneMA channel1 configuration
Control.SetOneMAChannelConfig 2, 1, 0
'Set the OneMA channel2 configuration
Control.SetOneMAParameter 80, 2000000, 0 'Set the parameters of function
Control.OneMAStart Card0Buf0(0) 'Start OneMA and store the data into buffer.
RetVal = Control.OneMAReadStatus ' Get the status of OneMA
IF RetValue =128 Then
Control.OneMAStop 'Stop OneMA function
End IF
    
```

**Comments:** Please refer to the demo21 and it is not for PCI-1002.

### 3.9.3 OneMAStart()

This method is used to start the function **OneMA** to capture analog input data and store into main memory.

**Prototype:** void OneMAStart(short FAR\* CardBuffer)

**Parameters:** CardBuffer : Buffer for storing AD data from Card0

**Example:**

```

wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control.SetOneMAChannelConfig 0, 1, 0
'Set the OneMA channel0 configuration
Control.SetOneMAChannelConfig 1, 1, 0
'Set the OneMA channel1 configuration
Control.SetOneMAChannelConfig 2, 1, 0
'Set the OneMA channel2 configuration
Control.SetOneMAParameter 80, 2000000, 0 'Set the parameters of function
Control.OneMAStart Card0Buf0(0) 'Start OneMA and store the data into buffer.
RetVal = Control.OneMAReadStatus ' Get the status of OneMA
IF RetValue =128 Then
Control.OneMAStop 'Stop OneMA function
End IF
    
```

**Comments:** Please refer to the demo21 and it is not for PCI-1002.

### 3.9.4 OneMAReadStatus()

This method is used to get the status of capturing data mechanism of function **OneMA**.

**Prototype:** short OneMAReadStatus ()

**Return:** 0 : Data is not ready, 1 : Function OneMA is working,  
8 :Data over flow, 128 : Data ready

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control.SetOneMAChannelConfig 0, 1, 0
'Set the OneMA channel0 configuration
Control.SetOneMAChannelConfig 1, 1, 0
'Set the OneMA channel1 configuration
Control.SetOneMAChannelConfig 2, 1, 0
'Set the OneMA channel2 configuration
Control.SetOneMAChannelConfig 7, 1, 0
'Set the OneMA channel7 configuration
Control.SetOneMAParameter 80, 2000000, 0 'Set the parameters of function
Control.OneMAStart Card0Buf0(0) 'Start OneMA and store the data into buffer.
RetVal = Control.OneMAReadStatus ' Get the status of OneMA
IF RetValue =128 Then
Control.OneMAStop 'Stop OneMA function
End IF
```

**Comments:** Please refer to the demo21 and it is not for PCI-1002.

### 3.9.5 OneMAStop()

This method is used to stop the capturing data mechanism of the function **OneMA**.

**Prototype:** void OneMAStop()

**Example:**

```

wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control.SetOneMAChannelConfig 0, 1, 0
'Set the OneMA channel0 configuration
Control.SetOneMAChannelConfig 1, 1, 0
'Set the OneMA channel1 configuration
Control.SetOneMAChannelConfig 2, 1, 0
'Set the OneMA channel2 configuration
Control.SetOneMAParameter 80, 2000000, 0 'Set the parameters of function
Control.OneMAStart Card0Buf0(0) 'Start OneMA and store the data into buffer.
RetVal = Control.OneMAReadStatus ' Get the status of OneMA
IF RetValue =128 Then
Control.OneMAStop 'Stop OneMA function
End IF
    
```

**Comments:** Please refer to the demo21 and it is not for PCI-1002.

### 3.9.6 SetTwoMAChannelConfig ()

This method sets up the AD channel to be scanned or not and configuration code of selected channel for the first and second multifunction boards.

