



Copyright 2003

Roper Scientific, Inc.
3660 Quakerbridge Rd
Trenton, NJ 08619
TEL: 609-587-9797
FAX: 609-587-1970

All rights reserved. No part of this publication may be reproduced by any means without the written permission of Roper Scientific, Inc.

Printed in the United States of America.

ActiveX, Visual Basic, Windows, and Windows NT are registered trademarks of Microsoft Corporation in the United States and/or other countries.

The information in this publication is believed to be accurate as of the publication release date. However, Roper Scientific, Inc. does not assume any responsibility for any consequences including any damages resulting from the use thereof. The information contained herein is subject to change without notice. Revision of this publication may be issued to incorporate such change.

Table Of Contents

Overview	7
Beginning in Visual Basic®	9
Basic Example	9
Data Collection Example	11
Data Manipulation Example	12
Things to Note about VB	12
Calling Syntax	12
Getting Data	13
Conclusion	13
WinX/32 Automation Objects	15
Introduction.....	15
Experiment.....	16
ExpSetup	16
ROIrect.....	18
ValidRange.....	19
DocFile	20
DocFiles	23
CalibObj	23
DocInfo.....	24
Display	25
DocFile	25
DocWindow	25
DocWindows	27
CalibObj	28
ROIrect.....	28
CursorObj.....	28
WinXColorTable.....	29
Data Processing.....	30
Overview	30
Configuring the Input	31
Configuring the Output	32
Configuring the Convolution Kernel.....	32
Configuring the Process	33
Running an Arithmetic/Logical Process.....	33
Running a Non-Arithmetic/Logical Process	34
Utilities.....	35
WinX32App	35
PrintWindow	38
PrintWindows.....	40
ArrayConverter.....	40
Spectrograph	41
SpectroObj.....	41
SpectroObjMgr.....	44

ValidRange.....	44
Pulser	45
PITG	45
ValidRange.....	48
WinX32 Automation Enumerations	49
Experiment.....	49
EXP_CMD (ExpSetup Parameters)	49
DM_CMD (DocFile Parameters)	58
Display	61
DI_PARAM (DocWindow Parameters).....	61
Data Processing.....	64
PIDP_IMAGEMATHOPERATION (Arithmetic/Logical Operators).....	64
PIIP_SetGetParam (Configurable Process Parameters)	65
PRC_PARAM (Process Output Control Parameters)	66
PROCERR (Process Error Codes)	67
ProcFileAccessType (Process Output File Access Types).....	67
ProcSaveOpts (Process Output File Save Options).....	67
Spectrograph.....	68
NAMETYPE (Spectrograph Instrument Name Types).....	68
SPEC_ERROR_CODE (General Spectrograph Error Codes)	68
SPEC_STEP_ERRORCODE (Stepper Motor Error Codes).....	68
SPECT_DLL_TYPES (Spectrograph DLL Types).....	69
SPT_CMD (General Spectrograph Parameters).....	69
SPT_PROCESS (General Spectrograph Processes).....	75
Pulser	77
PITG_BRACKET_TYPE (PI-Max bracket type).....	77
PITG_COUPLING (External trigger coupling)	77
PITG_DELAY_FROM (Where delay is calculated from).....	77
PITG_ERROR_CODES (PITG error codes)	77
PITG_GATINGMODE (PG200:gating mode status)	78
PITG_INCREMENT_TYPES (Sequential increment types).....	78
PITG_PTG_TRIG_COUNT_TYPE (PTG triggers counted).....	78
PITG_PULSING_MODE (Pulsing mode).....	79
PITG_SLOPE (External trigger slope).....	79
PITG_TERMINATION (External trigger termination)	79
PITG_TRIGMODES (Pulse triggering modes)	79
PITG_TRIGSYNCMODES (Where to synch PG200 to)	79
PITG_TYPES (Pulser types).....	80
TG_CMD (Pulser settings).....	80
TG_PROCESS (Pulser download processing).....	83
ActonSpec Type Library Enumerations	85
GRATINGSPEEDS (Speeds at which grating can be moved).....	85
MANUFACTURERS (Name of spectrograph manufacturer).....	85
MIRRORLOCATIONS (Port locations of mirrors)	85
MIRRORPOSITIONS (Deflection position of mirror)	85
SLITLOCATIONS (Port location of slits).....	86
SPECTROMETERS (Acton spectrometers supported).....	86

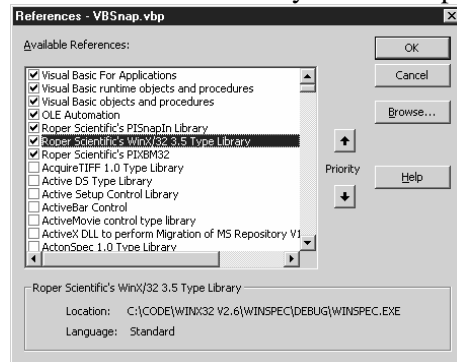
SpexSpec Type Library Enumerations.....	87
MANUFACTURERS (Name of spectrograph manufacturer).....	87
MIRRORLOCATIONS (Port locations of mirrors)	87
MIRRORPOSITIONS (Deflection position of mirror)	87
SLITLOCATIONS (Port location of slits).....	87
SPECTROMETERS (Spex spectrometers supported).....	88
WinX/32 Automation and VBScript.....	89
Replacement Scripting Functions	89
Creating a Snap-In DLL.....	91
Introduction.....	91
Using the Wizard:	91
Making Your Snap-In Appear on the WinX/32 Toolbar:	91
Converting a Visual Basic "Exe" project to a "Snap-In":	92
Putting up Your Main Form (omit if your program doesn't display a form):.....	92
Running Your DLL:.....	93
Index	95

This page intentionally left blank.

Overview

WinX/32 Automation provides a way to write programs to control the WinView/32 and WinSpec/32 family of Roper Scientific applications (hereafter referred to as WinX/32). This ability was provided to users of the previous generation of RS software via Macro Basic. Automation allows programs to be written in Visual Basic® or any other application language that supports the Microsoft® Windows® Automation standard, such as the macro languages in Microsoft® Excel and Microsoft® Word. (*Roper Scientific technical support will only support Microsoft® Visual Basic, Microsoft® Excel, and Microsoft® Word.*)

Each application that supports Automation provides entry points through objects whose methods can be called by an outside program. Windows uses a file called a *type library* to



provide information about what objects an application provides. The Windows registry stores the location of the type library. When WinX/32 is installed, the WinX/32 type library is registered. This means that “Roper Scientific's WinX/32 3.X Type Library” will show up in Windows’ list of automation support providers. Visual Basic and other programs provide a way to look at the list and choose which applications the user’s program will call. In Visual Basic the facility is called the *References* browser.

Automation programs may be written either as stand-alone programs (executables) or as dynamic link libraries (DLLs.) One advantage of a DLL user program is that WinX/32 will automatically show a menu item and a toolbar button for it. DLL user programs that are integrated with WinX/32 this way are called “Snap-Ins”. The easiest way to produce a program is to write a Visual Basic executable. Visual Basic Snap-Ins require additional effort to produce and have some limitations: they always run as modal, so there is no possibility of interacting with the WinX program while a VB Snap-In is running.

Microsoft supplies three versions of Visual Basic: the Learning Edition, the Professional Edition, and the Enterprise Edition. You must have the Professional or Enterprise edition in order to produce a DLL. The Learning Edition can produce a compiled executable program, but it may not run as fast as one compiled with the Professional or Enterprise editions. The examples in this document were produced using the Professional Edition.

Once your program is written, you can run it by clicking on the Run button in Visual Basic. This is a good way to debug your program. When you are fairly confident that the program is bug-free, you can compile it as an executable. The program can then be run in the normal way from an explorer window, or from the Execute Macro menu item in WinX/32. The Execute Macro menu item remembers the last 8 macros, so they can be selected quickly from a list.

A Visual Basic executable project can be converted to run as a Snap-In DLL. You must know how to add a resource file to the project; also, Visual Basic Snap-Ins have limitations that stand-alone programs do not have. Roper Scientific also supplies a **Visual Basic Snap-In Wizard**, which will set up a new Snap-In DLL project.

The macro languages of Microsoft programs such as Excel or Word also use the Visual Basic syntax and can be used to write programs that call WinX/32 functions. These macro programs run in interpreted mode, so things like For...Next loops will be slower than with a compiled program. However, if your macro mainly consists of calls to built-in functions of these applications, the overall speed of your macro may still be very good.

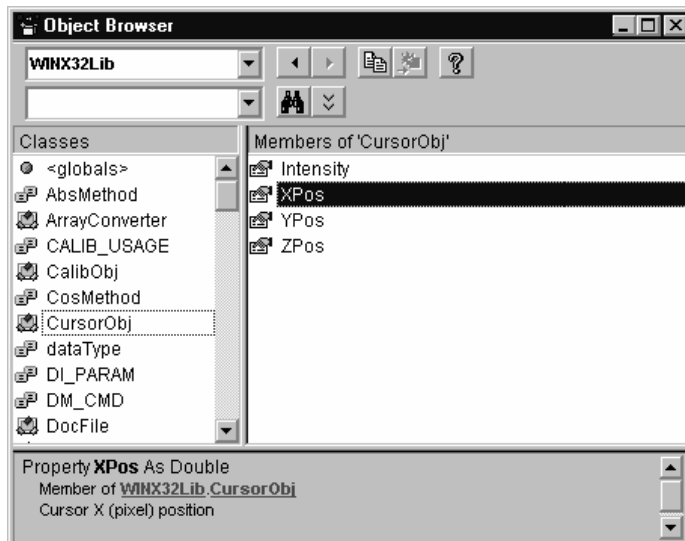
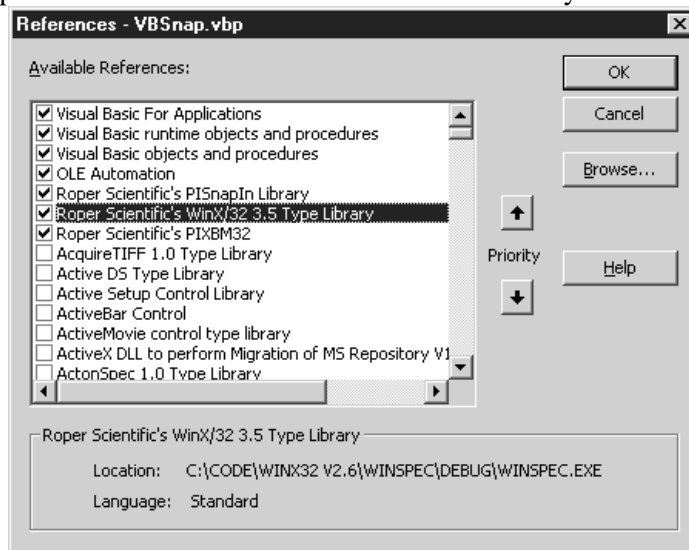
Beginning in Visual Basic®

Note: The examples in this section were produced using the Visual Basic Professional Edition.

Basic Example

This first example describes how to create a Visual Basic executable program that can automate WinX/32.

1. Start a new project of the type "Standard Executable". Visual Basic will create a project with a standard form or dialog. Your VB program will display this form when it first starts to run. Usually you will write your VB program by drawing controls on this form and by specifying what happens when a user interacts with these controls. If you want to make the program begin some action immediately when it starts, without waiting for user interaction, put the code in the **Load** function for the start-up form, or else set a function or subroutine as the start-up object instead of the form.
2. To gain access to the WinX/32 objects, go to the References browser (in the Project menu) and check the item "Roper Scientific's WinX/32 3.X Type Library". If this item is not present, then WinX/32 was not installed properly.
3. Once the correct references have been selected, the Visual Basic Object Browser (under the View menu) will show all of the type information available in the selected type libraries. To see inside the WinX/32 type library, use the drop-down box at the top of the Object Browser to select "WINX32Lib"; it defaults to "All". For each item, a short help string or "hint" is available and will be



displayed at the bottom of the Object Browser window when you select it. Part of this hint will also appear when you are typing the method or property into your program. You can leave this window open while you are writing your program to show what type information is available to you.

4. A WinX/32 automation object can be created in Visual Basic using the following syntax:

```
Dim object name As New Winx32Lib.object type
```

For example:

```
Dim objWinX As New WinX32App
```

5. You can omit the "Winx32Lib" part of these statements if you are sure no other type library also supplies an object of that type. For instance, if some other program also had an object called "DocFile", and you selected both that program and WinX/32 in the References browser, then VB would have no way of knowing which DocFile object to create.

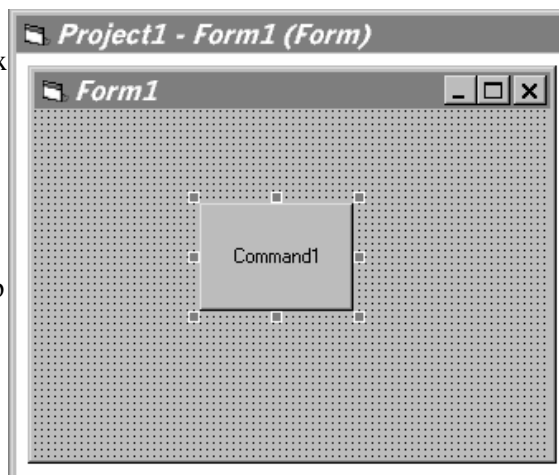
Once the object is created, you can access its properties and methods.

```
objWinX.ShowDemoBox "Hello World!"
```

6. Create a button on your new form by clicking the button icon on the VB toolbox and then drawing a box on the form. When you are done you should see something like the form on the right.

If you have trouble figuring out how to place a button on a form, or want more information, look in the Visual Basic Help Contents for "Designing a Form".

7. To get to the code that will execute when a user clicks on the new button, double-click on it with the mouse. You will be taken to a code window looking like the one below.



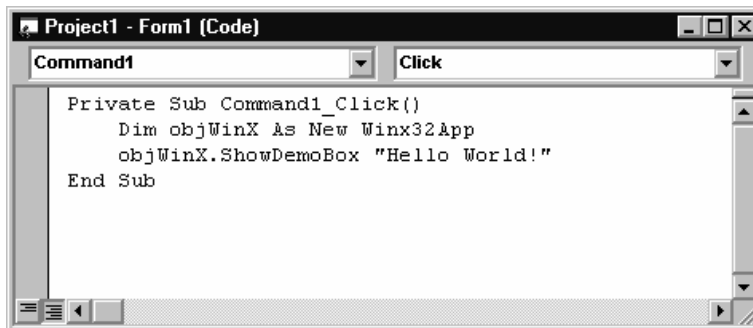
8. Into that window, after the line that says

```
Private Sub Command1_Click()
```

type the two lines already given, that is:

```
Dim objWinX As New WinX32App
objWinX.ShowDemoBox "Hello World!"
```

The result should look like the figure below:



9. Now just click on the Start button in the VB menu bar, or else choose the Start item under Run. You should see your new form come up. Click on the button. If a WinX/32 program wasn't running yet, it will start up at that point, and display a message box with the contents "Hello World!"

Data Collection Example

Next try giving your program the ability to collect an image or spectrum. Stop the running VB program, and replace the code in **Command1_Click()** with the following:

```
Private Sub cmdTest_Click() Dim objExp As New ExpSetup
Dim objDoc As DocFile
If objExp.Start(objDoc) Then ' start the experiment
Dim intStatus As Integer
While objExp.GetParam(EXP_RUNNING, intStatus) And intStatus = 0
DoEvents ' wait for the experiment to finish
Wend
If intStatus Then ' check for errors
MsgBox "Error running experiment."
End If
Dim vntFrame As Variant ' get the first frame
objDoc.GetFrame 1, vntFrame
Dim intIndex As Integer
Dim strData As String ' display some points
For intIndex = 0 To 4
strData = strData & vntFrame(intIndex, 0) & ", "
Next intIndex
strData = Left(strData, Len(strData) - 2)
MsgBox strData
End If
End Sub
```

This little routine will start an acquisition, wait until it is done, retrieve the first frame of data, and display a few values from it.

Data Manipulation Example

Finally try doing something with the data and putting it back into WinX/32. Replace the code in Command1_Click with the following example that takes an image, subtracts an offset, then puts the image back into WinX/32

```
Private Sub cmdTest_Click()
    Dim objExp As New ExpSetup
    Dim objDoc As DocFile
    If Not objExp.Start(objDoc) Then      ' start the experiment
        MsgBox "Error running experiment."
        Exit Sub
    End If
    Dim intStatus As Integer
    While objExp.GetParam(EXP_RUNNING, intStatus) And intStatus = 0
        DoEvents      ' wait for the experiment to finish
    Wend
    If intStatus Then      ' check for errors
        MsgBox "Error running experiment."
        Exit Sub
    End If
    Dim vntFrame As Variant ' get the first frame
    objDoc.GetFrame 1, vntFrame
    Dim intXDim As Integer ' determine the dimensions
    intXDim = objDoc.GetParam(DM_XDIM)
    Dim intYDim As Integer
    intYDim = objDoc.GetParam(DM_YDIM)
    Dim intX As Integer
    Dim intY As Integer
    For intY = 0 To intYDim - 1      ' subtract the offset
        For intX = 0 To intXDim - 1
            vntFrame(intX, intY) = vntFrame(intX, intY) - 10
        Next intX
    Next intY
    objDoc.PutFrame 1, vntFrame      ' put the data back
    objDoc.Update      ' update the display
End Sub
```

Things to Note about VB

Calling Syntax

Visual Basic differentiates between **Functions**, which return a value, and **Subroutines**, which do not. When calling either, you can put the arguments in parentheses or else just list them after the function name.

- ◆ For example, the **function** SetParam returns a value to indicate whether it succeeded in setting the designated parameter. If you don't want to bother checking whether SetParam succeeds (although you really should!) you can call it without assigning the return value to anything:

```
Dim objExp As New ExpSetup
objExp.SetParam(EXP_EXPOSURE, 0.1) ' with parentheses (gives
                                   syntax error)
objExp.SetParam EXP_EXPOSURE, 0.1 ' without parentheses
```

The line with parentheses above will generate a syntax error if you enter it as shown. This is because `SetParam` is defined as a **function** which returns a Boolean value. VB won't allow you to write the call to a **function** in the parenthesized form unless you also assign the return value to something; for instance:

```
intStatus = objExp.SetParam(EXP_EXPOSURE, 0.1) ' with parentheses;
this will work
```

In this case, VB creates a variable of type Boolean and assigns the return value of `SetParam` to it.

- ◆ The **function** `GetParam` returns the value of the designated parameter as its return value, for example:

```
dblExposure = objExp.GetParam(EXP_EXPOSURE) '
```

so you must use the syntax with parentheses.

- ◆ **Subroutines** are also sensitive to the presence or absence of parentheses. In this case, if you use parentheses, you have to proceed the **subroutine** call with the **Call** keyword:

```
objWin.SetPosition top, left, right, bottom ' without parentheses
Call objWin.SetPosition(top, left, right, bottom) ' with
parentheses
```

- ◆ More difficulties arise with **functions** that take a variable number of arguments; for instance, the `Print` **function** of `PrintWindow`. The best way to call this **function** is to use the form with parentheses, and assign the return value:

```
intStatus = objPrint.Print(1, 1, 0, "This is ", 21, " characters")
' this will work
```

Getting Data

When you use the `ExpSetup` object's `Start` or `StartFocus` function to begin collecting data, WinX/32 launches a separate thread to run the acquisition task. This means that your VB program can continue to run while data is being collected. Therefore, you may need to cause the program to wait until data is actually available before proceeding. Visual Basic provides the `DoEvents` function to allow a program to wait, while allowing the user to interact with the program in such ways as moving the window, clicking on a Stop button, etc. The code fragment below illustrates this process:

```
Dim objDoc As DocFile ' a document for the data
Dim vntFrame As Variant ' a local array for the data
Dim objExp As New ExpSetup ' to start data collection
objExp.Start(objDoc) ' take one set of frames and stop
While objExp.GetParam(EXP_RUNNING) ' wait for data to be acquired
    DoEvents ' allow Windows to process messages
Wend
objDoc.GetFrame 1, vntFrame ' now get data
```

Conclusion

At this point, you know enough to automate many common tasks, using stand-alone Visual Basic programs to control `WinView` or `WinSpec`. To find out how to create a Snap-In DLL, see the section entitled “Creating a Snap-In DLL”.

This page intentionally left blank.