**Prototype:** void SetTwoMAChannelConfig (short C0Channel, short C0ChannelVal, short C0ChannelConfig, short C1Channel, short C1ChannelVal, short C1ChannelConfig)

**Parameters:**

C0Channel : Select the Card0 scan channel,  
 C0ChannelVal : Set up scan(1) or no scan(0) for card0 selected channel,  
 C0ChannelConfig : Configuration code for card0 selected channel,  
 C1Channel : Select the Card1 scan channel,  
 C1ChannelVal : Set up scan(1) or no scan(0) for card1 selected channel,  
 C1ChannelConfig : Configuration code for card1 selected channel,

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control. SetTwoMAChannelConfig 0, 1, 0, 0, 1, 0
'Set the TwoMA channel configuration
Control. SetTwoMAChannelConfig 1, 1, 0, 1, 1, 0
Control. SetTwoMAChannelConfig 2, 1, 0, 2, 1, 0
Control.SetTwoMAParameter wClockDiv, wClockDiv, 20000, 20000, Priority
'Set the parameters of TwoMA function
Control.TwoMAStart Card0Buf0(0), Card0Buf1(0)
'Start TwoMA and store the data into buffer.
RetVal = Control.TwoMAREadStatus 'Read the TwoMA status
IF RetValue =128 Then
  Control.TwoMAStop 'Stop TwoMA function
Else iF (RetVal=8) then
  End 'Exit program
End IF
```

**Comments:** Please refer to the demo20 and it is not for PCI-1002.

### 3.9.7 SetTwoMAParameter ()

This method is used to set up the sampling rate, how many data can be read and priority for the first and second multifunction boards of **TwoMA**. The priority setting is described in below.

**Prototype:** void SetTwoMAParameter(short nCard0ClockDiv, short nCard1ClockDiv, long dwCard0MaxCount, long dwCard1MaxCount, short nPriority)

**Parameters:** nCard0ClockDiv : Define Card0 AD sampling rate, which is  $8M/nCard0ClockDiv$   
 nCard1ClockDiv : Define Card1 AD sampling rate, which is  $8M/nCard0ClockDiv$   
 dwCard0MaxCount: Set how many analog input data can be read from the Card0.  
 dwCard1MaxCount: Set how many analog input data can be read from the Card1  
 nPriority : Set the thread's priority. Here, the thread is used to get the AD data for these two PCI multifunction boards.

Priority Value	Priority ID
-2	THREAD_PRIORITY_LOWEST
-1	THREAD_PRIORITY_BELOW_NORMAL
0	THREAD_PRIORITY_NORMAL
1	THREAD_PRIORITY_ABOVE_NORMAL
2	THREAD_PRIORITY_HIGHEST
15	THREAD_PRIORITY_TIME_CRITICAL
others	THREAD_PRIORITY_NORMAL

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control. SetTwoMAChannelConfig 0, 1, 0, 0, 1, 0
'Set the TwoMA channel configuration
Control. SetTwoMAChannelConfig 1, 1, 0, 1, 1, 0
Control. SetTwoMAChannelConfig 2, 1, 0, 2, 1, 0
Control.SetTwoMAParameter wClockDiv, wClockDiv, 20000, 20000, Priority
'Set the parameters of TwoMA function
Control.TwoMAStart Card0Buf0(0), Card0Buf1(0)
'Start TwoMA and store the data into buffer.
RetVal = Control.TwoMAReadStatus 'Read the TwoMA status
IF RetVal =128 Then Control.TwoMAStop 'Stop TwoMA function
```

**Comments:** Please refer to the demo20 and it is not for PCI-1002.

### 3.9.8 TwoMAStart()

This method is used to start the function **TwoMA** to capture data and store into buffer.

**Prototype:** void TwoMAStart(short FAR\* Card0Buffer, short FAR\* Card1Buffer)

**Parameters:** Card0Buffer: Buffer for storing AD data from Card0

Card1Buffer: Buffer for storing AD data from Card1

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control. SetTwoMAChannelConfig 0, 1, 0, 0, 1, 0
'Set the TwoMA channel configuration
Control. SetTwoMAChannelConfig 1, 1, 0, 1, 1, 0
Control. SetTwoMAChannelConfig 2, 1, 0, 2, 1, 0
Control.SetTwoMAParameter wClockDiv, wClockDiv, 20000, 20000, Priority
'Set the parameters of TwoMA function
Control.TwoMAStart Card0Buf0(0), Card0Buf1(0)
'Start TwoMA and store the data into buffer.
RetVal = Control.TwoMAReadStatus 'Read the TwoMA status
IF RetVal =128 Then
    Control.TwoMAStop 'Stop TwoMA function
End IF
```

**Comments:** Please refer to the demo20 and it is not for PCI-1002.

### 3.9.9 TwoMAREadStatus()

This method is used to get the status of capture data mechanism by function **TwoMA**.

**Prototype:** short TwoMAREadStatus()

**Return:** 0 : Data is not ready, 1 : Function TwoMA is working,  
8 : Data over flow, 128 : Completely capture data.

**Example:**