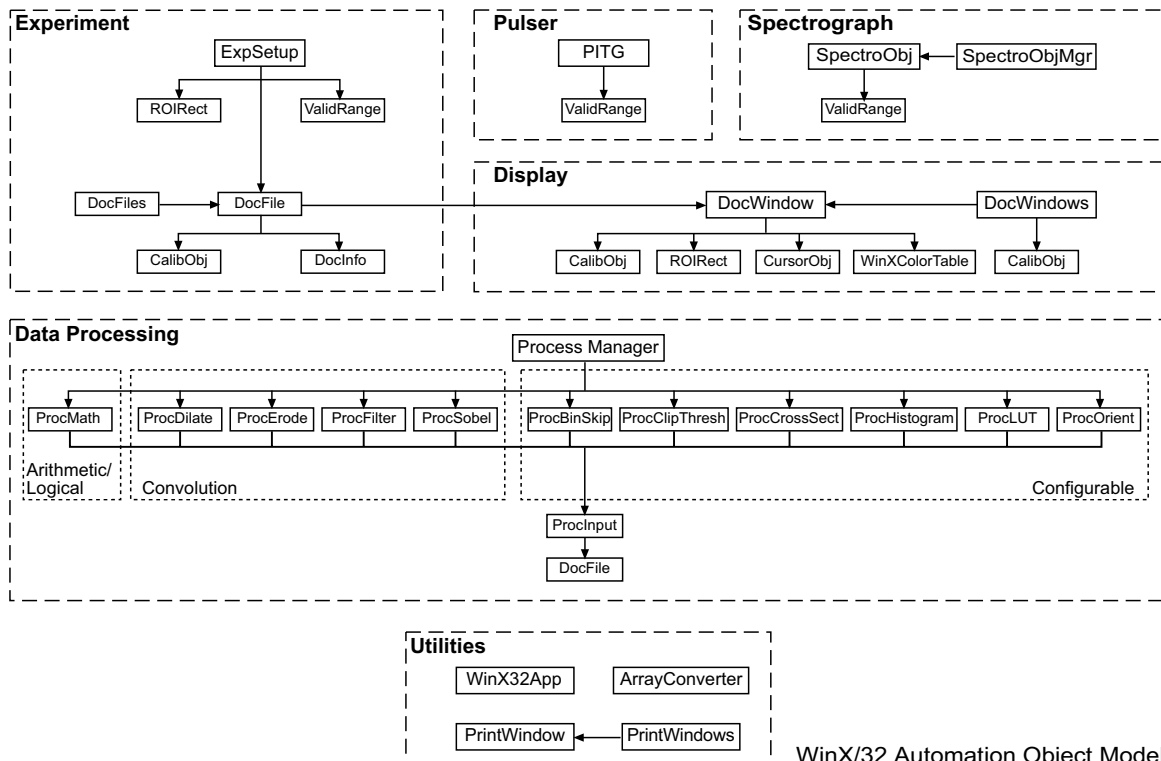
WinX/32 Automation Objects

Introduction

What follows is a list of the WinX/32 Automation objects. Each section corresponds to a different functionality of WinX/32. Below is the object model that shows all of the objects and how they relate to each other within each section.

WinX/32 Automation Object Model:

Functionality	Description
Experiment	Objects that control an experiment and data acquisition
Pulser	Objects that program a timing generator
Spectrograph	Objects that program a spectrograph
Display	Objects that control how data is displayed
Data Processing	Objects used for post-processing data via arithmetic, logical, kernel or other operations
Utilities	Objects used to display information to the user or make data handling more efficient



WinX/32 Automation Object Model

Experiment

ExpSetup

This object represents a WinX/32 experiment. It controls experiment parameters as well as data acquisition.

Function GetParam(Param As enum EXP_CMD, result As Integer) As Any

Function SetParam(Param As enum EXP_CMD, Value As Any) As Integer

Get and set experiment parameters (see the **EXP_CMD** parameters list for details).

For more information see GetParam function in the DocFile section, and also the discussion of Calling Syntax under "Things to Note".

Function GetDocument() As DocFile

Returns a pointer to an object of type DocFile. If the returned value is not empty, it is the data file associated with the current or last data collection. An empty return value indicates that either no data collection has run since the program started, or that the data file from the data collection was closed. You can use the returned object to call the DocFile interface functions for the document.

Function Start(File As DocFile) As Boolean

Function Start2(optional pRes) As DocFile

Function StartFocus(File As DocFile) As Boolean

Function StartFocus2(optional pRes) As DocFile

Begin collecting data. **Start** causes all of the frames in the experiment to be acquired; then the acquisition stops. **StartFocus** acquires the first frame in the experiment repeatedly, until the **Stop** function is called (or until the WinX user stops the acquisition). If the DocFile object supplied is empty, WinX/32 will create a new document and give it the default name for new documents, and the new data will be put into this document. If the DocFile object supplied is not empty, then the new data will be put into it. If an empty DocFile object is supplied, then on return it will be set to the new data file. The Boolean value indicates whether data collection was successfully started. The second version of these functions are used when it is necessary to bind to a DocFile object with the **Set** statement (as in scripting).

Example:

```
Dim objExp As New ExpSetup
Dim objDoc As New DocFile ' leave doc empty, WinX will supply
objExp.Start objDoc ' will acquire one set and stop
```

Function Stop() As Boolean

Stop collecting data. The return value indicates whether data collection was in progress before the function was called.


```
Function IsAvail(Param As enum EXP_CMD, optional Range
As ValidRange) As Boolean
Function IsAvail2(Param As enum EXP_CMD, optional pRes)
As ValidRange
```

Determines whether a feature corresponding to one of the parameters is available. You can call this function to find if the value returned from a given GetParam parameter will be valid, or to find whether the parameter value is able to be set via SetParam and what the range of valid values are. The second version of this function is used when it is necessary to bind to a ValidRange object with the **Set** statement (as in scripting).

Example:

```
Dim objExp As New ExpSetup
Dim objValid As ValidRange
Dim intStatus As Integer
intStatus = objExp.IsAvail(EXP_ADC_TYPE, objValid) ' Can we set
the ADC type?
If intStatus And vR.CurrentValue <> 2 And objValid.MinValue <=
2 And objValid.MaxValue >= 2 Then
intStatus = objExp.SetParam(EXP_ADC_TYPE, 2) ' Yes
End If
```

```
Function GetROI(index As Integer) As ROIrect
Function SetROI(Rect As ROIrect) As Boolean
```

Gets or sets a detector ROI. ROIs determine how pixels are read from the detector. The value of index must be 1 or greater. The rectangle information is passed as an ROIrect object, which is described below. You must call SetParam with the EXP_USEROI parameter to control whether the ROIs are used. You should call ClearROIs before setting up one or more ROIs with SetROI.

The GetROI function can return a “no” ROIrect object if the WinX/32 software has no ROIs set up. Be sure to check for this condition as in the following example:

Example:

```
Dim objExp As New ExpSetup
Dim objROI As ROIrect
Set objROI = objExp.GetROI(0)
If TypeName(objROI) = "Nothing" Then
MsgBox "No ROIs are available"
End If
```

Sub ClearROIs()

Erases all of the detector ROIs. This forces the detector to use the entire array for the next data collection. You can call SetParam with the EXP_USEROI parameter to control whether the ROIs are used without erasing them. You should call ClearROIs before setting up one or more ROIs with SetROI.

Function Load As Boolean

Function Save As Boolean

These functions allow you to save and restore the entire experiment setup in a disk file. The name of the file is set by a previous call to the SetParam function with the EXP_SETUPNAME parameter. The return value is False if the operation did not succeed, otherwise it is True.

Function WaitForExperiment As Boolean

This function waits for any pending data collection to complete. It is provided for programmers who do not wish to write DoEvents loops in their own code. The Start and StartFocus methods of ExpSetup return control almost immediately to the calling program. Often, a simple VB program that collects data will want to wait for the data collection to finish before proceeding. To do so, simply call this function:

If you want your program to be able to do some processing while data collection is proceeding, you will have to write a loop using the VB DoEvents keyword (see the example on page 11.)

If the return value is False, then either there was no controller or the program was stopped before data collection completed.

Function AcquireBackground() As Boolean

This function acquires a background file as if the user clicked on the "Acquire Background" toolbar button. Returns True on success.

Function AcquireFlatfield() As Boolean

This function acquires a flatfield file as if the user clicked on the "Acquire Flatfield" toolbar button. Returns True on success.

ROIrect

The ROIrect object represents a region of interest. It is used by the GetROI and SetROI functions of the DocWindow and ExpSetup objects.

Properties:

Name	Native Type	Description
Top	Double	X coordinate of the top left corner of the rectangle
Left	Double	Y coordinate of the top left corner of the rectangle
Bottom	Double	Y coordinate of the bottom right corner of the rectangle
Right	Double	X coordinate of the bottom right corner of the rectangle
XUnits	XCALIBUNIT	Units of the X coordinates
YUnits	YCALIBUNIT	Units of the Y coordinates
XGroup	Long	"Group size" used in the X direction (only significant for ExpSetup ROIs)
YGroup	Long	"Group size" used in the Y direction (only significant for ExpSetup ROIs)

```

Sub Get(top As Double, left As Double, bottom As
        Double, right As Double, xg As Long, yg As
        Long)
Sub Set(top As Double, left As Double, bottom As
        Double, right As Double, xg As Long, yg As
        Long)

```

Use these functions to get or set the ROI at once, without having to set each property.

ValidRange

The ValidRange object represents valid values that a parameter can hold. It is returned from the IsAvail function of the ExpSetup, PITG, and SpectroObj objects.

Properties:

Name	Native Type	Description
AvailValues	Long Array (only when DataType is X_ENUM, Empty otherwise)	All discrete, legal enum values for this parameter (only valid when DataType is X_ENUM)
Count	Integer	Number of valid items for this parameter
DataType	Integer	Datatype used for this parameter (specified by the dataType enum)
MinValue	Double	Minimum legal value for this parameter
MaxValue	Double	Maximum legal value for this parameter
DefaultValue	Double	Default value for this parameter
CurrentValue	Double	Current value of this parameter
Increment	Double	Minimum increment value for this parameter
ReadOrWrite	ExpRdWrType	Able to call get/set params

DocFile

This object represents a WinX/32 data file. Besides creating one, this object can be retrieved through the following objects: ExpSetup, DocWindow, and all process objects.

Properties:

Name	Native Type	Description
Title	String	The title in its associated DocWindow (excluding the dimensions)

Function Open(Name As String [, xLen As Integer, yLen As Integer, zLen As Integer, datatype As Long, newName As String]) As Boolean

Gets a pointer to an open document. If a name is supplied as the first parameter, then a pointer to an open document is returned, or a document is opened from disk (the document must already exist). The other parameters are optional, if Name is not a null string. If Name is "", Open creates a new document using the values for xLen, yLen, zLen and datatype to determine the size of the document, and newName to determine the name. If newName is "", the new document gets the default name. If xLen, yLen, or zLen are 0, the current controller setup determines the size of the document.

The data type is specified by the dataType enum.

Example:

```
Dim objDoc As New DocFile
objDoc.Open "Circuit.spe"
```

Function OpenNew(Name As String, pInfo As DocInfo) As Boolean

This is an alternative way to open a data file with the DocFile object. The String is the filename of a pre-existing file. If the String is empty, a new document will be created. This method uses the DocInfo object to specify information about the new document. In addition to the parameters used in the Open function, such as Xlen, Ylen, and Datatype, the DocInfo object allows you to specify whether to open the file in Append mode and whether to display the file in a window.

When a data file is opened in Append mode, any old file with the same name is not erased. Instead, the number of frames specified in the pInfo structure is added onto the end of the old file. The parameter DM_LASTFRAMERDY specifies which is the last frame with valid data. When a new file is first opened, it is set to zero. If an old file is appended to, then DM_LASTFRAMERDY is set to the number of frames in the file before the append occurred. The PutFrame and PutStrip functions update the DM_LASTFRAMERDY parameter; However, if you put data into the file using PutPixel, you must update DM_LASTFRAMERDY yourself.

For more information, including how to use the bShowWindow member of DocInfo, see the description of DocInfo.

Function GetWindow() As Object

Returns a pointer to an object of type DocWindow, which you may use to call the DocWindow interface (display) functions for a window displaying this document.

```
Sub GetFrame(frame As Integer, buffer As Variant)
Sub PutFrame(frame As Integer, buffer As Variant)
Sub GetStrip(frame As Integer, strip As Integer, buffer
    As Variant)
Sub PutStrip(frame As Integer, strip As Integer, buffer
    As Variant)
Function GetPixel(frame As Integer, strip As Integer,
    pixel As Integer) As Variant
Sub PutPixel(frame As Integer, strip As Integer, pixel
    As Integer, pixel As Variant)
```

GetFrame copies the data from a document into a Visual Basic array. If buffer is an empty Variant, **GetFrame** creates an array of the proper size and data type and sets buffer to point to it before copying the data.

PutFrame copies the data from a Visual Basic array into a document. The array should be one created with **GetFrame** to ensure that the size and data type match.

GetStrip and **PutStrip** are similar to **GetFrame** and **PutFrame**; they copy one strip to or from the document.

GetPixel returns a single pixel value from the document. **PutPixel** sets the value of a single pixel in the document. The pixel value is automatically converted from the type of the Visual Basic variable used to the correct data type for the document.

Example:

See **Data Collection Example** (page 11) in the "Beginning Visual Basic" section.

Sub AllocFrame(buffer As Variant)

AllocFrame takes an empty variant and constructs a Visual Basic array of the correct size and data type to hold one frame of data from the document. Normally it is not necessary to use this function since **GetFrame** and **GetStrip** will perform an **AllocFrame** if the buffer Variant supplied to them is not set up yet.

Example:

```
Dim objDoc As New DocFile
Dim vntFrame As Variant
objDoc.Open "circuit.spe"
objDoc.AllocFrame vntFrame
'
' later in the program...
'
objDoc.PutFrame 1, vntFrame
```

Function GetParam(Param As enum DM_CMD, result As Integer) As Any

Function SetParam(Param As enum DM_CMD, Value As Any) As Integer

Allows setting and getting document parameters (strips, frames, etc.). **SetParam** returns non-zero on error. Note that not all parameters may be set after a document is created. For instance, once created the **DM_XDIM** and **DM_YDIM** parameters can not be set, since this would require changing the size of the file on disk.

WinX/32 will try to convert the parameter to the correct type corresponding to the particular **DM_CMD**. If this cannot be done, Visual Basic reports a 'Type Mismatch' error. For more information see the discussion of Calling Syntax under "Things to Note".

Example:

```
Dim objDoc As New DocFile
ObjDoc.Open "circuit.spe"
Dim intXDim As Integer
intXDim = objDoc.GetParam(DM_XDIM) ' find the pixels per track
objDoc.SetParam DM_YLABEL, "Watts" ' set the y label to "Watts"
```

Function GetCalibration As CalibObj

Function SetCalibration(Calib As CalibObj) As Boolean

Allows setting and getting the X axis calibration information in the docfile. The information is passed as a CalibObj object, which is described below.

Example:

```
Dim objDoc As New Docfile
Dim objCal As CalibObj
objDoc.Open "Circuit.spe"
Set objCal = objDoc.GetCalibration
objCal.DisplayUnits = XW_WAVELENGTH ' Change display units only
bRes = objDoc.SetCalibration objCal
```

Sub Update()

Redraws all frames for this doc.

Function Save() As Boolean

Saves the data in the document with the current filename.

Function SaveAs(Name As String, nType As Long) As Boolean

Saves the data in the document with a new filename to a new file type. Use this function to convert a file to a new data type. The valid values for nType are located in docType enum.

Function Close() As Boolean

Closes all windows associated with this doc and closes the file. If the doc is unsaved, the user will be prompted to save it before closing. The return value is 1 if the document was open before the save, or zero otherwise.

DocFiles

The DocFiles object is a *collection* object used to refer to all of the open doc files. It can be used in a For...Each loop and DocFile objects can be accessed like array elements (i.e., objDocFiles(intIndex)).

Function GetAt(index As Integer) As DocFile

Returns the open DocFile, if any, corresponding to the zero-based index. The index value must be less than the number of open documents, or else the returned object will be empty.

Function Count() As Integer

Returns the number of open DocFile objects.

Sub CloseAll()

Closes all of the open DocFile objects.

CalibObj

The CalibObj object represents axis calibration. It is used by the SetCalib and GetCalib functions of the DocFile, DocWindow, and DocWindows objects. It has these properties:

Properties:

Name	Native Type	Description
DisplayUnit	XCALIBUNIT	Selected calibration unit for Display
Input Unit	XCALIBUNIT	Selected calibration unit for the CalibVal array
Order	BYTE (1-5)	Order of polynomial for calibration
PairCount	Integer	Number of valid pairs of PixelLoc and CalibVal
PixelLoc(10)	Double	Array of locations specified in pixels
CalibVal(10)	Double	Array of calibration values at PixelLoc positions.
PolyCoeffs(5)	Double	Array of polynomial coefficients for calibration
LaserPosition	Double	Laser wavenumber (valid for relative wavenumber units only)
Label	String	Calibration units label
UsageFlag	CALIB_USAGE	Flag to use calibration

Where XCALIBUNIT is one of:

XW_SYSTEM	Relative system units like channels, strips, and frames.
XW_DATA	Absolute units including offset and linear factor.
XW_WAVELENGTH	Polynomial calibration of wavelength

XW_ABSWAVENUM	Wavenumber units
XW_RELWAVENUM	Relative wavenumber units
XW_EVOLTS	Electron volts

The combination of DisplayUnits and InputUnits allows the programmer to supply calibration values (in the CalibVal array) in one unit, such as wavelength, while at the same time requiring the program to display the data (by default) in another unit, such as wavenumbers. The display units for a particular window may also be changed with the SetParam function of DocWindow.

Function Lambda(Pixel As Integer) As Double - returns the wavelength for a given pixel.

Function Pixel(Lambda As Double) As Integer - returns the pixel for a given wavelength.

DocInfo

The DocInfo object supplies information to the OpenNew method of DocFile.

Properties:

Name	Native Type	Description
X	integer	Number of pixels in each strip of the new document
Y	integer	Number of strips in each frame of the new document
Z	integer	Number of <u>new</u> frames in the new document
dataType	dataType	Data type for the new document
Name	String	The name for the new document
bShowWindow	Boolean	True if the document should be shown in a window
bAppend	Boolean	True if the document should be opened in append mode
FileType	docType	Type of file (SPE, TIF, etc.)

DocInfo defaults to bAppend = False, bShowWindow = True. If no name is supplied, the "Untitled" string is used.

Here is an example of using the DocInfo object with OpenNew to open a new data file:

```
Dim objDoc As New DocFile
Dim objInfo As New DocInfo ' supplies info about new document
objInfo.x = 200 ' 200 pixels per strip
objInfo.y = 2 ' 2 strips
objInfo.z = 1 ' 1 frame
objInfo.dataType = X_FLOAT ' floating point data
objInfo.bShowWindow = True ' display the file in a window
```



```

objInfo.bAppend = False ' overwrite; don't append to existing
                        file
objInfo.Name = "newdata" ' use this for the name
objInfo.FileType = dt_SPE
objDoc.OpenNew "", objInfo ' create the file

```

Display

DocFile

Refer to the **Experiment** section for the **DocFile** description (page 20).

DocWindow

This object represents the window in which a doc file is displayed. This object should not be created --- it should be retrieved through the DocFile or DocWindows objects.

Properties:

Name	Native Type	Description
Title	String	The title in the window (including the dimensions)
Visible	Boolean	The state of the shown/hidden window
ColorTable	WinXColorTable	The table that maps intensity to color.
Floating	Boolean	The state of attachment of this window to the application window.
Index	Long	The index of this window in the DocWindows collection.

```

Sub SetPosition(top As Integer, left As Integer, bottom
                As Integer, right As Integer)

```

```

Sub GetPosition(top As Integer, left As Integer, bottom
                As Integer, right As Integer)

```

Sets or gets the position of this window. Returns 0 on error.

```

Sub Update()

```

Redraws this frame.

```

Function GetParam(Param As enum DI_PARAM, result As
                  Integer) As Any

```

```

Function SetParam(Param As enum DI_PARAM, Value As Any)
                  As Integer

```

Allows setting and getting DocWindow parameters (axes, cross sections, etc.).

SetParam returns non-zero on error. It is recommended to call Update after calling SetParam. Not all parameters can be set after the window is created.

For more information, see GetParam function in the DocFile section, and also the discussion of Calling Syntax under "Things to Note".

Function GetDocument() As DocFile

Returns a pointer to an object of type DocFile, which you may use to call the DocFile interface functions for the document attached to this window.

Function NewWindow([winType As PICTURE_TYPE]) As DocWindow

Returns a pointer to a new DocWindow object representing a new window with the same document.

Example:

```
Dim objDoc As New DocFile
Dim objWin1 As New DocWindow
Dim objWin2 As New DocWindow
objDoc.Open("circuit.spe")
Set objWin1 = objDoc.GetWindow
Set objWin2 = objWin1.NewWindow(WINX_GRAPH)
```

Sub Next**Sub Prev****Sub NextSkip****Sub PrevSkip****Sub First****Sub Last**

Selects the strip or frame displayed by the window. For images, these subroutines control the displayed frame; for single curve graphs, they control the strip number; and for 3-D graphs, they control the strip number if the parameter DI_BPLOTFRAME is set to 0 or FALSE; otherwise, they control the frame number.

Sub Update()

Redraws this window.

Sub Close()

Closes this window.

Sub ZoomIn**Sub ZoomOut****Sub UnZoom**

ZoomIn zooms in on the current window. If the ROI rectangle is being displayed in the window, the zoom is done to the ROI. If the ROI rectangle is not visible, then the zoom is done to increase the zoom factor by 10%.

ZoomOut undoes the previous ZoomIn, if any. UnZoom undoes all the previous ZoomIns for this window.

Function GetROI(index As Integer) As ROIrect**Function SetROI(Rect As ROIrect) As Boolean**

Gets or sets the ROI rectangle for this window. Once the ROI rectangle is set, it can be displayed or hidden using the DI_BROIBOXON parameter in the SetParam function. The rectangle information is passed as an ROIrect object, which is described in that section.

Function GetCursor As CursorObj

Gets the CursorObj for this window. You can use the CursorObj to read or set the position of the cross cursor in this window. CursorObj functions and properties are listed in the CursorObj section.

Function GetCalibration As CalibObj**Function SetCalibration(Calib As CalibObj) As Boolean**

Allows setting and getting the X axis calibration information for the window. The information is passed as a CalibObj object, which is described below. These functions are the same as the GetCalibration and SetCalibration functions for DocFile, except that they operate on the calibration of a window instead (the window calibration can be different from the file calibration, and different windows of the same data file can have different calibrations.)

Function Activate() As Boolean

This function makes this window the active window. The function returns True if successful, otherwise it returns False.

Sub MoveCursor(X As Double, Y As Double)

This function sets to the cursor in the window to the coordinates (x,y).

Sub SetROIPos(top As Double, left As Double, bottom As Double, right As Double)

This function draws an ROI rectangle in the window whose top-left point is (left, top) and whose bottom-right point is (right, bottom).

Function GetHWnd() As Long

Returns the Win32 handle of this window. Can be used with SendMessage, ShowWindow, and other Windows API functions.

Sub ScrollTo(X As Long, Y As Long)

This function sets the horizontal scrollbar (if there is one) to the X position and the vertical scrollbar (if there is one) to the Y position.

DocWindows

The DocWindows object is a *collection* object that refers to all of the open DocWindows. It can be used in a For...Each loop and DocWindow objects can be accessed like array elements (i.e., objDocWindows(intIndex)). DocWindow properties referred to by SetParam or GetParam are the defaults for the WinX/32 program.

Function GetAt(index As Integer) As DocWindow

Returns the open DocWindow, if any, corresponding to the zero-based index. The index value must be less than the number of open DocWindows, or else the returned object will be empty.

Function Count() As Integer

Returns the number of open DocWindow objects.

Sub CloseAll()

Closes all of the open DocWindow objects.

Sub Tile(BOOL bHorizontal)

Sub Cascade();

Arranges all of the open DocWindow objects.

Function GetParam(Param As enum DI_PARAM, result As Integer) As Any

Function SetParam(Param As enum DI_PARAM, Value As Any) As Integer

Allows setting and getting DocWindow default parameters (axes, cross sections, etc.). SetParam returns non-zero on error. For more information see GetParam function in the DocFile section, and also the discussion of Calling Syntax under "Things to Note".

Function GetCalibration As CalibObj

Function SetCalibration(Calib As CalibObj) As Boolean

Allows setting and getting the default X axis calibration information for the program. The information is passed as a CalibObj object, which is described below. These functions are the same as the GetCalibration and SetCalibration functions for DocWindow, except that they operate on the default calibration of the program, rather than the calibration for a specific window

Function GetActive() As DocWindow

This function returns the active window if it's a data window, otherwise it returns Nothing.

CalibObj

Refer to the **Experiment** section for the **CalibObj** description (page 23).

ROIRect

Refer to the **Experiment** section for the **ROIRect** description (page 18).

CursorObj

The CursorObj object controls the cursor in a DocWindow. Call the GetCursor function of DocWindow to get the CursorObj for the window. The CursorObj properties are always current; that is, you don't have to do multiple GetCursor calls in order to get the most recent values, and when you set a property, the cursor in the window moves immediately. (This is not like the ROIRect object, for instance.)

Properties:

Name	Native Type	Description
XPos	Double	X coordinate (pixel or wavelength) of the cursor
YPos	Double	Y coordinate (strip) of the cursor
ZPos	Double	Z coordinate (frame) of the cursor
Intensity	Double	Intensity value of the pixel under the cursor

WinXColorTable

A WinXColorTable object contains an array of 256 colors (zero-based) that are used to create a bitmap for an image in a DocWindow. The way the data intensity is mapped to these colors depends on the current palette shape of the DocWindow (i.e., linear, logarithmic, inverted, etc.). This object should not be created -- it should be retrieved through the DocWindow object.

Properties:

Name	Native Type	Description
Val(<i>i</i>)	Long	The value of each color in the table at index <i>i</i> in &HBBGRR format
Red(<i>i</i>)	Integer	The red component (<i>RR</i>) of each color in the table at index <i>i</i> ranging from &H00 to &HFF
Green(<i>i</i>)	Integer	The green component (<i>GG</i>) of each color in the table at index <i>i</i> ranging from &H00 to &HFF
Blue(<i>i</i>)	Integer	The blue component (<i>BB</i>) of each color in the table at index <i>i</i> ranging from &H00 to &HFF

Sub Realize

Realizes the palette in the DocWindow.

Example:

```
Dim objDoc As New DocFile
Dim objWin As DocWindow
Dim objCT As WinXColorTable
Dim intIndex As Integer
objDoc.Open "circuit.spe"
Set objWin = objDoc.GetWindow
Set objCT = objWin.ColorTable
For intIndex = 120 To 140
    objCT.Red(intIndex) = 255 ' highlight mid-range data intensities
Next
objCT.Realize ' display the changes
```

Data Processing

Overview

Thus far, the documentation has been arranged on an object-by-object basis. Since the processing objects are related to each other in functionality, the documentation will now take a functional approach. Specific functionality will be introduced, followed by the properties and/or methods that define this functionality, followed by the objects that implement this functionality.

A multitude of objects allow processing of WinX/32 data files and they are all related to each other in very specific ways. Basically, these processes can be broken down into two types: Arithmetic/Logical and Non-Arithmetic/Logical. (Refer to the table below to see the type of each process.)

Arithmetic/Logical Processes are those processes that can be found under the WinX/32 Image/Spectra Math dialog. These processes use unary or binary operators (operators that take one or two inputs). If the operator is binary, the second input may represent a constant. Otherwise, all inputs represent data files. Since these operators are basic, the actual processes themselves need not be configured. An example of this type is adding two images together.

Non-Arithmetic/Logical Processes are the rest of the processes. These can be broken down into two subtypes, Convolution and Configurable. Convolution Processes apply a convolution kernel to a single input data file. Since the process itself implicitly defines the application of the kernel, the actual processes themselves need not be configured. An example of this type is dilating an image. Configurable Processes do not use an arithmetic/logical operator or a convolution kernel. They are processes whose operators are complex enough that they must be configured. These operators also are only applied on a single data file. An example of this type is creating a histogram for an image.

Arithmetic/Logical Objects	Non-Arithmetic/Logical Objects	
	Convolution	Configurable
ProcMath	ProcDilate	ProcBinSkip
	ProcErode	ProcClipThresh
	ProcFilter	ProcCrossSect
	ProcSobel	ProcHistogram
		ProcLUT
		ProcOrient

Configuring the Input

All data processing objects require at least one input to operate on. The objects that implement this functionality act as the input data. They can be defined as a data file or a constant.

Properties:

Name	Native Type	Description
Value	Variant	A numerical value (representing a constant) or a valid path to a data file.
dataType	DataType	The data type of the input file if the Value property contains a path.
Xstart	Long	The starting column of data to be used if the Value property contains a path.
Xend	Long	The ending column of data to be used if the Value property contains a path.
Ystart	Long	The starting strip of data to be used if the Value property contains a path.
Yend	Long	The ending strip of data to be used if the Value property contains a path.
Zstart	Long	The starting frame of data to be used if the Value property contains a path.
Zend	Long	The ending frame of data to be used if the Value property contains a path.
Xcount	Long	The number of pixels per strip to be used if the Value property contains a path.
Ycount	Long	The number of strips to be used if the Value property contains a path.
Zcount	Long	The number of frames to be used if the Value property contains a path.

Methods:

Sub GetFromDoc(pDocFile As DocFile)

Initializes this input using an existing DocFile object.

Implemented by:

ProcInput

Example:

See the example located in **Running an Arithmetic/Logical Process.**

Configuring the Output

All data processing objects produce one output as a result of the process. The objects that implement this functionality define how the output data is presented.

Methods:

```
Function GetProcParam(Param As PRC_PARAM, optional
                    pRes As PROCERR) As Any
Function SetProcParam(Param As PRC_PARAM, vSetVal As
                    Any) As PROCERR
```

Allows setting and getting general parameters for the output data file, such as displaying the output or enabling file increment. Calling these through the ProcessManager object reads/modifies the default for all processes. Otherwise, only the calling process is affected.

WinX/32 will try to convert the parameter to the correct type corresponding to the particular PRC_PARAM. If this cannot be done, Visual Basic reports a 'Type Mismatch' error.

Implemented by:

ProcessManager, ProcBinSkip, ProcClipThresh, ProcCrossSect, ProcDilate, ProcErode, ProcFilter, ProcHistogram, ProcLUT, ProcMath, ProcOrient, ProcSobel

Example:

See the example located in **Running an Arithmetic/Logical Process**.

Configuring the Convolution Kernel

Non-Arithmetic/Logical Processes that contain a convolution kernel must configure the kernel before applying the process. The kernel resides in the specific process object itself. It is a two dimensional array (zero-based) of values. Elements may be changed individually by accessing the property below. To change the kernel dimensions or to change all elements at once, use the function below. The specific object implicitly defines the application of the kernel, so that aspect need not be configured.

Properties:

Name	Native Type	Description
Kernel(x, y)	Double	The kernel element at indices (x, y).

Methods:

```
Sub SetKernel(pvKernel)
```

Replaces the kernel with pvKernel. The variable pvKernel is usually a two-dimensional array of type Double or Variant. It may also be single dimension array. If this is the case, the process object assumes the array size is a perfect square x^2 such that it will take on the dimensions x by x .

Implemented by:

ProcDilate, ProcErode, ProcFilter, ProcSobel

Example:

See the example located in **Running a Non-Arithmetic/Logical Process**.

Configuring the Process

Configurable Processes must be configured before applying the process. Convolution Processes may use this functionality to read back the current dimensions of the kernel.

Methods:

```
Function GetParam(Param As PIIP_SetGetParam,
                 optional pRes As PROCERR) As Any
Function SetParam(Param As PIIP_SetGetParam, vSetVal
                 As Any) As PROCERR
```

Allows setting and getting parameters that define a Configurable Process, such as setting a low clipping value for the ProcClipThresh object. If called through a Convolution Process the current kernel dimension can be returned.

WinX/32 will try to convert the parameter to the correct type corresponding to the particular PIIP_SetGetParam. If this cannot be done, Visual Basic reports a 'Type Mismatch' error.

Implemented by:

ProcBinSkip, ProcClipThresh, ProcCrossSect, ProcDilate, ProcErode, ProcFilter, ProcHistogram, ProcLUT, ProcOrient, ProcSobel

Example:

See the example located in **Running a Non-Arithmetic/Logical Process**.

Running an Arithmetic/Logical Process

The input (or inputs if the operator is binary) must be configured beforehand. If the output is not configured, the defaults for the process will be used.

Methods:

```
Function Run(pInputA As ProcInput, pInputB As
            ProcInput, eOp As
            PIDP_IMAGEMATHOPERATION, szOutput As
            String, eOutType As dataType, optional
            pResult As PROCERR) As DocFile
```

Runs an arithmetic/logical process. The inputs are set with pInputA and pInputB (if the operator is binary). Otherwise, just use pInputA. The parameter eOp determines the arithmetic or logical operation. The output filename and data type are determined by szOutput and eOutType, respectively. After the function call, if no error is set in pResult, the output data file will be returned.

Implemented by:

ProcMath

Example:

```
' configure the first input
Dim objDoc As New DocFile
objDoc.Open "circuit.spe"
Dim objInputA As New ProcInput
Call objInputA.GetFromDoc(objDoc) ' set Input A to circuit.spe
```

```

' configure the second input
Dim objInputB As New ProcInput
ObjInputB.Value = 10          ' set Input B to the constant 10

' configure the output
Dim objArith As New ProcMath
objArith.SetProcParam PRC_DISPLAYOUTPUT, False ' don't display output

' run the process
Dim objOutput As DocFile
Set objOutput = objArith.Run(objInputA, objInputB,
IMAGEMATH_ADDITION, "c.spe", X_FLOAT)
objOutput.Save
objOutput.Close

```

Running a Non-Arithmetic/Logical Process

The input must be configured beforehand. If it's a Convolution Process, the kernel must be configured process itself already defines it. If a Configurable Process is used, the process itself must be configured appropriately. If the output is not configured, the defaults for the process will be used.

Methods:

Function Run(pInputA As ProcInput, szOutput As String, eOutType As dataType, optional pResult As PROCERR) As DocFile

Runs a non-arithmetic/logical process. The input is set with pInputA. The output filename and data type are determined by szOutput and eOutType, respectively. After the function call, if no error is set in pResult, the output data file will be returned.

Implemented by:

ProcBinSkip, ProcClipThresh, ProcCrossSect, ProcDilate, ProcErode, ProcFilter, ProcHistogram, ProcLUT, ProcOrient, ProcSobel

Example:

```

' configure the first input
Dim objDoc As New DocFile
objDoc.Open "circuit.spe"
Dim objInput As New ProcInput
Call objInput.GetFromDoc(objDoc)      ' set input to circuit.spe

' configure the first output
Dim objMorph As New ProcDilate
objMorph.SetProcParam PRC_DISPLAYOUTPUT, true ' display output

' run the first process
Dim objOutput As DocFile
Set objOutput = morph.Run(objInput, "dilated.spe", X_FLOAT)
objOutput.Save

' configure the second input
Call objInput.GetFromDoc(objOutput)   ' set input to dilated.spe

' configure the second output
Dim objHist As New ProcHistogram
objHist.SetParam PRC_DISPLAYOUTPUT, false ' don't display output

```

```

' configure the second process
objHist.SetParam HISTOGRAM_OPERATIONS, HISTOGRAM_NORMAL
objHist.SetParam HISTOGRAM_LOWVAL, 1
objHist.SetParam HISTOGRAM_HIGHVAL, 4096
objHist.SetParam HISTOGRAM_GROUP, 1

' run the second process
Set objOutput = objHist.Run(objInput, "hist.spe", X_FLOAT)
objOutput.Save
objOutput.Close

```

Utilities

WinX32App

This object is used to control general functionality of the application such as sizing the window, displaying a message box, and hiding the applications.

Properties:

Name	Native Type	Description
Flavor	String	The WinX/32 program that is running. For instance, WinView/32 returns the string "WinView/32", and WinSpec/32 returns "WinSpec/32".
Version	String	The version of WinView or WinSpec that is running. For instance, version 2.5 of WinSpec returns a string like "2.5.0.0".
rectBottom	Long	Bottom coordinate of the main window.
rectLeft	Long	Left coordinate of the main window.
rectRight	Long	Right coordinate of the main window.
rectTop	Long	Top coordinate of the main window.
Visible	Boolean	The state of the shown/hidden main window.

Sub ShowDemoBox(Text As String)

Displays the text in Text via a simple message box.

Example:

```

Dim objWinX As New WinX32App
objWinX.ShowDemoBox "This is my text"

```

Function GetAppWnd() As Long

Returns the Win32 window handle of the WinX/32 application's main window. Can be used with SendMessage, ShowWindow, and other Windows API functions.

Example:

```
Public Declare Function ShowWindow Lib "user32" (ByVal hwnd As
    Long, ByVal nCmdShow As Long) As Long
Public Const SW_SHOWMAXIMIZED = 3
Dim objWinX As New WinX32App
Dim lngWnd As Long
lngWnd = objWinX.GetAppWnd
' Maximize the WinX/32 application window
Dim lngResult As Long
lngResult = ShowWindow(lngWnd, SW_SHOWMAXIMIZED)
```

Function GetAppRect(top As Long, left As Long, bottom As Long, right As Long) As Boolean

Retrieves the coordinates of the WinX/32 application's main window. Returns non-zero on error.

Function CountOpenDocs() As Long

Returns the number of documents open.

Function CloseOpenDocs(cDocs As Short) As Boolean

Closes all open document windows. If cDocs is not 0, it will have the number of windows closed on return. Returns 0 if no error.

Function InfoBox(Boolean bOn) As Boolean

Opens or closes the info box. Returns TRUE if the box was open before the call.

Example:

```
Dim objWinX As New WinX32App
Dim blnStatus As Boolean
blnStatus = objWinX.InfoBox (True)
```

Function StatusBar(Boolean bOn) As Boolean

Opens or closes the Status Bar. Returns non-zero if the bar was open before the call.

Sub Hide(Boolean bhide)

This function can hide or show the WinX/32 application and all child windows. If the user did not start WinView or WinSpec before running your program, the first time your program creates a COM object (such as a WinX32App or a DocFile) then WinView or WinSpec will start automatically. When the program detects that it is starting as a result of a COM object call, it will wait an additional 3 seconds before displaying its splash screen or main window. This is to give your program enough time to call the Hide function, which will prevent the window from ever being displayed. You can also call the Hide function after the WinX/32 application is visible. If the Boolean parameter is true, the application will not be visible; if False, it will be visible.

Function StatusBarMsg(Index As Integer, Color As Integer, Text As String) As Boolean

Use this function to display a message in the status bar at the bottom of the WinX/32 application. The status bar is divided into 8 panes. The leftmost pane corresponds to an Index of zero. Colors are defined as:

Black	1	Yellow	7
Red	2	Dark Gray	8
Green	3	Gray	9
Blue	4	Light Gray	10
Cyan	5	White	11
Pink	6		

Note that the WinX/32 application uses the status bar panes for its own purposes, and that the text you place onto the pane may be subsequently overwritten by the application.

Sub ShowProgress(Percent As Integer)

Use this function to control the progress indicator, which appears on the second pane of the status bar. Normally, this indicator is used to indicate the progress of some process the application is running. The value of Percent should be between zero and 100. A value of 0 makes the progress indicator disappear. Any other value will cause it to appear and display a bar to indicate the value of Percent. Since the application also uses the progress indicator, your setting may be overwritten if data collection or some other process is running.

Function Quit() As Boolean

Use this function to attempt to close WinX/32. This function acts as if the user clicked on the system close button ([x] in the top right corner of the window). In certain situations (such as the program in data acquisition mode), the user is prompted as to whether or not close the program. If the user declines to close, WinX/32 will remain open. The return value is True if WinX/32 was closed successfully, otherwise the return value is False.

Function ShowFileOpen(InName As String) As String

Function ShowFileSaveAs(InName As String) As String

Use these functions to show WinX/32's File Open/Save-As Dialog. The dialog is initialized with InName. If Open/Save was pressed, the function returns the name of the file selected (including the path). If Cancel was pressed, the function returns an empty String.

Sub LoadFactoryDefaults(INIFile As String)

If passed the path to a valid WinX/32 INI file, it will reload these hardware defaults. If passed an empty string, it will reload the hardware defaults stored in the controller's NVRAM.

Sub SaveAll

Saves all documents that are open and have been modified.

PrintWindow

This object represents a text window. The PrintWindow appears when the user program first calls the Print or Clear methods of the object. Besides being created, this object can be retrieved from the PrintWindows object.