```

wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control. SetTwoMAChannelConfig 0, 1, 0, 0, 1, 0
'Set the TwoMA channel configuration
Control. SetTwoMAChannelConfig 1, 1, 0, 1, 1, 0
Control. SetTwoMAChannelConfig 2, 1, 0, 2, 1, 0
Control.SetTwoMAParameter wClockDiv, wClockDiv, 20000, 20000, Priority
'Set the parameters of TwoMA function
Control.TwoMAStart Card0Buf0(0), Card0Buf1(0)
'Start TwoMA and store the data into buffer.
RetVal = Control.TwoMAREadStatus 'Read the TwoMA status
IF RetValue =128 Then
    Control.TwoMAStop 'Stop TwoMA function
End IF
    
```

**Comments:** Please refer to the demo20 and it is not for PCI-1002.

### 3.9.10 TwoMAStop()

This method is used to stop the **TwoMA** capture data mechanism.

**Prototype:** void TwoMAStop ()

**Example:**

```

wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control. SetTwoMAChannelConfig 0, 1, 0, 0, 1, 0
'Set the TwoMA channel configuration
Control. SetTwoMAChannelConfig 1, 1, 0, 1, 1, 0
Control. SetTwoMAChannelConfig 2, 1, 0, 2, 1, 0
Control.SetTwoMAParameter wClockDiv, wClockDiv, 20000, 20000, Priority
'Set the parameters of TwoMA function
Control.TwoMAStart Card0Buf0(0), Card0Buf1(0)
'Start TwoMA and store the data into buffer.
RetVal = Control.TwoMAReadStatus 'Read the TwoMA status
IF RetValue =128 Then
    Control.TwoMAStop 'Stop TwoMA function
End IF
    
```

**Comments:** Please refer to the demo20 and it is not for PCI-1002.

### 3.10 The Continuous Capture Methods

The following functions are used to continuously capture analog input data from multifunction boards. The methods can't be used with PCI-1002 series multifunction board because magicscan function is adapted.

Method Name	Data type of returned value	Data size	Descriptions
<b>SetFirstCardChannelConfig</b>	Void		Set channel configuration of first Card.
<b>FirstCardStartScan</b>	Void		Start data scan mechanism of first Card.
<b>GetFirstCardChannelData</b>	short	2 bytes	Get the data and status of first Card
<b>FirstCardStop</b>	Void		Stop the continuous capture of first Card
<b>SetSecondCardChannelConfig</b>	Void		Set channel configuration of second Card.
<b>SecondCardStartScan</b>	Void		Set start data scan mechanism of second Card.
<b>GetSecondCardChannelData</b>	short	2 bytes	Get data and status of second Card.
<b>SecondCardStop</b>	Void		Stop the continuous capture second Card

The procedures for how to use FirstCard/SecondCard-function is shown as following.

1. The operation procedure for FirstCard :

FirstCard: SetFirstCardChannelConfig -> FirstCardStartScan ->

GetFirstCardChannelData -> FirstCardStop

2. The operation procedure for SecondCard :

SecondCard: SetSecondCardChannelConfig -> SecondCardStartScan ->

GetSecondCardChannelData -> SecondCardStop

### 3.10.1 SetFirstCardChannelConfig ()

This method is used to set up the Channel Configuration of function **FirstCard()** for the first board.

**Prototype:** void SetFirstCardChannelConfig(short nChannel, short nChannelVal, short nConfig)

**Parameters:**

- nChannel : Select the scan channel for first card.
- nChannelVal : Set up scan(1) or no scan(0) for card0 selected channel.
- nConfig : Configuration code for selected channel.

**Example:**