Properties:

Name	Native Type	Description
Title	String	The title in the window

Function OpenFile(Name As String) As Boolean

Opens a print window file from disk. The file must already exist. The function returns False if not successful.

Function Print(xpos As Integer, line As Integer, color As Long, ParamArray Pstuff() As Variant) As Boolean

Print to the print window at a character position xpos on a given line using a given color. Any number of items of any type can be printed in one call. The color parameter takes a long hex number in the format &HBBGRR, where RR (amount of red), GG (amount of green), and BB (amount of blue) can be in the range of &H00 to &HFF. Returns True on success.

Note: Due to the variable number of parameters in ParamArray, the return value must be assigned. Otherwise, Visual Basic will not recognize the function.

Example:

```
Dim objPrint As New PrintWindow
Dim blnStatus As Boolean
blnStatus = objPrint.Print(0, 1, 0, "This is", text")
blnStatus = objPrint.Print(5, 2, &HFF, "This is", " more", " text")
blnStatus = objPrint.Print(10, 3, &HFF00, "This is", " line", 3)
```

Function SaveAs(Name As String) As Boolean

Saves a print window to a file on disk. If the file already exists, it is overwritten without warning. The function returns False if not successful. The file name should include an extension of “.PRN” in order for the file to be recognized by WinX/32 and Windows as a WinX/32 print window file.

Function SetFont(FaceName As String, PointSize As Long) As Boolean

This function sets the font for subsequently printed text in the print window. FaceName should be the name of an existing font on the system; if the font does not exist, Windows will try to find as close a match as possible. PointSize gives the size of the font in points. The function returns False if not successful

Function SetPosition(top As Integer, left As Integer) As Boolean

Set the position of the print window with respect to the screen (can exceed the boundaries of the app window). Returns False on error.

**Function SetSize(width As Integer, height As Integer)
As Boolean**

Set the position of the print window with respect to the screen (can exceed the boundaries of the app window). Returns 0 on error.

Sub SysCmd(Cmd As Long, Param As Long) As Long

This is a function for more advanced programmers. It allows a VB program to send a WM_SYSCOMMAND message to the print window. The WM_SYSCOMMAND message takes a parameter which causes the window to perform one of the actions in the “system menu”, the menu which appears when you click the box in the upper left corner of the window. The actions which a PrintWindow can perform in response to a WM_SYSCOMMAND message include, but are not limited to, those in the following table:

Cmd	Value	Action performed
SC_SIZE	&HF000	Starts the keyboard-driven sizing of a window
SC_MOVE	&HF010	Starts the keyboard-driven moving of a window
SC_MINIMIZE	&HF020	Minimizes the window
SC_MAXIMIZE	&HF030	Maximizes the window
SC_NEXTWINDOW	&HF040	Selects the next open window
SC_PREVWINDOW	&HF050	Selects the previous open window
SC_CLOSE	&HF060	Closes the window
SC_VSCROLL	&HF070	Causes the window to scroll vertically
SC_HSCROLL	&HF080	Causes the window to scroll horizontally
SC_ARRANGE	&HF110	Arranges the icons of minimized windows
SC_RESTORE	&HF120	Restores the window from the minimized or maximized states

The value for Param will depend on the value for Cmd; for all of the Cmds listed, it can be zero.

SysCmd returns a value which depends on the Cmd; for all of the Cmds listed, it will be zero if successful.

Sub Clear()

Clears the contents of the print window.

Sub Close()

Closes the print window.

Sub Open()

This function opens the print window.

Function GetHwnd() As Long

Returns the Win32 handle of this window. Can be used with SendMessage, ShowWindow, and other Windows API functions.

Function GetRichEditWnd() As Long

This function is for advanced programmers only. It returns the handle of the window's RichEditCtrl. The RichEditCtrl is a Windows object; PrintWindow objects use it to display text. By using the handle of the control, the programmer can send messages directly to the control in order to achieve text effects not supported directly by the PrintWindow.

PrintWindows

The PrintWindows object is a collection object, which refers to all of the open PrintWindow objects. It can be used in a For...Each loop and PrintWindow objects can be accessed like array elements (i.e., objPrintWindows(intIndex)).

Function GetAt(index As Integer) As PrintWindow

Returns the open PrintWindow, if any, corresponding to the zero-based index. The index value must be less than the number of open PrintWindow objects, or else the returned object will be empty.

Function Count() As Integer

Returns the number of open PrintWindow objects.

Sub CloseAll()

Closes all of the open PrintWindow objects.

ArrayConverter

An ArrayConverter object is a utility object used to manage array data types. It allows in-place data type conversions or copy conversions. The data types supported are: **Byte**, **Integer**, **Long**, **Single**, **Double** and **Variant**. All data types may be converted to larger data types. All data types except **Variant** can be truncated to smaller data types. To use these functions, the user program must create an ArrayConverter object and call the functions through it.

Properties:

Name	Native Type	Description
UnsignedInput	Boolean	Treat the input array as unsigned data type (e.g., a Integer data type would be treated as a 16-bit unsigned number).

Sub Convert(Array As Any, Type As VbVarType)

Performs an in-place conversion on Array to the data type Type. Array is a Variant containing an array of the supported data types. Type is an enum coming from the Visual Basic for Applications type library (included in standard Visual Basic projects).

Example:

```
' get an array of data
Dim objDoc As New DocFile
objDoc.Open "circuit.spe"
Dim vntFrame
objDoc.GetFrame 1, vntFrame

' display original data type
MsgBox TypeName(vntFrame)

' convert in-place
Dim objAC As New ArrayConverter
objAC.Convert vntFrame, vbDouble

' display new data type
MsgBox TypeName(vntFrame)
```

**Function CopyConvert(Array As Any, Type As VbVarType)
As Any**

Converts Array to the data type Type and returns the result. Array retains its original data type. Array is a Variant containing an array of the supported data types. Type is an enum coming from the Visual Basic for Applications type library (included in standard Visual Basic projects).

Example:

```
' get an array of data
Dim objDoc As New DocFile
objDoc.Open "circuit.spe"
Dim vntFrame
objDoc.GetFrame 1, vntFrame

' display original data type
MsgBox TypeName(vntFrame)

' convert and store the result
Dim vntFrameCopy
Dim objAC As New ArrayConverter
vntFrameCopy = objAC.CopyConvert(vntFrame, vbDouble)

' display original and the new data types
MsgBox TypeName(vntFrame)
MsgBox TypeName(vntFrameCopy)
```

Spectrograph

SpectroObj

A SpectroObj object represents a spectrograph. This object should not be created -- you must retrieve this object from the SpectroObjMgr.

There are additional type libraries associated with the SpectroObj. These type libraries contain relevant enumerated types that may be used with the functions below. Use the type library associated with the manufacturer DLL of the spectrograph that ships with WinSpec/32.

Function GetEnumerator(Param As SPT_CMD, index As Integer, optional pResult As Any) As Long

Returns an enum parameter, which can then be passed to GetEnumerator. Param is the same SPT_CMD that will be passed to GetEnumerator. If more than one enum parameter is possible for a given SPT_CMD (determined by Count in the ValidRange object pertaining to that SPT_CMD), use the zero-based index to select a particular enum parameter. Success is determined by a non-zero value in pResult.

Example:

See the example located under the GetEnumerator function.

Function GetEnumeratorString(Param As SPT_CMD, EnumVal As Integer, optional pResult As Any) As String

Returns a string related in some way to the SPT_CMD Param (usually it is dialog text). EnumVal must be obtained from GetEnumerator using the same SPT_CMD. For SPT_CMD parameters that contain more than one string, multiple enum parameters must be used. See GetEnumerator for more details on how to do this. Success is determined by a non-zero value in pResult.

Example:

```
' use the current spectrograph
Dim objSpecs As New SpectroObjMgr
Dim objSpec As SpectroObj
Set objSpec = objSpecs.Current

' get information about the mirror location param
Dim objValid As ValidRange
Set objValid = objSpec.IsAvail(SPT_INST_MIRROR_LOCATION)

' get and display each mirror location string
Dim intIndex As Integer
Dim intParam As Integer
For intIndex = 0 To objValid.Count - 1
    intParam = objSpec.GetEnumerator(SPT_INST_MIRROR_LOCATION,
                                     intIndex)
    MsgBox objSpec.GetEnumeratorString(SPT_INST_MIRROR_LOCATION,
                                       intParam)
Next intIndex
```

Function GetParam(Param As SPT_CMD, optional index As Integer, optional pResult As Any) As Any

Function SetParam(Param As SPT_CMD, pSetVal As Any, optional index As Integer) As Integer

Allows setting and getting parameters for the spectrograph object. The optional index (zero-based) is only used for SPT_CMD parameters that require it. Non-zero (in pResult for GetParam and returned by SetParam) indicates success.

Note that these functions relate to the software object, not the actual hardware. The physical spectrograph is only programmed by the appropriate Process call.

WinX/32 will try to convert the parameter to the correct type corresponding to the particular SPT_CMD. If this cannot be done, Visual Basic reports a 'Type Mismatch' error.

Example:

See the example located under the **Move** function.

Function IsAvail(Param As SPT_CMD) As ValidRange

Returns a ValidRange object that corresponds to the particular SPT_CMD. Refer to the section on the ValidRange object for more details.

Example:

See the example located under the **GetEnumString** function.

Sub Load(FileName As String)

Configures the spectrograph object with all of the settings in an INI file pointed to by FileName.

Function Move() As Integer

Moves the physical spectrograph to the position currently set in the object.

Example:

```
' create a new spectrograph (note use of ActonSpec TypeLib)
Dim objSpecs As New SpectroObjMgr
Dim objSpec As SpectroObj
Set objSpec = objSpecs.Add(ACTON_TYPE,
ACTONSPECLib.ACTON_SP275)
                                ' initialize the spectrograph

objSpec.SetParam SPT_PORT_TYPE, 6           ' set to use COM port
objSpec.SetParam SPT_PORT_ADDRESS, 1       ' set to port 1
objSpec.Process SPTP_CREATE_PORT           ' connect to the port
objSpec.Process SPTP_INITIALIZE            ' general init
objSpec.Process SPTP_INST_LOADCONFIGURATION ' load from inst

' determine and set new pos based on current pos
Dim dblPos As Double
dblPos = objSpec.GetParam(SPT_CUR_POSITION)
objSpec.SetParam SPT_NEW_POSITION, dblPos + 10
' move the instrument

objSpec.Move                               ' destroy the spectrograph
objSpecs.Remove objSpecs.Count ' use the index of the last one created
```

Function Process(Param As SPT_PROCESS) As Integer

Downloads applicable settings pertaining to the selected SPT_PROCESS to the physical spectrograph. A non-zero value is returned on success.

Example:

See the example located under the **Move** function.

Sub Save(FileName As String)

Saves all of the current settings in the spectrograph object to an INI file specified by FileName.

SpectroObjMgr

The SpectroObjMgr object is a *collection* object used to refer to all of the spectrographs. Use this object to create, remove or get access to the SpectroObj objects.

Properties:

Name	Native Type	Description
Count	Integer	The number of spectrographs available
Current	SpectroObj	The current spectrographs
Item(<i>i</i>)	SpectroObj	Access a spectrographs by collection index (one-based)

Function Add(DLLType As SPECT_DLL_TYPES, SpecID As Integer) As SpectroObj

Creates a new spectrograph object based on the manufacturer DLL and the spectrograph type. If successful, returns the created SpectroObj.

Note that the SpecID is an enumerated type from one of the spectrograph type libraries that ship with WinSpec/32.

Example:

See the example located under the **Move** function in **SpectroObj**

Function GetAt(index As Integer) As SpectroObj

Returns the SpectroObj at the given collection index (one-based).

Sub Remove(iVal As Any)

Removes the SpectroObj at the given collection index (one-based).

Example:

See the example located under the **Move** function in **SpectroObj**.

ValidRange

Refer to the **Experiment** section for the **ValidRange** description (page 19).

Pulser

PITG

A PITG object represents a pulser. It can be any one of the pulsers supported by WinX/32. If the user program does not create a specific pulser (using the **Create** function below) this object will represent WinX/32's currently selected pulser.

Properties:

Name	Native Type	Description
DLLVersion	Variant	A Long value representing the DLL version (v2.5.1.4 would be represented as 2050104)
DLLVersionString	String	A String representing the DLL version (v2.5.1.4 would be represented as "2.5.1.4")

Sub Create(TGtype As PITG_TYPES)

Creates the specified pulser type using the default port for that type. If **Create** is not called, the object represents WinX/32's current pulser (if there is one). Due to the intrinsic coupling between the PTG and the camera controller, it is highly recommended to avoid creating a PTG object and use WinX/32's current selected PTG instead.

Example:

See the example located under the **Start** function.

Function GetEnumParam(paramID As TG_CMD, index As Integer, optional pResult As Any) As Long

Returns an enum parameter, which can then be passed to GetEnumString. The argument paramID is the same TG_CMD that will be passed to GetEnumString. If more than one enum parameter is possible for a given TG_CMD (determined by Count in the ValidRange object pertaining to that TG_CMD), use the zero-based index to select a particular enum parameter. Success is determined by a non-zero value in pResult.

Example:

See the example located under the **GetEnumString** function.

Function GetEnumString(paramID As TG_CMD, EnumParam As Long, optional pResult As Any) As String

Returns a string related in some way to the TG_CMD paramID (usually it is dialog text). EnumParam must be obtained from GetEnumParam using the same TG_CMD. For TG_CMD parameters that contain more than one string, multiple enum parameters must be used. See GetEnumParam for more details on how to do this. Success is determined by a non-zero value in pResult.

Example:

```

' use the current pulser
Dim objPulser As New PITG

' get information about the delay width time units
Dim objValid As ValidRange
Set objValid = objPulser.IsAvail(TGC_PULSE_DELAY_UNITS)

' get and display each time unit
Dim intIndex As Integer
Dim intParam As Integer
For intIndex = 0 To objValid.Count - 1
    intParam = objPulser.GetEnumParam(TGC_PULSE_DELAY_UNITS,
                                     intIndex)
    MsgBox objPulser.GetEnumString(TGC_PULSE_DELAY_UNITS,
                                   intParam)
Next intIndex

```

Function GetParam(Param As TG_CMD, optional pResult As Any) As Any

Function SetParam(Param As TG_CMD, SetVal As Any) As Integer

Allows setting and getting parameters for the pulser object. Non-zero (in pResult for GetParam and returned by SetParam) indicates success.

Note that these functions relate to the software object, not the actual hardware. The physical pulser is only programmed by the appropriate Process call. The only two exceptions are TGC_TEMPERATURE and TGC_TRIGGER_COUNT, which read the values in the hardware each time they are gotten.

WinX/32 will try to convert the parameter to the correct type corresponding to the particular TG_CMD. If this cannot be done, Visual Basic reports a 'Type Mismatch' error.

Example:

See the example located under the **Start** function.

Function IsAvail(paramID As TG_CMD) As ValidRange

Returns a ValidRange object that corresponds to the particular TG_CMD. Refer to the section on the ValidRange object for more details.

Example:

See the example located under the **GetEnumString** function.

Function Process(ProcessID As TG_PROCESS) As Integer

Downloads applicable settings pertaining to the selected TG_PROCESS to the physical pulser. A non-zero value is returned on success.

Example:

See the example located under the **Start** function.

Function Reset() As Integer

Programs the pulser to a known default state. The PG200 will receive the Reset command. DG535 has outputs A, B, CD, and D set to 4V, 0V offset, high Z termination and single shot. The PTG is stopped, reloaded with the current timing pattern, and its trigger counter is reset. A non-zero return value designates success.

Function SetGateLive(GateWidth As Double GateDelay As Double) As Integer

Changes the pulser's gate width and gate delay on the fly. The time units used for the parameters are the ones previously set with TGC_SEQ_ARRAY_UNITS. A return value of non-zero designates success.

Function Start() As Integer

Starts the pulser. The PG200 will receive the Start sequence command. DG535 receives a single shot trigger if the timing mode is set to single shot. The PTG's main trigger is enabled. A non-zero return value designates success.

Example:

```
' create a PG200 object
Dim objPulser As New PITG
objPulser.Create PI_PG200

' prepare the pulser for a sequence
Const cintShots As Integer = 10
objPulser.SetParam TGC_SEQ_NUM_SHOTS, cintShots
objPulser.SetParam TGC_SEQ_ARRAY_UNITS, 1 ' microseconds

' designate a user-generated sequence
objPulser.SetParam TGC_SEQ_INC_TYPE, INC_CUSTOM

' generate a sequence
Dim dblDelays(1 To cintShots) As Double
Dim dblWidths(1 To cintShots) As Double
Dim intIndex As Integer
For intIndex = 1 To cintShots
    dblWidths(intIndex) = 50
    dblDelays(intIndex) = intIndex * 10
Next intIndex
objPulser.SetParam TGC_SEQ_WIDTH_ARRAY_HANDLE, dblWidths
objPulser.SetParam TGC_SEQ_DELAY_ARRAY_HANDLE, dblDelays

' download the sequence to the pulser now
' NOTE: if this pulser represented WinX's pulser, this step
' would be unnecessary as the download is taken care of when
' the experiment runs
objPulser.Process TGP_PRELOAD_SEQUENCE

' set up the experiment
Dim objExp As New ExpSetup
objExp.SetParam EXP_SEQUENTS, cintShots

' run the experiment
' NOTE: if this pulser represented WinX's pulser, this step
' would be unnecessary as the pulser is started when the
' experiment runs
objPulser.Start ' always start the pulser first
objExp.Start2
```

Function Stop() As Integer

Stops the pulser. The PG200 will receive the Stop sequence command. DG535 does nothing. The PTG's main trigger is disabled. A non-zero return value designates success.

ValidRange

Refer to the **Experiment** section for the **ValidRange** description (page 19).

WinX32 Automation Enumerations

Experiment

EXP_CMD (ExpSetup Parameters)

U = Upgrade, S = Special

Parameter Name	Set	Get	Avail	Description	Hardware Support												
					ST130	ST138	ST133	ST133 5 MHz	MicroMax	SpectroMax	MicroView	PentaMax	Mini-Cam	MSP-100	VICCD		
EXP_5X_GAIN_ENABLE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	5X Gain Enable for 1MHz-only ADCs.			<input checked="" type="checkbox"/>										
EXP_ABSORBFILENAME	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Name of the experiment absorbance source file.		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
EXP_ABSORBSTRIP	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Absorbance strip number.		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
EXP_ACCESSNAME	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Name of the experiment access pattern save file.	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
EXP_ACCUMS	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Number of accumulations to perform.	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
EXP_ACQBACK_SHUTTER	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Acquire Background Shutter Mode.	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
EXP_ACQFLAT_SHUTTER	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Acquire Flat Field Shutter Mode.	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
EXP_ACTUAL_TEMP	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Detector actual temperature.			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>							

Parameter Name	Set	Get	Avail	Description	Hardware Support											
					ST130	ST138	ST133	ST133 5 MHz	MicroMax	SpectroMax	MicroView	PentaMax	Mini-Cam	MSP-100	VICCD	
EXP_LOGIC_OUTPUT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Output of BNC in the back of the controller: For ST133 V3, selects Not Scan (0) or Shutter (1). PentaMax V5 can also be Not Ready (2), Logic 0 (3), Cleaning (4), Not Frame Transfer Image Shift (5), and Logic 1 (7).			✓ V3					✓ V5				
EXP_MAXTHRESHOLDVALUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Maximum Threshold Value	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EXP_MIN_BLOCK	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Minimum block size to bin 'skips' close to the valid data		✓	✓	✓	✓	✓		✓				
EXP_MINTHRESHOLDVALUE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Minimum Threshold Value	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EXP_NEWWINDOW	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Use new window for every run	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EXP_NOP_ADC	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NOP count to slow up transfer speed. Some computers need this to use an ISA interface card with 1MHz controllers.		✓	✓	✓	✓	✓	✓					
EXP_NUM_MIN_BLOCK	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Minimum number of skip blocks to use.		✓	✓	✓	✓	✓		✓				

Parameter Name	Set	Get	Avail	Description	Hardware Support										
					ST130	ST138	ST133	ST133 5 MHz	MicroMax	SpectroMax	MicroView	PentaMax	Mini-Cam	MSP-100	VICCD
EXP_RUNNING_EXPERIMENT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	1 = Entire Experiment Running, 0 = Not Running	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EXP_SAVEROPT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Screen Saver/Blanker options	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EXP_SEQUENTS	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Number of sequential frames	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EXP_SETUPNAME	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Name of the experiment setup save file	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EXP_SHT_GATE_MODE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Shutter/gating mode for PI-Max systems: Gated (1), Shutter (2), and Safe Mode (0).			✓								
EXP_SHT_PREOPEN	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Mode for shutter: Pre-opened (1, usually in external sync mode), or opened and closed each shot (0).	✓	✓	✓	✓	✓	✓	✓	✓			
EXP_SHUTTER_COMP_TIME_MS	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	This returns the shutter compensation time in ms.			✓	✓	✓	✓	✓	✓			

Parameter Name	Set	Get	Avail	Description	Hardware Support											
					ST130	ST138	ST133	ST133 5 MHz	MicroMax	SpectroMax	MicroView	PentaMax	Mini-Cam	MSP-100	VICCD	
EXP_USEROI	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Use Full chip / ROI flag												
EXP_VIDEO_TYPE	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Video type: NTSC (1) or Pal (2). Also requires hardware jumpers to be installed.			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>				
EXP_VTCLK	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Vertical Clock register (1-16).	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>							
EXP_XDIM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Get data X dimension; total size of all ROIs	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
EXP_XDIMDET	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Get detector X dimension	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
EXP_YDIM	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Get data Y dimension; total size of all ROIs	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
EXP_YDIMDET	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Get detector Y dimension	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
EXP_YTNAME	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Name of the experiment YT data file	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

DM_CMD (DocFile Parameters)

Note: These functions only affect values in the data file, not in the controller.

Parameter Name	Description/Native Type
DM_ACCUMS	Accumulations
DM_ADCBITADJUST	ADC bit adjust
DM_ADCRATE	ADC rate

Parameter Name	Description/Native Type
DM_ADCRESOLUTION	ADC resolution
DM_ADCTYPE	ADC type (FAST or SLOW)
DM_ASYNEN	Set asynchronous or synchronous (1 async, 0 sync)
DM_BACKGROUNDAPPLIED	Background subtraction flag
DM_BACKSUBNAME	Background subtraction file name
DM_BLEMISHAPPLIED	Blemish flag
DM_BLEMISHFILE	Blemish file name
DM_CLEANS	Cleans at start of acquisition
DM_CLKSPD	Clock speed, (kinetics/frame transfer)
DM_CLKSPD_us	Vertical clock speed in micro-seconds
DM_CONTOLLERNUM	Controller number if multiple controller system
DM_CONTOLLERTYPE	3 = ST130, 5 = ST138, 6 = PentaMAX, 7 = MicroMAX,
DM_COSMICAPPLIED	Cosmic ray removal flag
DM_COSMICTHRESHOLD	Cosmic ray removal threshold
DM_COSMICTYPE	Type of cosmic array if applied
DM_DATATYPE	Data type
DM_DETTYPE	Type of detector (i.e., Kodak 1400)
DM_DLABEL	Data label string
DM_DOCTYPE	Internal document type (SPE, TIFF, etc.)
DM_EXPOSEC	Exposure in seconds
DM_FILEDATE	Date file was created
DM_FILENAME	Name of file on disk
DM_FILETITLE	Name of file in window
DM_FILEVERSION	Software Version
DM_FLATFIELDAPPLIED	Flat field flag
DM_FLATFIELDNAME	Flat field file name
DM_FRAMECOUNT	Frames in this file
DM_FRAMEINPROS	Number of current frame in process/storage
DM_FRAMESIZE	Size of one frame in bytes
DM_GAIN	Gain
DM_GEOMETRIC	Geometric transformation: rotate = 1, reverse = 2, flip = 4
DM_GPIBADD	GPIB address

Parameter Name	Description/Native Type
DM_GPIBCONTROLADD	GPIB controller address
DM_HOURS	Hour of experiment
DM_HWACCUMS	Hardware Accumulation flag
DM_INTERFACETYPE	Interface card
DM_INTLEVEL	Interrupt level
DM_INUSE	Increment or decrement in use count
DM_IOADD1	IO base address
DM_KINTRIGMODE	Kinetics Trigger Mode
DM_LASTFRAMERDY	Number of last frame finished acquiring
DM_MIN	Minutes of experiment
DM_MINBLK	Minimum block grouping for skip before valid pixel
DM_NUMEXPREPEATS	Number of Times Experiment Repeated
DM_NUMEXPACCUMS	Number of Times Experiment Accumulated
DM_NUMFRAMES	Number of frames in file
DM_NUMMINBLK	Number of min blocks to do before doing geometric
DM_NUMROI	Number of ROIs in header. If 0, 1 is assumed
DM_NUMSKPPERCLN	Number of skips per clean
DM_PIXELSIZE	Size of each data point in bytes
DM_READOUTMODE	Readout FULL/KINETICS/FRAME XFER
DM_SCANCOUNT	Scans in this file
DM_SECS	Seconds of experiment
DM_SHUTTERCOMP	Shutter compensation
DM_SHUTTERTYPE	Shutter type
DM_STOREFLAG	Flag to store incoming data
DM_STORESYNC	Store synchronous flag
DM_SWMADE	Software package ID: WinView = 1, WinSpec = 2
DM_SWVERSION	File header version
DM_TEMPFLAG	Flag = TRUE if temporary file
DM_TIMINGMODE	Timing mode of experiment
DM_USERCOMMENT1	User comment field 1
DM_USERCOMMENT2	User comment field 2
DM_USERCOMMENT3	User comment field 3

Parameter Name	Description/Native Type
DM_USERCOMMENT4	User comment field 4
DM_USERCOMMENT5	User comment field 5
DM_USERINFO	User information area
DM_WINDOWSIZE	Window size (kinetics)
DM_xCAL_CALIB_COUNT	Number of pixel-value pairs in array
DM_xCAL_COEFFS	Polynomial coefficient array
DM_xCAL_CURR_UNIT	The currently selected scaling unit
DM_xCAL_FACTOR	Factor for ditto
DM_xCAL_FILE_LABEL	Label for calibration file
DM_xCAL_INPUT_UNIT	The units of the PXL_VAL
DM_xCAL_LASERLINE	Laser line (nm) for rel cm-1
DM_xCAL_OFFSET	Offset for absolute data scaling
DM_xCAL_POLYNOM_ORDER	Order of calibration polynomial
DM_xCAL_POLYNOM_UNIT	The units of the COEFFS
DM_xCAL_PXL_POS	Pixel array
DM_xCAL_PXL_VAL	Wavelength value array
DM_xCAL_STRING	The label for the X axis
DM_xCAL_VALID	Flag is cal is valid
DM_XDIM	X dimension of actual data stored in this file
DM_XDIMDET	X Dimension of detector physical
DM_XLABEL	X label string
DM_YDIM	Y dimension of actual data stored in this file
DM_YDIMDET	Y Dimension of detector physical
DM_YLABEL	Y label string

Display

DI_PARAM (DocWindow Parameters)

Parameter Name	Description
DI_BAUTOASPECT	Maintain fixed image aspect ratio
DI_BAUTOZOOM	Maintain fixed image zoom factor
DI_BCOLBARON	Show color bar pane

Parameter Name	Description
DI_BFULLRANGEX	Set X axis to full range
DI_BHIDLINE	Draw 3-D with hidden line or surface
DI_BHIDSURF	Draw 3-D with hidden surface
DI_BHXSECTION	X cross section on or off
DI_BINFOBARON	Info Bar on or off
DI_BINITAUTOAUTO	TRUE autoscale every live scan
DI_BINITAUTORANGE	TRUE ignore range settings in page, FALSE, use them
DI_BINITAUTOSCALE	TRUE do autoscale, FALSE use intensity settings in page.
DI_BMARK3D	Draw cursor curve in different color
DI_BMARKCURV	Draw 3-D marker curves in different color
DI_BNEWDATA	Flag that data may have changed
DI_BPLOTFRAME	Flag to plot 3-D in MultiStripSameFrame mode
DI_BPRINTCOLOR	Flag to print in color
DI_BROIBOXON	Flag to show ROI box
DI_BTOOLBARON	Tool Bar on or off
DI_BUPDATE	Flag to update window contents
DI_BVXSECTION	Y cross section on or off
DI_BXAXLBLON	X axis label on or off
DI_BXAXON	X axis on or off
DI_BYAXLBLON	Y axis label on or off
DI_BYAXON	Y axis on or off
DI_BZENDACTIV	Adjust Z axis position interactively
DI_DATAINTENHI	Max data intensity for display
DI_DATAINTENLO	Min data intensity for display
DI_DATASTEP	Pixel, strip, or frame skip value
DI_DATAVAIL	Bounds of available data (in data units)
DI_DATAVIS	Bounds of visible data (in data units)
DI_DBRITE	Window brightness setting 0 – 100
DI_DCONTR	Window contrast setting 0 – 100
DI_LABELDAT	Label string for data
DI_LABELXAX	Label string for x-axis

Parameter Name	Description
DI_LABELYAX	Label string for y-axis
DI_NEEDAUTO	0 = no autoscale, 1 = do normal autoscale, 2 = do 5%/95%
DI_NMARKSTEP	Number of curves between 3-D marker curves
DI_PALETTE_SHAPE	Linear, log, histogram, etc.
DI_PALETTE_TYPE	Gray scale or false color
DI_PICTURE_TYPE	Graph, 3-D or Image (Read-only)
DI_ZAXISOFFSETX	X component of Z axis offset
DI_ZAXISOFFSETY	Y component of Z axis offset
DI_ZOOMXY	Ratio of picture size to data size, x & y.

Data Processing

PIDP_IMAGEMATHOPERATION (Arithmetic/Logical Operators)

Operator Name	Description
IMAGEMATH_ABSOLUTE	Take the absolute value of the first input
IMAGEMATH_ADDITION	Add the two inputs
IMAGEMATH_AND	Apply the bitwise AND operator to the two inputs
IMAGEMATH_COMPLEMENT	Apply the bitwise COMPLIMENT operator to the two inputs
IMAGEMATH_DIVISION	Divide the two inputs
IMAGEMATH_LOG10	Take the log base 10 of the first input
IMAGEMATH_MAX	Take the maximum value between the two inputs
IMAGEMATH_MIN	Take the minimum value between the two inputs
IMAGEMATH_MULTIPLICATION	Multiply the two inputs
IMAGEMATH_NATURALLOG	Take the natural log of the first input
IMAGEMATH_NOT	Apply the logical NOT operator to the first input
IMAGEMATH_OR	Apply the bitwise OR operator to the two inputs
IMAGEMATH_SQUARED	Square the first input
IMAGEMATH_SQUAREROOT	Take the Square Root of the first input
IMAGEMATH_SUBTRACTION	Subtract the two inputs
IMAGEMATH_XOR	Apply the bitwise XOR operator to the two inputs

Parameter Name	Description	Applicable Process									
		ProcBinSkip	ProcClipThresh	ProcCrossSect	ProcDilate	ProcErode	ProcFilter	ProcHistogram	ProcLUT	ProcOrient	ProcSobel
ORIENT_MODE	Orientation mode									✓	
ORIENT_REVERSE	Reverse image flag									✓	
ORIENT_ROTATE	Rotate image flag									✓	
SUMMATION	Cross section sum flag			✓							
THRESHOLD	Threshold flag		✓								
THRESHVAL	Threshold value		✓								
XCROSS	X-cross section			✓							
YCROSS	Y-cross section			✓							
ZCROSS	Z-cross section			✓							

PRC_PARAM (Process Output Control Parameters)

Parameter Name	Description
PRC_AUTOSAVE	Output file save options
PRC_DISPLAYOUTPUT	Display the results
PRC_FILEACCESS	Output file access type
PRC_FILEINCCOUNT	File increment counter value
PRC_FILEINCENABLE	Use auto-increment
PRC_NEWWINDOW	Open a new window
PRC_OUTFILENAME	Default output filename
PRC_OVERWRITECONFIRM	Confirm overwriting the output file
PRC_RUNNING	Process is running

PROCERR (Process Error Codes)

Parameter Name	Description
PROCERR_ABSIZEMISMATCH	Unequal ROI sizes for inputs A and B
PROCERR_BADINPUTA	Value of input A is invalid
PROCERR_BADINPUTB	Value of input B is invalid
PROCERR_BADOP	Arithmetic/logical operator is invalid
PROCERR_BADPARAM	Get/Set parameter is invalid
PROCERR_BADRANGEA	Invalid range set in input A
PROCERR_BADRANGEB	Invalid range set in input B
PROCERR_FILEOPENA	Cannot open input A
PROCERR_FILEOPENB	Cannot open input B
PROCERR_FILEOPENOUTPUT	Cannot open output file
PROCERR_NOERROR	No error
PROCERR_OUTFILECONFLICTA	Input A conflicts with output file
PROCERR_OUTFILECONFLICTB	Input B conflicts with output file

ProcFileAccessType (Process Output File Access Types)

Parameter Name	Description
PRCFA_APPEND	Append to output file if it exists
PRCFA_OVERWRITE	Overwrite output file if it exists

ProcSaveOpts (Process Output File Save Options)

Parameter Name	Description
PRCAS_AUTO	Automatically save file after each run
PRCAS_NOSAVE	Don't autosave or prompt the user
PRCAS_PROMPT	Prompt user whether to save after each run

Spectrograph

NAMETYPE (Spectrograph Instrument Name Types)

Parameter Name	Description
INST	Instrument name only
INSTCOMM	Instrument name + COM port name
INSTUSER	Instrument name + user name

SPEC_ERROR_CODE (General Spectrograph Error Codes)

Parameter Name	Description
CREATE_COMM_FAIL	Failure to Create Comm Port
CREATE_SPECT_FAIL	Failure to Create Spectrograph Object
INIT_COMM_FAIL	Failure to Communicate (Init failed)
NO_IO_ACCESS	Failure to Initialize I/O Device
NO_IO_DEVICE	No I/O Device
NO_SPEC_COMMUN	Communication Error
NO_SPEC_ERROR	No Error
PROCESS_FAILURE	Failure to Perform Process
SPEC_GEN_ERROR	General Error
WRONG_GRATING	Invalid Grating
WRONG_SLIT_WIDTH	Invalid Slit Size
WRONG_WAVELENGTH	Invalid Wavelength

SPEC_STEP_ERRORCODE (Stepper Motor Error Codes)

Parameter Name	Description
STEP_COMMANDERROR	Wrong command/parameter detected
STEP_CONNECTERROR	Communication failure
STEP_ERROR	Error during status report
STEP_NEEDRESET	Flag to remember reset answer checking
STEP_READY	Selected motor is ready
STEP_RESET	Flag to remember reset answer checking
STEP_WAITING	Waiting, selected motor running

SPECT_DLL_TYPES (Spectrograph DLL Types)

Parameter Name	Description
ACTON_TYPE	Acton DLL
GENERIC_SPECT_1_TYPE	Generic1 DLL
GENERIC_SPECT_2_TYPE	Generic2 DLL
SPEX_TYPE	Spex DLL

SPT_CMD (General Spectrograph Parameters)

Parameter Name	Description
SPT_ACCESS_ALL_GRAT_SPEEDS	T = can access all grating speeds at same time, F = only current
SPT_ACTIVE_GRAT_POS	New position for active grating
SPT_ACTIVE_GRATING_NUM	Grating number to set (change)
SPT_ACTIVE_MIRROR_LOC	Location of active mirror
SPT_ACTIVE_MIRROR_POS	New position for active mirror
SPT_ACTIVE_SLIT_LOC	Location of active slit
SPT_ACTIVE_SLIT_POS	New position for active slit
SPT_ACTIVE_TURRET_NUM	Active turret number
SPT_ADJUST_AVAIL	Linear adjust available flag
SPT_ARRAYSIZE	Size of used array in pixel
SPT_AUTOCALIBSUPPORTED	Auto calibration supported flag
SPT_BAUDRATE	COMM port baud rate
SPT_CALC_WL_INCENTERWL	Input for WL calc: center wavelength
SPT_CALC_WL_INDIST	Input for WL calc: distance
SPT_CALC_WL_OUTWL	Output for WL calc
SPT_CALIBUPDATE	System calibration is updated flag
SPT_COEFF	Coefficient value
SPT_COUNTERSUPPORTED	Counter supported flag
SPT_CTSFLOW	COMM port CTS flow control
SPT_CUR_GRATING	Index of previous used grating (1-based)
SPT_CUR_POSITION	Current center of ROI position
SPT_CURRGRATING	Current grating number (1-based)

Parameter Name	Description
SPT_CURRTURRET	Current turret number (1-based)
SPT_CUSTOM_NAME	User defined name for generic types
SPT_DATABITS	COMM port # data bits
SPT_DEF_DETECTORANGLE	Default detector angle value
SPT_DEF_FOCALLENGTH	Default focal length value
SPT_DEF_GRAT_BACKLASH_REQ	Default grating backlash required
SPT_DEF_GRAT_BACKLASHSTEPS	Default grating backlash steps
SPT_DEF_GRAT_MAX_SPEED	Default maximum grating speed
SPT_DEF_GRAT_MIN_SPEED	Default minimum grating speed
SPT_DEF_GRAT_SPEED	Default grating speed
SPT_DEF_GRAT_STEPSPERUNIT	Default steps/unit for spectrograph
SPT_DEF_GRAT_USERNAME	Default user name
SPT_DEF_GRATINGSPERTURRET	Default gratings per turret
SPT_DEF_GROOVES	Default grooves as shown in combo box
SPT_DEF_INCLUSIONANGLE	Default inclusion angle value
SPT_DEF_MAGNIFICATION	Default magnification
SPT_DEF_PIXELWIDTH	Default detector pixel width
SPT_DEF_PORTADDRESS	Default port address
SPT_DEF_PORTNAME	Default port name
SPT_DEF_PORTTYPE	Default port type
SPT_DEF_REFGRATING	Default reference grating value
SPT_DEF_SLIT_BACKLASH_REQ	Default slit backlash required
SPT_DEF_SLIT_BACKLASHSTEPS	Default slit backlash steps
SPT_DEF_SLIT_MAX_SPEED	Default maximum slit speed
SPT_DEF_SLIT_MIN_SPEED	Default minimum slit speed
SPT_DEF_SLIT_SPEED	Default slit speed
SPT_DEF_SLIT_STEPSPERUNIT	Default slit steps per unit
SPT_DFLT_BIN_FLAG	For step & glue: 1 = default bin size always used
SPT_DLL_FILE_VERSION	File version
SPT_DLL_PRODUCT_VERSION	Product version
SPT_ENDING_WAVELENGTH	For step & glue: ending WL (in nm)
SPT_ERROR_CODE	General error code

Parameter Name	Description
SPT_EXCITATION	Laser wavelength or zero if ignored
SPT_FORCEMOVE	Force a move even if new == current
SPT_GENERIC_CHECKBOX	User defined checkbox
SPT_GENERIC_CHECKBOX_TXT	Title for user defined checkbox
SPT_GENERIC_COMBOBOX	User defined combo box
SPT_GENERIC_COMBOBOX_TXT	Title for user defined combo box
SPT_GENERIC_DLG_INDEX	Index of defining dialog controls
SPT_GENERIC_DLG_MAIN_TITLE	Title of generic dialog
SPT_GENERIC_EDITBOX	User defined edit box
SPT_GENERIC_EDITBOX_TXT	Title for user defined edit box
SPT_GENERIC_RADIOBUTTON	User defined radiobutton
SPT_GENERIC_RADIOBUTTON_GRP	Title for user defined radiobutton group
SPT_GENERIC_RADIOBUTTON_TXT	Title for user defined radiobutton
SPT_GENERIC_SPINBUTTON	User defined spin button
SPT_GLUE_BIN_SIZE	For step & glue: bin size (in nm)
SPT_GLUE_FILE_NAME	For step & glue: glue file name
SPT_GPIB_IO_SUPPORTED	GPIB I/O available
SPT_GRAT_ADJUST	Slope adjust value for HW-based calibration
SPT_GRAT_BACKLASH_REQ	T = grating backlash required
SPT_GRAT_CENTERADJ	Center adjust (0 offset) value
SPT_GRAT_DETECT_ANGLE	Detector angle
SPT_GRAT_FOCAL_LEN	Focal length value
SPT_GRAT_GROOVES	Grooves/nm of this grating
SPT_GRAT_HI_CAL_WAVE	High calibration wavelength
SPT_GRAT_HI_POS_WAVE	High position wavelength
SPT_GRAT_INCL_ANGLE	Inclusion angle (gamma)
SPT_GRAT_LINEAR_ADJ	Linear adjust value for SW-based calibration
SPT_GRAT_LO_CAL_WAVE	Low calibration wavelength
SPT_GRAT_LO_POS_WAVE	Low position wavelength
SPT_GRAT_MAG	Magnification
SPT_GRAT_MOVE_MODE	Grating movement mode (i.e. slew/scan)
SPT_GRAT_OFFSET	Linear adjust value for HW-based calibration