```

wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control. SetFirstCardChannelConfig 0, 1, 0
'Set the FirstCardchannel0 configuration
Control. SetFirstCardChannelConfig 1, 1, 0
'Set the FirstCard channel1 configuration
Control. SetFirstCardChannelConfig 2, 1, 0
'Set the FirstCard channel2 configuration
Control. FirstCardStartScan 80, AdNumber
'Set the parameter and start continuous capture of function FirstCard
RetVal = Control.GetFirstCardChannelData(Buf(0), Buf1(0), wStatus)
' Get data and status of FirstCard
IF RetVal=0 then
    Control. FirstCardStop 'Stop the FirstCard
End IF
    
```

**Comments:** Please refer to the demo13 and it is not for PCI-1002.

### 3.10.2 FirstCardStartScan ()

This method is used to start continuous data capture mechanism of **FirstCard** by the configuration of AD sampling rate and how many data can be got from the selected channel.

**Prototype:** void FirstCardStartScan(short nSampleRate, short nCount)

**Parameters:**

nSampleRate : AD sampling rate = 8M/ nSampleRate.

nSampleRate=80    sampling rate=8M/80=100K

nCount            : How many AD data can be got in one batch time from selected scan channel

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control. SetFirstCardChannelConfig 0, 1, 0
'Set the FirstCard channel0 configuration
Control. SetFirstCardChannelConfig 1, 1, 0
'Set the FirstCard channel1 configuration
Control. SetFirstCardChannelConfig 2, 1, 0
'Set the FirstCard channel2 configuration
Control. FirstCardStartScan 80, AdNumber
'Set the parameter and start continuous capture of function FirstCard
RetVal = Control.GetFirstCardChannelData(Buf(0), Buf1(0), wStatus)
' Get data and status of FirstCard
IF RetValue=0 then
    Control. FirstCardStop 'Stop the FirstCard
End IF
```

**Comments:** Please refer to the demo13 and it is not for PCI-1002.

### 3.10.3 GetFirstCardChannelData ()

This method is used to return the first card scan states and get the AD data from continuous capture mechanism of Card0 into two format. The first format is in a circular queue of scan sequence control, like (012...N012...N.....012...N). The second way is to put the scan analog input data in channel format, like (00000.....11111.....22222.....NNNNN.....).

**Prototype:** short GetSecondCardChannelData(short FAR\* nSBuf, short FAR\* nCBuf,  
short FAR\* nStatus)

**Parameters:** nSBuf : Data buffer, which store data in scan sequence order format  
(012...N012...N.....012...N)

nCBuf : Data buffer, which store data in channel sequence order format  
(00000.....11111.....22222.....NNNNN.....)

nStatus :1=thread start, 2=TimeOut, 8=FIFO overflow, 0x80=thread finish

**Return:** 0: Data is ready 1: Data not ready

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control. SetFirstCardChannelConfig 0, 1, 0
'Set the FirstCard channel0 configuration
Control. SetFirstCardChannelConfig 1, 1, 0
'Set the FirstCard channel1 configuration
Control. SetFirstCardChannelConfig 2, 1, 0
'Set the FirstCard channel2 configuration
Control. FirstCardStartScan 80, AdNumber
'Set the parameter and start continuous capture of function FirstCard
RetVal = Control.GetFirstCardChannelData(Buf(0), Buf1(0), wStatus)
' Get data and status of FirstCard
IF the RetValue=0 then
Control. FirstCardStop 'Stop the FirstCard
ENDIF
```

**Comments:** Please refer to the demo13 and it is not for PCI-1002.

### 3.10.4 FirstCardStop ()

This method is used to stop **FirstCard** continuously capturing analog input data from the first board.

**Prototype:** void FirstCardStop()

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 0'Set the active board' number is 0
Control. SetFirstCardChannelConfig 0, 1, 0
'Set the FirstCard channel0 configuration
Control. SetFirstCardChannelConfig 1, 1, 0
'Set the FirstCard channel1 configuration
Control. SetFirstCardChannelConfig 2, 1, 0
'Set the FirstCard channel2 configuration
Control. FirstCardStartScan 80, AdNumber
'Set the parameter and start continuous capture of function FirstCard
RetVal = Control.GetFirstCardChannelData(Buf(0), Buf1(0), wStatus)
' Get data and status of FirstCard
IF the RetValue=0 then
Control. FirstCardStop 'Stop the FirstCard
ENDIF
```

**Comments:** Please refer to the demo13 and it is not for PCI-1002.

### 3.10.5 SetSecondCardChannelConfig ()

This method is used to set up the Channel Configuration of function **SecondCard** for the second board. Note that function **SecondCard** only can be used when Function firstCard is used firstly.