Parameter Name	Description
SPT_GRAT_REF_GRATING	Reference grating
SPT_GRAT_REF_WAVE	Reference wavelength
SPT_GRAT_SLOPE_ADJ	Slope adjust value for SW-based calibration
SPT_GRAT_STEPSPUNIT	Steps/unit for this grating
SPT_GRAT_STEPSUNIT_UNIT	Unit of grating steps/unit
SPT_GRAT_TYPE	Grating type (normal, mirror, not installed)
SPT_GRAT_USERNAME	User entered portion grating name as string
SPT_GRATING_SPEED	Current grating speed (offset)
SPT_GRATING_SPEED_INDEX	Index of grating to access at upper class level
SPT_GRATING_SPEED_MAX	Maximum grating speed (offset)
SPT_GRATING_SPEED_MIN	Minimum grating speed (offset)
SPT_GRATING_SPEED_NAME	Name for grating speed (i.e. 'ramp')
SPT_GRATING_SPEED_UNITS	Units text for grating speeds
SPT_GRATINGBACKLASHSTEPS	Grating backlash steps
SPT_GRATINGSPERTURRET	Gratings per turret
SPT_GRATINGWARNING	When changing grating flag
SPT_INST_CUR_GRAT_NUM	Get current grating number from instrument
SPT_INST_CUR_GRAT_POS	Get current grating position from instrument
SPT_INST_CURR_TURRET	Get current turret number from instrument
SPT_INST_GRAT_GROOVES	Grooves/nm of this grating from instrument
SPT_INST_GRAT_SPEED	Grating speed from instrument
SPT_INST_GRAT_USERNAME	User name taken from instrument info
SPT_INST_MIRROR_LOCATION	Location of mirror from instrument
SPT_INST_MIRROR_POSITION	Position of mirror from instrument
SPT_INST_MODELNAME	Model name as read from instrument
SPT_INST_NUM_MIRRORS	Number of supported mirrors from instrument
SPT_INST_NUM_SLITS	Number of supported slits from instrument
SPT_INST_NUMTURRETS	Number of turrets from instrument
SPT_INST_SERIALNUMBER	Serial number as read from instrument
SPT_INST_SLIT_LOCATION	Location of slit from instrument
SPT_INST_SLIT_POSITION	Position of slit from instrument
SPT_INST_SLIT_SPEED	Slit speed from instrument

Parameter Name	Description
SPT_INSTCOMMNAME	Inst name + Comm port name as string
SPT_INSTNAME	Actual spectrograph inst name as string
SPT_INSTUSERNAME	Inst name + user name as string
SPT_LASERWARNING	Warn when crossing laser line flag
SPT_LINEARADJUSTSUPPORTED	Adjustment of linear adjust supported
SPT_MANUFACTURE_NAME	Returns manufacturer of specific instrument
SPT_MARK_EXCITATION	Spectrographs marker position
SPT_MAX_GRATINGS_PER_TURRET	Max gratings per turret allowed
SPT_MAX_NUM_MIRRORS	Max mirrors allowed
SPT_MAX_NUM_SLITS	Max slits allowed
SPT_MAX_NUM_TURRETS	Max turrets allowed
SPT_MAX_SLIP_POS	Default max slit position
SPT_MIN_OVERLAP	For step & glue: minimum overlap (in nm)
SPT_MIN_SLIP_POS	Default min slit position
SPT_MIRROR_CURPOSITION	Current position of mirror (front, lateral, etc.)
SPT_MIRROR_ENABLED	T = mirror enabled
SPT_MIRROR_LOCATION	Location of mirror (entrance, exit, etc.)
SPT_MIRROR_NEWPOSITION	Requested position of mirror (front, lateral, etc.)
SPT_MODELNAME	Model name
SPT_MOTOR	Motor number
SPT_NEED_GRATING_SPEED_SET	T = grating speed needs to be set
SPT_NEED_SLIT_SPEED_SET	T = slit speed needs to be set
SPT_NEW_GRATING	Index of active grating (1 based)
SPT_NEW_MIRROR_INDEX	Index of newly added mirror
SPT_NEW_POSITION	Next position to move center of ROI to
SPT_NEW_SLIT_INDEX	Index of newly added slit
SPT_NMTOSTEPS_STEPS	Steps converted from wavelength (nm)
SPT_NMTOSTEPS_WL	Wavelength (nm) to convert to steps
SPT_NUM_SETABLE_GRAT_SPEEDS	Number of speeds for each grating
SPT_NUM_SETABLE_SLIT_SPEEDS	Number of speeds for each slit
SPT_NUM_SLITS	Number of supported slits
SPT_NUM_SPECT_AVAIL	Total number of spectrographs available

Parameter Name	Description
SPT_NUM_STEPS	For step & glue: number of steps
SPT_NUMCOEFFS	Number of coefficients
SPT_NUMMIRRORS	Number of supported mirrors
SPT_NUMSTAGES	Number of stages per instrument
SPT_NUMTURRETS	Number of turrets
SPT_OFFSET_AVAIL	0 offset value available flag
SPT_OFFSETADJUSTSUPPORTED	Adjustment of offset supported
SPT_ORDER	Optical observed order
SPT_PARITY	COMM port parity
SPT_PIXEL_WIDTH	Pixel width
SPT_PORT_ADDRESS	Port number (serial) or address (GPIB)
SPT_PORT_NAME	Port device name
SPT_PORT_TYPE	Port type – serial (6), GPIB (5)
SPT_POSITIONMEMORY	Position remembered by spectrograph flag
SPT_POSITIONSUPPORTED	Position supported flag
SPT_PROCESS_MOVE_STATUS	Status of move processes
SPT_RESET	Hardware reset available or not
SPT_RESET_TIMEOUT_MS	Reset timeout in milliseconds
SPT_ROI_CENTER	Center of the scan pattern
SPT_ROI_OFFSET	ROI offset for the new position
SPT_SERIAL_IO_SUPPORTED	Serial I/O available
SPT_SERIALNUMBER	Serial number
SPT_SLAVE_CNT	Number of associated step spectrographs sharing the same port
SPT_SLIT_BACKLASH	Slit backlash steps
SPT_SLIT_BACKLASH_REQ	T = slit backlash required
SPT_SLIT_CURPOSITION	Current position of slit
SPT_SLIT_ENABLED	T = slit enabled
SPT_SLIT_LOCATION	Location of slit (Front entrance, side exit, etc.)
SPT_SLIT_NEWPOSITION	Requested position of slit
SPT_SLIT_POS_UNITS	Slit units (i.e., μm)
SPT_SLIT_SPEED	Current slit speed (offset)

Parameter Name	Description
SPT_SLIT_SPEED_INDEX	Index of slit to access at upper class level
SPT_SLIT_SPEED_MAX	Maximum slit speed (offset)
SPT_SLIT_SPEED_MIN	Minimum slit speed (offset)
SPT_SLIT_SPEED_NAME	Name for slit speed (i.e. 'start')
SPT_SLIT_SPEED_UNITS	Units of text for slit speeds
SPT_SLIT_STEPSUNIT	Current slit steps/unit
SPT_SLIT_STEPSUNIT_UNIT	Unit of slit steps/unit
SPT_SPECTS_SUPPORTED	Spectrometers (used at high level)
SPT_SPEED_FOR_EACH_GRATING	T = each grating has own speed, F = 1 speed for all
SPT_SPEED_FOR_EACH_SLIT	T = each slit has own speed, F = 1 speed for all
SPT_STARTING_CENTER_WAVE	For step & glue: starting center wave (in nm)
SPT_STARTING_WAVELENGTH	For step & glue: starting WL (in nm)
SPT_STEP_ERROR_CODE	Error code for motors
SPT_STEP_SIZE	For step & glue: step size
SPT_STOPBITS	COMM port # stop bits
SPT_TRIMTOPPRECISION	Value trimmed to precision
SPT_TYPE	Spectrograph type ID of this object (get only)
SPT_USERNAME	User entered portion of name as string
SPT_VALIDITYFLAG	T = valid object, F = not valid
SPT_WAVE_ORIENTED	Wavelength oriented instrument flag
SPT_WL_PRECISION	Precision of wavelength values

SPT_PROCESS (General Spectrograph Processes)

Parameter Name	Description
SPTP_ADD_MIRROR	Adds an empty mirror structure
SPTP_ADD_SLIT	Adds an empty slit structure
SPTP_CALC_WL	Calculate WL from input parameters
SPTP_CREATE_PORT	Creates a communications port
SPTP_EVAL_STEP_STATUS	Evaluates step status
SPTP_GET_SLOPEADJ_WL	Special slope adjust calc

Parameter Name	Description
SPTP_INITIALIZE	Reads from instrument
SPTP_INST_GRATING_SPEED	Get grating speed(s) from instrument
SPTP_INST_LOADCONFIGURATION	Reads inst config & cur positions, sets grating/mir/slit objs
SPTP_INST_MODELNAME	Read from instrument
SPTP_INST_READCURPOSITIONS	Reads current positions of ALL items
SPTP_INST_RESETEALLPOSITIONS	Reads current positions and resets ALL items
SPTP_INST_SERIALNUMBER	Reads from instrument
SPTP_INST_SLIT_SPEED	Get slit speed(s) from instrument
SPTP_NM_TO_STEPS	Converts nm value to steps
SPTP_PROCESS_MOVE_STATUS	Get status of move processes
SPTP_PROCESS_MOVES	Process spectrograph moves as setup
SPTP_REMOVE_ALL_MIRRORS	Removes ALL mirrors
SPTP_REMOVE_ALL_SLITS	Removes ALL slits
SPTP_RESET	Resets instrument
SPTP_SET_GRATING	Changes grating
SPTP_SET_GRATING_POS	Move the active grating
SPTP_SET_GRATING_SPEED	Set grating speed(s) on instrument
SPTP_SET_MIRROR_POS	Move the active mirror
SPTP_SET_SLIT_POS	Move the active slit
SPTP_SET_SLIT_SPEED	Set slit speed(s) on instrument
SPTP_SET_SLOPEADJ_WL	Special slope adjust calc
SPTP_SET_TURRET	Changes turret
SPTP_SETUP_OPTIONS	Reads options installed on instrument
SPTP_START	Sets up all processes and starts them
SPTP_STOP	Stops all processes
SPTP_TRIMTOPRECISION	Trims value to required precision

Pulser

PITG_BRACKET_TYPE (PI-Max bracket type)

Parameter Name	Description
GEN3_BRACKET	PI-Max – Gen III bracket
ORIGINAL_BRACKET	PI-Max – Original bracket

PITG_COUPLING (External trigger coupling)

Parameter Name	Description
AC_COUPLED	AC coupled
DC_COUPLED	DC coupled

PITG_DELAY_FROM (Where delay is calculated from)

Parameter Name	Description
PTG_EXT_TRIG_IN	Calculate delay from Trigger-In
PITG_T0_OUT	Calculate delay from T0

PITG_ERROR_CODES (PITG error codes)

Parameter Name	Description
EC_DG535_DELAY_LINKAGE_ERROR	DG535 – Delay linkage error
EC_DG535_DELAY_RANGE_ERROR	DG535 – Delay range error
EC_DG535_RECALLED_DATA_WAS_CORRUPTED	DG535 – Recalled data was corrupted
EC_DG535_UNRECOGNIZED_COMMAND	DG535 – Unrecognized command
EC_DG535_VALUE_OUTSIZE_RANGE	DG535 – Value out of range
EC_DG535_WRONG_MODE_FOR_THE_COMMAND	DG535 – Invalid mode
EC_DG535_WRONG_NUMBER_OF_PARAMETERS	DG535 – Invalid number of parameters
EC_PG200_NEED_SETTLING_TIME	PG200 – Need settling time
EC_PITG_COMM_DLL	DLL load failure
EC_PITG_COMM_PORT	Communication port failure
EC_PITG_ILLEGAL_VALUE	TG_CMD value out of range

Parameter Name	Description
EC_PITG_NO_ERROR	No error
EC_PITG_PTG_CNTRRAM_LOAD	PTG – RAM load error
EC_PITG_PTG_CNTRRAM_READ	PTG – RAM read error
EC_PITG_PTG_CNTRRAM_WRITE	PTG – RAM write error
EC_PITG_PTG_REACHED_END_OF_RAM	PTG – Out of RAM
EC_PITG_PTG_TRIGGER_COUNT_OVERFLOW	PTG – Trigger count overflow
EC_PITG_READ_FR_TG_FAILED	Read from pulser failure
EC_PITG_SEND_TO_TG_FAILED	Write to pulser failure
EC_PITG_SHORT_PULSE_DEL_FOR_BRACKET	PI-Max – Pulse delay shorter than bracket delay
EC_PITG_UNINIT_VAR_AT_PROCESS	Not all necessary TG_CMD's are set before TG_PROCESS
EC_PITG_UNKNOWN_ERROR	Unknown error (bad code)
EC_PITG_UNSUPPORTED_PARM_ERROR	Invalid TG_CMD TG_PROCESS

PITG_GATINGMODE (PG200:gating mode status)

Parameter Name	Description
GATING_DISABLED	PG200 – Disabled
GATING_ENABLED	PG200 – Enabled

PITG_INCREMENT_TYPES (Sequential increment types)

Parameter Name	Description
INC_CUSTOM	Increment is custom (does not touch arrays)
INC_EXPONENTIAL	Exponential increment
INC_FIXED	Fixed increment

PITG_PTG_TRIG_COUNT_TYPE (PTG triggers counted)

Parameter Name	Description
TRIGCNT_MAIN	PTG – Count main triggers only
TRIGCNT_MAIN_BURST	PTG – Count main and burst triggers

PITG_PULSING_MODE (Pulsing mode)

Parameter Name	Description
CONTINUOUS_PULSE	Continuous pulse
SEQUENTIAL_PULSE	Sequential pulse

PITG_SLOPE (External trigger slope)

Parameter Name	Description
NEGATIVE_SLOPE	Negative slope
POSITIVE_SLOPE	Positive slope

PITG_TERMINATION (External trigger termination)

Parameter Name	Description
FIFTY_OHMS	Fifty ohms
HIGH_Z	High impedance

PITG_TRIGMODES (Pulse triggering modes)

Parameter Name	Description
USE_EXTERNAL_TRIG	External trigger
USE_INTERNAL_TRIG	Internal trigger
USE_SINGLESHOT	PTG DG535 – Single shot
USE_SINGLESHOT_INTERNAL	PTG – Single shot (internal trigger)

PITG_TRIGSYNCMODES (Where to synch PG200 to)

Parameter Name	Description
SYNC_TO_GATE	PG200 – Synchronize to gate
SYNC_TO_TRIGGER	PG200 – Synchronize to trigger

PITG_TYPES (Pulser types)

Parameter Name	Description
PI_PG200	PG200 pulser
PI_PTG	PTG pulser
PITG_NONE	No pulser
SRS_DG535	DG535 pulser

TG_CMD (Pulser settings)

Parameter Name	Description
TGC_ANTICIPATE_ACTIVE	PTG – Enable (non-zero)/disable(0) anticipator
TGC_ANTICIPATE_TIME	PTG – Anticipator time (microseconds)
TGC_AUX_DELAY	PTG – Aux delay
TGC_AUX_DELAY_UNITS	PTG – Aux delay units
TGC_AUX_TRIGTOGATE	PG200 – Trigger to gate aux delay
TGC_AUX_TRIGTOGATE_UNITS	PG200 – Trigger to gate aux delay units
TGC_AUX_TRIGTOTRIG	PG200 – Trigger to trigger aux delay
TGC_AUX_TRIGTOTRIG_UNITS	PG200 – Trigger to trigger aux delay units
TGC_AUXTRIGSYNC_MODE	PG200 – Aux trigger synch mode
TGC_BURST_ACTIVE	Enable(non-zero)/disable(0) burst triggers
TGC_BURST_COUNTS	Number of burst triggers per main trigger
TGC_BURST_PERIOD	Burst trigger period
TGC_BURST_PERIOD_UNITS	Burst trigger period units
TGC_CLB_ACTIVE	Enable (non-zero)/disable(0) calibration
TGC_COUPLING	External trigger coupling
TGC_DEL_TRIGTOGATE	PG200 – Trigger to gate delay
TGC_DEL_TRIGTOGATE_UNITS	PG200 – Trigger to gate delay units
TGC_DEL_TRIGTOTRIG	PG200 – Trigger to trigger delay
TGC_DEL_TRIGTOTRIG_UNITS	PG200 – Trigger to trigger delay units
TGC_DELTRIGSYNC_MODE	PG200 – Delay trigger synch mode

Parameter Name	Description
TGC_DISPLAY_COMMAND_SENT	DG535 – Non-zero to display sent commands
TGC_ERROR_CODE	Last error code set
TGC_EXT_SYNC_TO_CONT	DG535 – Output D delay after gating pulse done
TGC_EXT_SYNC_TO_CONT_UNITS	DG535 – Output D delay units
TGC_TRIG_THRESHOLD	External trigger threshold
TGC_FRONT_PANEL_VIEWRATE	PG200 – Front panel view rate
TGC_GATINGMODE	PG200 – Synch mode
TGC_NUM_COMMAND_REPEATS	DG535 – Number of attempts to send a failed command
TGC_ON_CHIP_ACCUM	PTG – Number of triggers for on-chip accumulation
TGC_PULSE_DELAY	Gate pulse delay
TGC_PULSE_DELAY_FROM	PTG – Pulse delay measured from
TGC_PULSE_DELAY_UNITS	Gate pulse delay units
TGC_PULSE_WIDTH	Gate pulse width
TGC_PULSE_WIDTH_UNITS	Gate pulse width units
TGC_PULSING_MODE	Pulsing mode
TGC_SEQ_ARRAY_UNITS	Sequential array units
TGC_SEQ_AUX_ARRAY_HANDLE	Sequential aux delay array of doubles
TGC_SEQ_DELAY_ARRAY_HANDLE	Sequential delay array of doubles
TGC_SEQ_DELAY_END	Final sequential gate delay
TGC_SEQ_DELAY_END_UNITS	Final sequential gate delay units
TGC_SEQ_DELAY_START	Initial sequential gate delay
TGC_SEQ_DELAY_START_UNITS	Initial sequential gate delay units
TGC_SEQ_FAST_AMPLITUDE	Fast sequential amplitude
TGC_SEQ_FAST_TIMECONSTANT	Fast sequential time constant
TGC_SEQ_FAST_TIMECONSTANTUNITS	Fast sequential time constant units
TGC_SEQ_FX_DELAY_INC	Fixed sequential delay increment
TGC_SEQ_FX_DELAY_INC_UNITS	Fixed sequential delay increment units
TGC_SEQ_FX_WIDTH_INC	Fixed sequential width increment
TGC_SEQ_FX_WIDTH_INC_UNITS	Fixed sequential width increment units

Parameter Name	Description
TGC_SEQ_INC_TYPE	Sequencing increment type
TGC_SEQ_NUM_SHOTS	Number of timing patterns in a sequence
TGC_SEQ_ONCHIP_ACCUM_ARRAY_HANDLE	Sequential on-chip accum array of longs
TGC_SEQ_SLOW_AMPLITUDE	Slow sequential amplitude
TGC_SEQ_SLOW_TIMECONSTANT	Slow sequential time constant
TGC_SEQ_SLOW_TIMECONSTANTUNITS	Slow sequential time constant units
TGC_SEQ_WIDTH_ARRAY_HANDLE	Sequential width array of doubles
TGC_SEQ_WIDTH_END	Final sequential gate width
TGC_SEQ_WIDTH_END_UNITS	Final sequential gate width units
TGC_SEQ_WIDTH_START	Initial sequential gate width
TGC_SEQ_WIDTH_START_UNITS	Initial sequential gate width units
TGC_SLOPE	External trigger slope
TGC_SWEEP_NUM_OF_STEPS	Number of sweep steps
TGC_SWEEP_START_DEL_UNITS	Initial sweep delay units
TGC_SWEEP_START_DELAY	Initial sweep delay
TGC_SWEEP_STEP_SIZE	Sweep step size
TGC_SWEEP_STEP_SIZE_UNITS	Sweep step size units
TGC_TEMPERATURE	PTG – Get the temperature (read-only)
TGC_TERMINATION	External trigger termination
TGC_TRIG_FREQUENCY	Internal trigger frequency
TGC_TRIGGER_COUNT	PTG – Get the trigger count (read-only)
TGC_TRIGGER_COUNT_TYPE	PTG – Trigger count type to read
TGC_TRIGGER_MODE	Specify the trigger source
TGC_USE_FIXED_GATE	PI-Max – Use the fixed gate

TG_PROCESS (Pulser download processing)

Parameter Name	Description
TGP_ANTICIPATE_TIME	Load anticipator settings
TGP_BURST_ACTIVE	Load burst enable
TGP_GEN_SEQUENCE_ARRAY	Generate sequential arrays
TGP_PRELOAD_SEQUENCE	PG200/PTG – Load the sequence
TGP_RESET_TRIGGER_COUNT	Reset the trigger count
TGP_SEQ_ADD	Load sequential add
TGP_SEQ_CLEAR	Load sequential clear
TGP_SEQ_SKIP	Load sequential skip
TGP_SEQ_START	Load sequential start
TGP_SEQ_STOP	Load sequential stop
TGP_SET_AUX_TRIGTOGATE	Load trigger to gate aux delay
TGP_SET_AUX_TRIGTOTRIG	Load trigger to trigger aux delay
TGP_SET_BURST	Load burst settings
TGP_SET_DEL_TRIGTOGATE	Load trigger to gate delay
TGP_SET_DEL_TRIGTOTRIG	Load trigger to trigger delay
TGP_SET_FRONT_PANEL_VIEWRATE	Load front panel view rate
TGP_SET_MAIN_TRIGGER	Load main trigger settings
TGP_SET_SWEEP	Load sweep
TGP_SET_TIMING_PATTERN	Load a timing pattern
TGP_SET_TRIGGER_MODE	Load main trigger type

This page intentionally left blank.