**Prototype:** void SetSecondCardChannelConfig(short nChannel, short nChannelVal,  
short nConfig)

**Parameters:**

nChannel : Select the scan channel for second card.  
nChannelVal : Set up scan(1) or no scan(0) for card1 selected channel.  
nConfig : Configuration code for selected channel.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 1'Set the active board' number is 1
Control. SetSecondCardChannelConfig 0, 1, 0
'Set the SecondCardchannel0 configuration
Control. SetSecondCardChannelConfig 1, 1, 0
'Set the SecondCardchannel1 configuration
Control. SetSecondCardChannelConfig 2, 1, 0
'Set the SecondCardchannel2 configuration
Control. SecondCardStartScan 80, AdNumber
'Set the parameter and start continuous capture of function SecondCard
RetVal = Control.GetSecondCardChannelData(Buf(0), Buf1(0), wStatus)
' Get data and status of SecondCard
IF RetValue=0 then
    Control. SecondCardStop 'Stop the SecondCard
End IF
```

**Comments:** Please refer to the demo13 and it is not for PCI-1002.

### 3.10.6 SecondCardStartScan ()

This method is used to start continuous data capture mechanism of Function **Secondcard** by the configuration of AD sampling rate and how many data can be got from the selected channel in SecondCard.

**Prototype:** void SecondCardStartScan(short nSampleRate, short nCount)

**Parameters:**

nSampleRate : AD sampling rate = 8M/ nSampleRate.

nSampleRate=80     sampling rate=8M/80=100K

nCount            : How many AD data can be got in one batch time from selected scan channel

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 1'Set the active board' number is 1
Control. SetSecondCardChannelConfig 0, 1, 0
'Set the SecondCardchannel0 configuration
Control. SetSecondCardChannelConfig 1, 1, 0
'Set the SecondCardchannel1 configuration
Control. SetSecondCardChannelConfig 2, 1, 0
'Set the SecondCardchannel2 configuration
Control. SecondCardStartScan 80, AdNumber
'Set the parameter and start continuous capture of function SecondCard
RetVal = Control.GetSecondCardChannelData(Buf(0), Buf1(0), wStatus)
' Get data and status of SecondCard
IF RetValue=0 then
    Control. SecondCardStop 'Stop the SecondCard
End IF
```

**Comments:** Please refer to the demo14 and it is not for PCI-1002.

### 3.10.7 GetSecondCardChannelData ()

This method is used to return the second card scan states and get the AD data of the second card from continuous capture mechanism of Card1 into two format. The first format is in a circular queue of scan sequence control, like (012...N012...N.....012...N). The second way is to put the scan analog input data in channel format, like (000...111...222...NNN).

**Prototype:** short GetSecondCardChannelData(short FAR\* nSBuf, short FAR\* nCBuf,  
short FAR\* nStatus)

**Parameters:** nSBuf : Data buffer, which store data in scan sequence order format  
(012...N012...N.....012...N)

nCBuf : Data buffer, which store data in channel sequence order format  
(000.....111.....222....NNN....)