ActonSpec Type Library Enumerations

GRATINGSPEEDS (Speeds at which grating can be moved)

Parameter Name	Description
FAST_SLEW_MODE	Grating speed: slew mode (faster)
NUM_GRATING_SPEEDS	Number of grating speeds available
SLOW_SCAN_MODE	Grating speed: scan mode (slower)

MANUFACTURERS (Name of spectrograph manufacturer)

Parameter Name	Description
MNFCTR_ID_ACTON	Acton Research Corp.
MNFCTR_ID_PI	Princeton Instruments
MNFCTR_ID_RS	Roper Scientific
NUM_MANUFACTURERS	Number of manufacturer names available

MIRRORLOCATIONS (Port locations of mirrors)

Parameter Name	Description
MIR_LOC_ENTRANCE	Mirror located at entrance port
MIR_LOC_EXIT	Mirror located at exit port
NUM_MIRROR_LOCATION	Number of mirror locations available

MIRRORPOSITIONS (Deflection position of mirror)

Parameter Name	Description
MIR_POS_FRONT	Mirror deflection position: front
MIR_POS_SIDE	Mirror deflection position: side
NUM_MIR_POS	Number of mirror positions available

SLITLOCATIONS (Port location of slits)

Operator Name	Description
NUM_SLIT_LOCATION	Number of slit locations available
SLIT_LOC_FR_ENTR	Slit located at front entrance port
SLIT_LOC_FR_EXIT	Slit located at front exit port
SLIT_LOC_SD_ENTR	Slit located at side entrance port
SLIT_LOC_SD_EXIT	Slit located at side exit port

SPECTROMETERS (Acton spectrometers supported)

Parameter Name	Description
ACTON_AM505	Acton AM505 Spectrometer
ACTON_SP150	Acton SP150 Spectrometer
ACTON_SP275	Acton SP275 Spectrometer
ACTON_SP300i	Acton SP300i Spectrometer
ACTON_SP500	Acton SP500 Spectrometer
ACTON_SP500i	Acton SP500i Spectrometer
ACTON_SP700	Acton SP700 Spectrometer
ACTON_SP700i	Acton SP700i Spectrometer
NUM_SPECTROGRAPHS	Number of spectrometers available
PI_320PI	PI 320PI Spectrometer

SpexSpec Type Library Enumerations

MANUFACTURERS (Name of spectrograph manufacturer)

Parameter Name	Description
MNFCTR_ID_ISA	Instruments S. A.
MNFCTR_ID_JY	Jobin-Yvon
MNFCTR_ID_SPEX	Spex
NUM_MANUFACTURERS	Number of manufacturer names available

MIRRORLOCATIONS (Port locations of mirrors)

Parameter Name	Description
MIR_LOC_ENTRANCE	Mirror located at entrance port
MIR_LOC_EXIT	Mirror located at exit port
NUM_MIRROR_LOCATION	Number of mirror locations available

MIRRORPOSITIONS (Deflection position of mirror)

Parameter Name	Description
MIR_POS_FRONT	Mirror deflection position: front
MIR_POS_SIDE	Mirror deflection position: side
NUM_MIR_POS	Number of mirror positions available

SLITLOCATIONS (Port location of slits)

Operator Name	Description
NUM_SLIT_LOCATION	Number of slit locations available
SLIT_LOC_FR_ENTR	Slit located at front entrance port
SLIT_LOC_FR_EXIT	Slit located at front exit port
SLIT_LOC_SD_ENTR	Slit located at side entrance port
SLIT_LOC_SD_EXIT	Slit located at side exit port

SPECTROMETERS (Spex spectrometers supported)

Parameter Name	Description
NUM_SPECTROGRAPHS	Number of spectrometers available
SPEX_270M	Spex 270M
SPEX232_RETRO	Spex232 Interface Retro-Fit
TRIAX_180	Triax 180
TRIAX_190	Triax 190
TRIAX_320	Triax 320

WinX/32 Automation and VBScript

WinX/32 Automation cannot only be called in Visual Basic, but it can also be called in VBScript. In fact, any program that supports VBScript can call WinX/32 Automation objects! Since VBScript is basically a subset of Visual Basic, it is inherently limiting. These limitations (check the Visual Basic or VBScript documentation for specifics) may not allow some of the automation functions to be called. A full explanation of this is beyond the scope of this manual, but it involves supported data types of the scripting engine. In order to allow full access to the automation objects, scripting versions of the ‘offending’ functions were created. These functions perform the same actions as the ‘offending’ functions, but their parameter lists are slightly different. Below is a table of the ‘offending’ functions with their scripting counterparts.

Also note that VBScript only supports arrays of variants. This means that array data of different types cannot be accessed from VBScript (although the array can still be passed through automation functions). If it is necessary to access the data through VBScript, first use the ArrayConverter object and convert the data to the variant data type. If it is desired to pass this array back to WinX/32 automation, make sure to convert the data back to the original type.

Replacement Scripting Functions

Object	Original Function	Scripting Version	Scripting Version Parameter List
DocFile	GetParam	SGetParam	(Param As DM_CMD, optional result As Any)
DocWindow	GetParam	SGetParam	(Param As DI_PARAM, optional result As Any)
DocWindows	GetParam	SGetParam	(Param As DI_PARAM, optional result As Any)
ExpSetup	GetParam	SGetParam	(Param As EXP_CMD, optional result As Any)
WinX32App	GetAppRect	SGetAppRect	(top As Any, left As Any, bottom As Any, right As Any)
	CloseOpenDocs	SCloseOpenDocs	(cDocs As Any)
All Process Objects	GetProcParam	SGetProcParam	(Param As PRC_PARAM, optional result As Any)
Configurable Process Objects	GetParam	SGetParam	(Param As PIIP_SetGetParam, optional result As Any)
Arithmetic/ Logical Process Objects	Run	SRun	(pInputA As ProcInput, pInputB As ProcInput, eOp As PIDP_IMAGEMATHOPERATION, szOutput As String, eOutType As dataType, optional pResult As Any)

Object	Original Function	Scripting Version	Scripting Version Parameter List
Non-Arithmetic/ Logical Process Objects	Run	SRun	(pInputA As ProcInput, szOutput As String, eOutType As dataType, optional pResult As Any)

Creating a Snap-In DLL

Introduction

To create a Snap-In DLL, use the PI Snap-In Generator Wizard to generate a generic SnapIn; then use the Resource Editor Add-In to add bitmaps for the toolbar buttons and strings for the menu item. The PI Snap-In Generator Wizard should be available from the Add-In menu. If it is not, try going to the Add-In Manager and make sure that the check box next to the Snap-In Generator Wizard is checked.

Using the Wizard:

1. Begin by closing all open projects (use the Remove Project item of the File menu if necessary.)
2. Then select the Snap-In Generator Wizard from the Add-In menu.
3. Go to the screen that requests the project name.
4. Click on the **ellipsis (...)** to browse for a folder, and enter the name of your snap-in.
5. Click on **Next** until the **Finish** button is available.
6. Then click on **Finish**. The Wizard will create various source files, then compile them to produce a Snap-In DLL. When it is done, you will get a message box with an **OK** button.
7. After you click on **OK**, you will be able to edit the Snap-In code to produce your own actions.

Making Your Snap-In Appear on the WinX/32 Toolbar:

To make your snap-in appear on the toolbar you will have to add toolbar bitmaps and menu string resources to the project. Additionally, you will have to register the DLL.

1. To add the toolbar bitmap and menu string, you will need the resource editor Add-In. This editor is not shipped with Visual Basic, version 5.0. However, you can locate the file **ResEditI.exe** at either www.microsoft.com or our ftp site: <ftp://roperscientific.com/software/official/winx32/support/ResEditI.exe> and then run it to install the Add-In resource editor on your computer.
2. Next, then use the Add-In resource editor in Visual Basic to enable it. This action will add the Resource Editor item to the VB Tools menu.
3. Create the bitmap you will be adding to the project. You will have to use Windows Paint or a similar program to create the bitmap first. The bitmap should be 16 X 15 pixels and should be saved as a 256-color bitmap. Give the bitmap an ID of 101.

4. Using the Resource Editor, add two strings to the resource file. The first one will contain the menu item text for your snap-in and should have an ID of 101. The second will have the status line text for the snap-in and should have an ID of 102.

Note: If you want a tooltip for your snap-in, you will have to be able to insert a carriage return/linefeed sequence into the string by pressing <CTRL><Tab> and <CTRL><Enter> in the string table (depending on the VB version, you may only need to press <CTRL><Enter>) - the tooltip text is separated from the status line text by these two characters.

5. Open the Immediate Window in Visual Basic (the menu item is under View).
6. Then type the single command Register (case-sensitive) and hit **Enter**.

Next time you start WinX/32, you should have a new menu item under Tools in WinX/32, and a new toolbar with at least one button. Selecting the menu item or the toolbar button runs the code in the PiSnapIn_OnCommand function of the clsSnapIn. You can put code in that function to bring up a form or do whatever you need.

Converting a Visual Basic "Exe" project to a "Snap-In":

1. In the Project Properties window, change the Project type to "ActiveX DLL".
2. Add a module called **DllMain**. To this module, add a Public subroutine called Main.
3. In the Project Properties window, change the startup object to **Sub Main**
4. In the Project References window, make sure "Roper Scientific's PIXBM32" and "Roper Scientific's Snap-In Library" are checked.
5. Add the RS-supplied class **clsSnapIn** to the project.

Putting up Your Main Form (omit if your program doesn't display a form):

1. In some module (**DllMain** will do) add a Public variable of type Form:


```
Public theForm As Form
```
2. In the **OnCommand** subroutine in **clsSnapIn**, display the form as follows:
 - a) Assign your form to the form variable
 - b) Show the form in modal state.
 - c) Assign "nothing" to the form variable.

Example:

```
Set theForm = frmMain
theForm.Show vbModal
Set theForm = Nothing
```

This approximates the behavior of an executable, where the form is loaded when the program starts, and is destroyed when the program ends. Your Snap-In DLL will be "running" as long as WinX/32 is running, but the toolbar icon or menu command will bring up the main form of the snap-in; closing the form will destroy it.

Running Your DLL:

1. You'll need to add a menu string and (optionally) a toolbar bitmap. To add these things to a Visual Basic project, you'll need the Resource Editor add-in. Install the add-in (you can get it from Roper Scientific or from the Microsoft Web site) and enable it using the add-in manager. Then use it to add a resource file to your project and add the bitmap and string resources. Finally, in **clsSnapIn**'s **GetMenuTextID** and **GetBitmapID** functions, return the value of the ID of the string and bitmap, respectively.
2. Compile your project. After it is compiled (successfully) open the Immediate Window and type the single command Register and hit **Enter**.

This page intentionally left blank.

Index

#			
3-D			
graphs	26, 63		
hidden line	62		
hidden surface	62		
marker curve color	62		
number of curves between markers	63		
plotting	62		
A			
Absolute data scaling	61		
Absorbance			
method	52		
source	49		
strip number	49		
AC_COUPLED	77		
Accumulations	58		
AcquireBackground	18		
AcquireFlatfield	18		
Activate			
DocWindow	27		
window	27		
ActiveX DLL	92		
Action spectrometers supported			
ACTON_AM505	86		
ACTON_SP150	86		
ACTON_SP275	86		
ACTON_SP300i	86		
ACTON_SP500	86		
ACTON_SP500i	86		
ACTON_SP700	86		
ACTON_SP700i	86		
NUM_SPECTROGRAPHS	86		
PI_320PI	86		
ACTON_TYPE	69		
ADC			
bit adjust	58		
NOP count	54		
offset reset	50		
rate	58		
resolution	59		
speed	50		
type	17, 59		
Add new spectrograph	44		
Add-In Manager	91		
Add-In resource editor	91		
Address	60		
AllocFrame	21		
API functions			
SendMessage		36	
ShowWindow		36	
Arithmetic/Logical process		30	
example code		33	
running		33	
ArrayConverter object		40	
Aspect ratio		61	
Asynchronous		59	
Automation interface		7, 15	
DLLs		7	
executables		7	
stand-alone programs		7	
Windows® Automation standard		7	
Autoscale		62, 63	
AvailValues		19	
Axes			
default parameters		28	
getting/setting		25	
B			
Background subtraction			
filename		59	
flag		59	
source file		51	
bAppend		24	
Basic		<i>See</i> Visual Basic	
Binning parameters		65	
BINSKIP_AVGFLAG		65	
BINSKIP_BINX		65	
BINSKIP_BINY		65	
BINSKIP_SKIPX		65	
BINSKIP_SKIPY		65	
Bitmaps		91	
ID		91, 93	
toolbar		93	
Blemish			
filename		59	
flag		52, 59	
source file		50	
Blocks			
min blocks		60	
min number of skip blocks		54	
minimum grouping		60	
bOn		36	
Bounds			
available data		62	
visible data		62	
Brightness setting		62	

Browser	10	Code Examples (cont.)	
bShowWindow	24	ExpSetup	13, 16, 17
C		GetAppWnd	36
CalibObj	23	InfoBox	36
example code	22	Non-arithmetic/logical process	34
Calibration		PITG	46, 47
DocWindow functions	27	PrintWindow	38
file label	61	Pulser	47
GetCalibration example	22	ROIRect	17
offset	61	ShowDemoBox	35
order of polynomial	61	SpectroObj	43
polynomial	23	SpectroObjMgr	43
polynomial coefficient	61	WinX32App	35, 36
scaling unit	61	WinXColorTable	29
units	23	Color	
usage flag	23	3-D marker curves	62
valid flag	61	cursor	62
values	23, 24	flag	62
wavelength	61	gray scale/false colorscale	63
X-axis	28	Color bar pane	61
CalibVal	23, 24	Comment	60
Calling syntax	12	CONTINUOUS_PULSE	79
Cascade	28	Contrast setting	62
Cleans	59	Controller	
Clear	39	address	60
ClearROIs	17	alive	51
CLIP_BOTH	65	initialization	53
CLIP_HIGH	65	multiple	59
CLIP_LOW	65	number	59
CLIPFLAGS	65	set temperature	57
Clipping/threshold parameters	65	status	55
Clock		type	59
register	53, 58	version	51
speed	59	Convert	
Close		example code	41
current window	26	Convert VB executable to Snap-In	92
DocFile	22	Convolution kernal	32
DocWindow	26	CopyConvert	
PrintWindow	39	example code	41
windows/file	22	Cosmic ray	
CloseAll		flag status	52
DocFiles	23	ray removal	59
DocWindows	27	sensitivity	51
PrintWindows	40	source file	51
CloseOpenDocs	36	Count	19
Code Examples		DocFiles	23
Arithmetic/Logical process	33	DocWindows	27
CalibObj	22	PrintWindows	40
Convert	41	CountOpenDocs	36
CopyConvert	41	Create	
Data Collection	11	pulser type	45
Data Manipulation	12	CREATE_COMM_FAIL	68
DocFile	13, 16, 20, 21, 22, 26	CREATE_SPECT_FAIL	68
DocInfo	24	Cross sections	
DocWindow	26	getting/setting	25, 28

Cross sections (cont.)		Delay time	51
turn X on/off	62	Detector	
turn Y on/off	62	temperature	49
CROSS_MATHTYPE	65	X-dimension	58
CROSS_SECTION_TYPE	65	Y-dimension	58
Cross-section parameters	65, 66	DG535	
CS_AVERAGE	65	pulser settings	81
CurrentValue	17, 19	DI_BAUTOASPECT	61
Cursor		DI_BAUTOZOOM	61
color	62	DI_BCOLBARON	61
control of	28	DI_BFULLRANGEX	62
coordinates	27	DI_BHIDLINE	62
intensity at position	28	DI_BHIDSURF	62
position	27	DI_BHXSECTION	62
X coordinate	28	DI_BINFOBARON	62
Y coordinate	28	DI_BINITAUTOAUTO	62
Z coordinate	28	DI_BINITAUTORANGE	62
CursorObj	28	DI_BINITAUTOSCALE	62
D		DI_BMARK3D	62
Data		DI_BMARKCURV	62
collection	11	DI_BNEWDATA	62
manipulation	12	DI_BPLOTFRAME	26, 62
source compensation	55	DI_BPRINTCOLOR	62
X-dimension	58	DI_BROIBOXON	26, 62
Y-dimension	58	DI_BTOOLBARON	62
YT file	58	DI_BUPDATE	62
Data display		DI_BVXSECTION	62
3-D graph	63	DI_BXAXLBLON	62
image (read-only)	63	DI_BXAXON	62
picture to image ratio	63	DI_BYAXLBLON	62
Data files		DI_BYAXON	62
default filename	51	DI_BZENDACTIV	62
overwrite/append	52	DI_DATAINTENHI	62
Data processing		DI_DATAINTENLO	62
Arithmetic/Logical process	30	DI_DATASTEP	62
configuring convolution kernel	32	DI_DATAVAIL	62
configuring the process	32	DI_DATAVIS	62
GetParam	33	DI_DBRITE	62
GetProcParam	32	DI_DCONTR	62
inputs	31	DI_LABELDAT	62
Non-Arithmetic/Logical process	30	DI_LABELXAX	62
output	32	DI_LABELYAX	63
running a Non-Arithmetic/Logical process	34	DI_NEEDAUTO	63
running an Arithmetic/Logical process	33	DI_NMARKSTEP	63
SetParam	33	DI_PALETTE_SHAPE	63
SetProcParam	32	DI_PALETTE_TYPE	63
dataType	19	DI_PARAM	25, 61
Datatypes		DI_PICTURE_TYPE	63
experiment	51	DI_ZAXISOFFSETX	63
DC_COUPLED	77	DI_ZAXISOFFSETY	63
Decrement count	60	DI_ZOOMXY	63
Default		DisplayUnit	23, 24
calibration	28	DLLs	13, 91
DefaultValue	19	ActiveX®	92
		integrated	7