nStatus: 1=thread start, 2=TimeOut, 8=FIFO overflow, 0x80=thread finish

**Return:** 0: Data is ready 1: Data not ready

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 1 'Set the active board' number is 1
Control. SetSecondCardChannelConfig 0, 1, 0
'Set the SecondCardchannel0 configuration
Control. SetSecondCardChannelConfig 1, 1, 0
'Set the SecondCardchannel1 configuration
Control. SetSecondCardChannelConfig 2, 1, 0
'Set the SecondCardchannel2 configuration
Control. SecondCardStartScan 80, AdNumber
'Set the parameter and start continuous capture of function SecondCard
RetVal = Control.GetSecondCardChannelData(Buf(0), Buf1(0), wStatus)
' Get data and status of SecondCard
IF RetVal=0 then
Control. SecondCardStop 'Stop the SecondCard
End IF
```

**Comments:** Please refer to the demo14 and it is not for PCI-1002.

### 3.10.8 SecondCardStop ()

This method is used to stop **SecondCard** continuously capturing analog input data from the second board.

**Prototype:** void SecondCardStop()

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard = 1 'Set the active board' number is 1
Control. SetSecondCardChannelConfig 0, 1, 0
'Set the SecondCardchannel0 configuration
Control. SetSecondCardChannelConfig 1, 1, 0
'Set the SecondCardchannel1 configuration
Control. SetSecondCardChannelConfig 2, 1, 0
'Set the SecondCardchannel2 configuration
Control. SecondCardStartScan 80, AdNumber
'Set the parameter and start continuous capture of function SecondCard
RetVal = Control.GetSecondCardChannelData(Buf(0), Buf1(0), wStatus)
' Get data and status of SecondCard
IF RetValue=0 then
Control. FirstCardStop 'Stop the SecondCard
End IF
```

**Comments:** Please refer to the demo14 and it is not for PCI-1002.

### 3.11 Interrupt Methods

Note that the following methods can't be used with other series multifunction board except PCI-1002 because interrupt method is adapted.

Method Name	Data type of returned value	Data size	Descriptions
<b>GetIrqNo</b>	short	2 bytes	Get the IRQ number of the active PCI-1002 board
<b>GetIrqCount</b>	long	4 bytes	Obtain the number of times which interrupt have happened.
<b>ADIrqStop</b>	void		Stop the interrupt
<b>ADIrqStart</b>	void		Start the interrupt.
<b>InstallIrq</b>	void		Install interrupt handler for a specific IRQ
<b>GetBuffer</b>	void		Transfer analog input data obtained from interrupt method into the user's buffer by word format.
<b>GetFloatBuffer</b>	void		Transfer analog input data obtained from interrupt method into the user's buffer by floating-point format.

### 3.11.1 GetIrqNo()

This method is used to get the IRQ number of the active PCI-1002 board installed in the system. It can let you know what IRQ number is used by PCI-1002. This function is not necessary method for your program.

**Prototype:** short GetIrqNo()

**Return:** IRQ number.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard =0 'Set the first card to active.
Print Control.GetIrqNo 'Print the IRQ number
```

**Comments:** Please refer to the Interrupt demo and it is only for PCI-1002.

### 3.11.2 GetIrqCount

This method is used to get IRQ counter value of the active PCI-1002 board installed in the system, which means how many times of PCI-1002 card's interrupt has been happened. This function is not necessary for your program.

**Prototype:** long GetIrqCount()

**Return:** Counter data.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard =0 'Set the first card to active.
Print Control.GetIrqCount 'Print the IRQ count number
```

**Comments:** Please refer to the Interrupt demo and it is only for PCI-1002.

### 3.11.3 ADIrqStop()

This method is used to stop the PCI-1002 card's interrupt and remove the installed interrupt handler.

**Prototype:** void ADIrqStop()

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard= 0 'Set the active board=0
Control.InstallIrq hEvent, 100 'Install the event and counter
Control.ADIrqStart 0, 0, c2
'Set the channel, configuration code and sampling rate of the ADIrqStart method.
.....
Control.GetFloatBuffer DataNo, WaveData(0)
'Set the parameter and get Interrupt data after ADIrqStart
Control.ADIrqStop 'Stop the analog convert to digital of Interrupt
```

**Comments:** Please refer to the Interrupt demo and it is only for PCI-1002.

### 3.11.4 ADIrqStart()

This method is used to start the PCI-1002 card's interrupt for a specific AD channel , programmable gain code and sampling rate.

**Prototype:** void ADIrqStart(short nChannel, short nGain, short wFreqDiv)

**Parameters:** nChannel: Select the AD channel.

nGain: Set the Gain for selected channel, refer to Hardware manual Section 3.2.7

wFreqDiv: The sampling rate is 2M/wFreqDiv.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard= 0 'Set the active board=0
Control.InstallIrq hEvent, 100 'Install the event and counter
Control.ADIrqStart 0, 0, 100 'the sampling rate is 20K
'Set the channel, configuration code and sampling rate of the ADIrqStart method.
.....
Control. GetFloatBuffer DataNo, WaveData(0)
'Set the parameter and get Interrupt data after the ADIrqStart function starts.
Control.ADIrqStop 'Stop the AD(Analog-Digital) Interrupt.
```

**Comments:** Please refer to the Interrupt demo and it is only for PCI-1002.

### 3.11.5 InstallIrq()

This method is used to install interrupt handler for a specific IRQ level n and set the maximum count number for interrupts.

**Prototype:** void InstallIrq(long\* IEvent, long Count)

**Parameters:** IEvent : The user must use the CreateEvent function to create the event object and obtain its handle and pass the handle into this function.

Count : Maximum numbers of counter to interrupt transfer.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard= 0 'Set the first card to active.
Control.InstallIrq hEvent, 100 'Install the event and counter
Control.ADIrqStart 0, 0, c2
'Set the channel, configuration code and sampling rate of the Start ADIrqStart
function.
.....
Control. GetFloatBuffer DataNo, WaveData(0)
'Set the parameter and get Interrupt data after the ADIrqStart starts.
Control.ADIrqStop 'Stop the AD(Analog-Digital) Interrupt.
```

**Comments:** Please refer to the Interrupt demo and it is only for PCI-1002.

### 3.11.6 GetBuffer()

This method is used to Transfer analog input data obtained from interrupt method into user's buffer by word format.

**Prototype:** void GetBuffer(long dwNum, short FAR\* wBuffer)

**Parameters:** dwNum: The total number to be transfered to user's buffer.

wBuffer: The first address of the wBuffer buffer (word Array) that store the AD value in HEX format.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard= 0 'Set the active board=0
Control.InstallIrq hEvent, 100 'Install the event and counter
Control.ADIrqStart 0, 0, c2
'Set the channel, configuration code and sampling rate for Start ADIrqStart
method.
.....
Control. GetBuffer DataNo, WaveData(0)
'Set the parameter and get Interrupt data after ADIrqStart
Control.ADIrqStop 'Stop the AD (Analog-Digital) Interrupt.
```

**Comments:** Please refer to the Interrupt demo and it is only for PCI-1002.

### 3.11.7 GetFloatBuffer()

This subroutine is used to transfer analog input data obtained from interrupt method into the user's buffer by floating point format.

**Prototype:** void GetFloatBuffer(long dwNum, float FAR\* fAdVal)

**Parameters:** dwNum: The total number to be transferred into user's buffer.

fAdVal : The first address of the fAdVals buffer (float Array) that store the analog input value into floating-point format.

**Example:**

```
wTotalBoard = Control.DriverInit 'Initialize the driver and get the total boards
Control.ActiveBoard =0 'Set the active board=0
Control.InstallIrq hEvent, 100 'Install the event and counter
Control.ADIrqStart 0, 0, c2
'Set the channel, configuration code and sampling rate for Start ADIrqStart
method.
.....
Control. GetFloatBuffer DataNo, WaveData(0)
'Set the parameter and get Interrupt data after ADIrqStart
Control.ADIrqStop 'Stop the AD (Analog-Digital) Interrupt.
```

**Comments:** Please refer to the Interrupt demo and it is only for PCI-1002.

## 3.12 General Events

The PCI1800X (PCI1602X, PCI1202X, PCI1002X) has only one "Event", as shown in the following:

---

### 3.12.1 OnError ()

This event is used for default procedure and it is called when an error occurs. You could code an error message when necessary.

**Prototype:** void OnError(long IErrorCode)

**Arguments:** IErrorCode : Error code in "long" data type.

The OCX passes this argument into the procedure. Please Refer to it for further using.

**Example:**

```
MsgBox "Error Code: " + Str(IErrorCode) + Chr(13) _  
      + "Error Message: " + Control.ErrorString
```