DLLs (cont.)		DM_NUMROI	60
Snap-Ins	7	DM_NUMSKPPERCLN	60
Visual Basic Enterprise edition	7	DM_PIXELSIZE	60
Visual Basic Professional edition	7	DM_READOUTMODE	60
DM_ACCUMS	58	DM_SCANCOUNT	60
DM_ADCBITADJUST	58	DM_SECS	60
DM_ADCRATE	58	DM_SHUTTERCOMP	60
DM_ADCRESOLUTION	59	DM_SHUTTERTYPE	60
DM_ADCTYPE	59	DM_STOREFLAG	60
DM_ASYNEN	59	DM_STORESYNC	60
DM_BACKGROUNDAPPLIED	59	DM_SWMADE	60
DM_BACKSUBNAME	59	DM_SWVERSION	60
DM_BLEMISHAPPLIED	59	DM_TEMPFLAG	60
DM_BLEMISHFILE	59	DM_TIMINGMODE	60
DM_CLEANS	59	DM_USERCOMMENT1	60
DM_CLKSPD	59	DM_USERCOMMENT2	60
DM_CLKSPD_us	59	DM_USERCOMMENT3	60
DM_CMD	22, 32, 58	DM_USERCOMMENT4	61
DM_CONTROLLERNUM	59	DM_USERCOMMENT5	61
DM_CONTROLLERTYPE	59	DM_USERINFO	61
DM_COSMICAPPLIED	59	DM_WINDOWSIZE	61
DM_COSMICTHRESHOLD	59	DM_xCAL_CALIB_COUNT	61
DM_COSMICTYPE	59	DM_xCAL_COEFFS	61
DM_DATATYPE	59	DM_xCAL_CURR_UNIT	61
DM_DETTYE	59	DM_xCAL_FACTOR	61
DM_DLABEL	59	DM_xCAL_FILE_LABEL	61
DM_DOCTYPE	59	DM_xCAL_INPUT_UNIT	61
DM_EXPOSEC	59	DM_xCAL_LASERLINE	61
DM_FILEDATE	59	DM_xCAL_OFFSET	61
DM_FILENAME	59	DM_xCAL_POLYNOM_ORDER	61
DM_FILETITLE	59	DM_xCAL_POLYNOM_UNIT	61
DM_FILEVERSION	59	DM_xCAL_PXL_POS	61
DM_FLATFIELDAPPLIED	59	DM_xCAL_PXL_VAL	61
DM_FLATFIELDNAME	59	DM_xCAL_STRING	61
DM_FRAMECOUNT	59	DM_xCAL_VALID	61
DM_FRAMEINPROS	59	DM_XDIM	22, 61
DM_FRAMESIZE	59	DM_XDIMDET	61
DM_GAIN	59	DM_XLABEL	61
DM_GEOMETRIC	59	DM_YDIM	22, 61
DM_GPIBADD	59	DM_YDIMDET	61
DM_GPIBCONTROLADD	60	DM_YLABEL	61
DM_HOURS	60	DocFile	
DM_HWACCUMS	60	AllocFrame	21
DM_INTERFACETYPE	60	CalibObj	23
DM_INTLEVEL	60	calibration functions	27
DM_INUSE	60	Close	22
DM_IOADD1	60	COM object	36
DM_KINTRIGMODE	60	datatypes	20
DM_LASTFRAMERDY	20, 60	example code	13, 16, 20, 21, 22, 26
DM_MIN	60	GetCalibration	22
DM_MINBLK	60	GetDocument	16
DM_NUMEXPACCUMS	60	GetFrame	21
DM_NUMEXPREPEATS	60	GetParam	16, 21, 28
DM_NUMFRAMES	60	GetPixel	21
DM_NUMMINBLK	60	GetStrip	21

DocFile (cont.)		DocWindow (cont.)	
GetWindow	21	ZoomIn	26
interface	10	ZoomOut	26
opening a datafile	20	DocWindows	
OpenNew	20, 24	Cascade	28
parameters	58	CloseAll	27
PutFrame	21	Count	27
PutPixel	21	default calibration	28
PutStrip	21	default parameters	28
Save	22	GetActive	28
SaveAs	22	GetAt	27
SetCalibration	22	GetCalibration	28
SetFrame	21	GetParam	28
SetParam	12, 21	index value	27
Update	22	number open	27
DocFiles		SetCalibration	28
close all open DocFiles	23	SetParam	28
CloseAll	23	Tile	28
Count	23	DoEvents	13, 18
get number of open DocFiles	23	Dynamic Link Libraries	<i>See</i> DLLs
get value of open DocFile	23		
GetAt	23	E	
DocInfo		EC_DG535_DELAY_LINKAGE_ERROR	77
example code	24	EC_DG535_DELAY_RANGE_ERROR	77
properties	24	EC_DG535_RECALLED_DATA_WAS_CORR	
DocWindow	24, 28, 61	UPTED	77
Activate	27	EC_DG535_UNRECOGNIZED_COMMAND	77
calibration functions	28	EC_DG535_VALUE_OUTSIZE_RANGE	77
Close	26	EC_DG535_WRONG_MODE_FOR_THE_CO	
example code	26	MMAND	77
First	26	EC_DG535_WRONG_NUMBER_OF_PARAM	
GetCalibration	27	ETERS	77
GetCursor	27	EC_PG200_NEED_SETTLING_TIME	77
GetDocument	26	EC_PITG_COMM_DLL	77
GetHWnd	27	EC_PITG_COMM_PORT	77
GetParam	25	EC_PITG_ILLEGAL_VALUE	77
GetPosition	25	EC_PITG_NO_ERROR	78
GetROI	26	EC_PITG_PTG_CNTRRAM_LOAD	78
Last	26	EC_PITG_PTG_CNTRRAM_READ	78
MoveCursor	27	EC_PITG_PTG_CNTRRAM_WRITE	78
NewWindow	26	EC_PITG_PTG_REACHED_END_OF_RAM	78
Next	26	EC_PITG_PTG_TRIGGER_COUNT_OVERFL	
NextSkip	26	OW	78
Prev	26	EC_PITG_READ_FR_TG_FAILED	78
PrevSkip	26	EC_PITG_SEND_TO_TG_FAILED	78
redraw	25	EC_PITG_SHORT_PULSE_DEL_FOR_BRAC	
ROIs	18	KET	78
ScrollTo	27	EC_PITG_UNINIT_VAR_AT_PROCESS	78
SetCalibration	27	EC_PITG_UNKNOWN_ERROR	78
SetParam	25	EC_PITG_UNSUPPORTED_PARM_ERROR	78
SetPosition	25	Erase ROIs	17
SetROI	26	Error codes	
SetROIPos	27	pulser	77
UnZoom	26	spectrograph	68
Update	25, 26	stepper	68

Error syntax	12	EXP_DISABLE_LASER	52
Example Code		EXP_DOABSORB	52
Arithmetic/Logical process	33	EXP_DOBLEMISH	52
CalibObj	22	EXP_DOCOSMIC	52
Convert	41	EXP_EDGE_TRIGGER	52
CopyConvert	41	EXP_ETACTUAL	52
Data Collection	11	EXP_EVENT_HANDLE	52
Data Manipulation	12	EXP_EXPOSURE	52
DocFile	13, 16, 20, 21, 22, 26	EXP_FILEACCESS	52
DocInfo	24	EXP_FILEINCCOUNT	53
DocWindow	26	EXP_FILEINCENABLE	53
ExpSetup	13, 16, 17	EXP_FLATFLDNAME	53
GetAppWnd	36	EXP_FORCE_PP_INIT	53
Non-arithmetic/logical process	34	EXP_HD_CHECK	53
PITG	46, 47	EXP_HTCLK	53
PrintWindow	38	EXP_INTENSIFER_GAIN	53
Pulser	47	EXP_KINETICS_WINDOWSIZE	53
ROIrect	17	EXP_LOGIC_OUTPUT	54
ShowDemoBox	35	EXP_MAXTHRESHOLDVALUE	54
SpectroObj	42, 43	EXP_MIN_BLOCK	54
SpectroObjMgr	42, 43	EXP_MINTHRESHOLDVALUE	54
WinX32App	35, 36	EXP_NEWWINDOW	54
WinXColorTable	29	EXP_NOP_ADC	54
Excel	7, 8	EXP_NUM_MIN_BLOCK	54
Execute Macro	7	EXP_NUM_OF_STRIP_PER_CLN	55
EXP_5X_GAIN_ENABLE	49	EXP_NUMACCUMSEXP	55
EXP_ABSORBFILENAME	49	EXP_NUMBER_OF_CLEANS	55
EXP_ABSORBSTRIP	49	EXP_NUMREPEATSEXP	55
EXP_ACCESSNAME	49	EXP_OVERWRITECONFIRM	55
EXP_ACCUMS	49	EXP_READOUT_TIME	55
EXP_ACQBACK_SHUTTER	49	EXP_RET_SRCCMP	55
EXP_ACQFLAT_SHUTTER	49	EXP_ROICOUNT	55
EXP_ACTUAL_TEMP	49	EXP_RS170	55
EXP_ADC_OFFSET	50	EXP_RUNNING	55
EXP_ADC_TYPE	17, 50	EXP_RUNNING_APP	55
EXP_AMP_DIALOG	50	EXP_RUNNING_EXPERIMENT	56
EXP_ANALOG_GAIN	50	EXP_SAVEROPT	56
EXP_AUTOSAVE	50	EXP_SEQUENTS	56
EXP_AVGAIN	50	EXP_SETUPNAME	17, 56
EXP_AVGAIN_ENABLED	50	EXP_SHT_GATE_MODE	56
EXP_AVGAIN_SETPOINT	50	EXP_SHT_PREOPEN	56
EXP_BLEMISHFILENAME	50	EXP_SHUTTER_COMP_TIME_MS	56
EXP_CACCUMS	50	EXP_SHUTTER_CONTROL	57
EXP_CMD	16, 49	EXP_SKIPFRAMES	57
EXP_CONT_CLNS	51	EXP_STORE_STROBE	57
EXP_CONTROLLER_ALIVE	51	EXP_TEMP_STATUS	57
EXP_CONTROLLER_VERSION	51	EXP_TEMPERATURE	57
EXP_COSMICFILENAME	51	EXP_THRESHOLD_DEMAND_UPDATE	57
EXP_COSMICSENS	51	EXP_TIMING_MODE	57
EXP_CSEQUENTS	51	EXP_TTL_LINES	57
EXP_CUSTOM_SHUTTER	51	EXP_USEMAXTHRESHOLDING	57
EXP_DARKNAME	51	EXP_USEMINTHRESHOLDING	57
EXP_DATATYPE	51	EXP_USEROI	17, 58
EXP_DATFILENAME	51	EXP_VIDEO_TYPE	58
EXP_DELAY_TIME	51	EXP_VTCLK	58

EXP_XDIM	58	Flags (cont.)	
EXP_XDIMDET	58	calibration valid	61
EXP_YDIM	58	color printing	62
EXP_YDIMDET	58	data change	62
EXP_YTNAME	58	full chip/ROI	58
Experiment		show ROI	62
accumulation number	60	store incoming data	60
datatype	51	store synchronous	60
number of accums	55	temporary file	60
number of hardware cleans	55	update window	62
repeat	55	Flatfield	59
repeat number	60	shutter mode	49
setup file	56	source filename	53
setup functions	16	Flip	59
status	56	Frame skip value	62
Experiment setup parameters	<i>See</i> EXP_...	Frame transfer	59
Exposure time	52, 59	Frames	
ExpSetup		in a file	60
example code	13, 16, 17	number of sequential	56
GetDocument	16	G	
GetParam	16	Gain	59
IsAvail	17	Gating mode	78
IsAvail2	17	GATING_DISABLED	78
ROIs	18	GATING_ENABLED	78
SetParam	16	GEN3_BRACKET	77
Start	13, 16	GENERIC_SPECT_1_TYPE	69
Start2	16	GENERIC_SPECT_2_TYPE	69
StartFocus	13, 16	Geometric transformation	59
StartFocus2	16	Get window handle	<i>See. See</i>
Stop	16	GetActive	28
External trigger		GetAppRect	36
coupling	77	GetAppWnd example code	36
slope	79	GetAt	23, 44
termination	79	DocWindows	27
F		PrintWindows	40
False color	63	GetCalib	23
FAST_SLEW_MODE	85	GetCalibration	22
FIFTY_OHMS	79	DocWindow	27
File Open/Save-As Dialog	37	DocWindows	28
Files		GetCursor	27
autosave	50	GetDocument	
background source	51	DocWindow	26
blemish source	50	ExpSetup	16
blemishflag	52	GetEnumPara	42
cosmic source	51	GetEnumParam	45
default datafile name	51	GetEnumString	42, 45
experiment setup	56	GetFrame	21
overwrite/append	52	GetFromDoc	
FileType	24	data processing input	31
First strip or frame	26	GetHWnd	40
Fixed zoom factor	61	DocWindow	27
Flags		PrintWindow	40
3-D plot	62	GetMenuTextID	93
blemish file	52		

GetParam	13, 21, 27, 28, 33	IMAGEMATH_MAX	64
configurable process	33	IMAGEMATH_MIN	64
DocWindow	25	IMAGEMATH_MULTIPLICATION	64
ExpSetup	16	IMAGEMATH_NATURALLOG	64
PITG	46	IMAGEMATH_NOT	64
SpectroObj	42	IMAGEMATH_OR	64
GetPixel	21	IMAGEMATH_SQUARED	64
GetPosition	25	IMAGEMATH_SQUAREROOT	64
GetProcParam	32	IMAGEMATH_SUBTRACTION	64
GetRichEditWnd	40	IMAGEMATH_XOR	64
GetROI	17, 18, 26	INC_CUSTOM	78
GetStrip	21	INC_EXPONENTIAL	78
GetWindow	20, 21	INC_FIXED	78
GPIB address	59, 60	Increment	19
Grating speed		Increment count	60
FAST_SLEW_MODE	85	Index	17
NUM_GRATING_SPEEDS	85	InfoBox	36
SLOW_SCAN_MODE	85	INIT_COMM_FAIL	68
GRATINGSPEEDS	85	Input Unit	23
Gray scale	63	InputUnit	24
Group size		INST	68
X direction	18	INSTCOMM	68
Y direction	18	INSTUSER	68
Grouping	60	Intensity	28
H		max displayed	62
Hidden		min displayed	62
line	62	Interfaces	
surface	62	ISA	53
Hide	36	PCI	53
HIGH_Z	79	Interrupt level	60
HIGHCLIPVAL	65	ISA interface	53, 54
Histogram	63	IsAvail	17, 55, 57
Histogram parameters	65	PITG	46
HISTOGRAM_CUMULATIVE	65	SpectroObj	43
HISTOGRAM_GROUP	65	IsAvail2	17
HISTOGRAM_HIGHVAL	65	K	
HISTOGRAM_LOWVAL	65	Kernal parameters	65
HISTOGRAM_NORMAL	65	KERNELX	65
HISTOGRAM_OPERATIONS	65	KERNELY	65
I		Kinetics	59, 60, 61
I/O base address	60	L	
IDs		Label	23
bitmap	93	data	62
menu string	93	X-axis	62
software package	60	Y-axis	63
Image aspect ratio	61	Laser	
Image display	63	disable	52
Image math operators	64	line (nm)	61
IMAGEMATH_ABSOLUTE	64	wavenumber	23
IMAGEMATH_ADDITION	64	LaserPosition	23
IMAGEMATH_AND	64	Last strip or frame	26
IMAGEMATH_COMPLEMENT	64	Load	17
IMAGEMATH_DIVISION	64	spectrograph settings from INI file	43
IMAGEMATH_LOG10	64	Load function	9

LoadFactoryDefaults	37	Non-Arithmetic/Logical process (cont.)	
LOWCLIPVAL	65	Run	34
LUT_INPUT_FILENAME	65	running	34
M		Non-Arithmetic/Logical Process	
Macro	7	convolution kernel	32
Macro Basic	7	NTSC	58
Macro languages		NUM_GRATING_SPEEDS	85
Excel	7, 8	NUM_MANUFACTURERS	85, 87
Word	7, 8	NUM_MIR_POS	85, 87
MANUFACTURERS		NUM_MIRROR_LOCATION	85, 87
Acton spectrograph type	85	NUM_SLIT_LOCATION	86, 87
Spex spectrograph type	87	NUM_SPECTROGRAPHS	88
Math operators	64	O	
MaxValue	17, 19	Object Model	15
Menu strings	91, 93	Objects	
MicroMAX	59	ArrayConverter	40
Microsoft®		CalibObj	23
Excel	7	CursorObj	28
Windows	7	DocFile	20
Word	7	DocFiles	23
Min block size (skips)	54	DocInfo	24
Min blocks	60	DocWindow	25
Min number of skip blocks	54	DocWindows	27
MinValue	17, 19	ExpSetup	16
MIR_LOC_ENTRANCE	85, 87	PITG	45
MIR_LOC_EXIT	85, 87	PrintWindow	38
MIR_POS_FRONT	85, 87	PrintWindows	40
MIR_POS_SIDE	85, 87	ROIrect	18
Mirror		SpectroObj	41
location	85, 87	SpectroObjMgr	44
position	85, 87	ValidRange	19
MIRRORLOCATIONS	85	Windows API	36
MIRRORPOSITIONS	85	WinX32App	35
MNFCTR_ID_ACTON	85	OnCommand	92
MNFCTR_ID_ISA	87	Open	
MNFCTR_ID_JY	87	PrintWindow	39
MNFCTR_ID_PI	85	OpenFile	38
MNFCTR_ID_RS	85	Optional screen saver	56
MNFCTR_ID_SPEX	87	Order	23
Move	43	ORIENT_FLIP	65
MoveCursor	27	ORIENT_MODE	66
Multiple parameters	42	ORIENT_REVERSE	66
MultiStripSameFrame	62	ORIENT_ROTATE	66
N		ORIGINAL_BRACKET	77
NAMETYPE	68	Overwrite confirmation	55
NEGATIVE_SLOPE	79	P	
NewWindow	26	PairCount	23
Next strip or frame	26	Palette shape	63
NO_IO_ACCESS	68	Parameter lists	49
NO_IO_DEVICE	68	PCI interface	53
NO_SPEC_COMMUN	68	PentaMAX	59
NO_SPEC_ERROR	68	PG200	
Non-Arithmetic/Logical process	30	pulser settings	80
example code	34	pulsing mode	79

PG200 (cont.)		PRC_FILEACCESS	66
synchronize to gate or trigger	79	PRC_FILEINCCOUNT	66
PG200 gating mode	78	PRC_FILEINCENABLE	66
PI Snap-In Generator wizard	91	PRC_NEWWINDOW	66
PI_PG200	80	PRC_OUTFILENAME	66
PI_PTG	80	PRC_OVERWRITECONFIRM	66
Picture size	63	PRC_PARAM	66
PICTURE_TYPE	63	PRC_RUNNING	66
PIDP_IMAGE MATH OPERATION	64	PRCAS_AUTO	67
PIIP_SetGetParam	33, 65	PRCAS_NOSAVE	67
PI-Max bracket type	77	PRCAS_PROMPT	67
PITG		PRCFA_APPEND	67
example code	46, 47	PRCFA_OVERWRITE	67
GetParam	46	Previous strip or frame	26
IsAvail	46	PrintWindow	
Process	46	Clear	39
Reset	47	Close	39
SetGateLive	47	example code	38
SetParam	46	font selection	38
start pulser	47	GetHWnd	40
stop pulser	48	GetRichEditWnd	40
PITG functions	45	Open	39
GenEnumString	45	OpenFile	38
GetEnum Param	45	position of	38
PITG_BRACKET_TYPE	77	Print	13, 38
PITG_COUPLING	77	SaveAs	38
PITG_DELAY_FROM	77	SC_ARRANGE	39
PITG_ERROR_CODES	77	SC_CLOSE	39
PITG_GATINGMODE	78	SC_HSCROLL	39
PITG_INCREMENT_TYPES	78	SC_MAXIMIZE	39
PITG_NONE	80	SC_MINIMIZE	39
PITG_PTG_TRIG_COUNT_TYPE	78	SC_MOVE	39
PITG_PULSING_MODE	79	SC_NEXTWINDOW	39
PITG_SLOPE	79	SC_PREVWINDOW	39
PITG_T0_OUT	77	SC_RESTORE	39
PITG_TERMINATION	79	SC_SIZE	39
PITG_TRIGMODES	79	SC_VSCROLL	39
PITG_TRIGSYNCMODES	79	SetFont	38
PITG_TYPES	80	SetPosition	38
PIXBM32	92	SetSize	39
Pixel skip value	62	SysCmd	39
PixelLoc	23	PrintWindows	
Pixel-value pairs	61	close all open	40
Plot in 3-D	62	CloseAll	40
PolyCoeffs(5)	23	Count	40
Polynomial	61	GetAt	40
Polynomial calibration	23	number of open	40
Polynomial coefficients	23, 61	open window	40
Polynomial order	23	ProcBinSkip	32, 33, 34
Port		ProcClipThresh	32, 33, 34
ISA	53	ProcCrossSect	32, 33, 34
PCI	53	ProcDilate	32, 33, 34
POSITIVE_SLOPE	79	ProcErode	32, 33, 34
PRC_AUTOSAVE	66	PROCERR	67
PRC_DISPLAYOUTPUT	66	PROCERR_ABSIZEMISMATCH	67

PROCERR_BADINPUTA	67	PXL_VAL	61
PROCERR_BADINPUTB	67	Q	
PROCERR_BADOP	67	Quit WinX32App	37
PROCERR_BADPARAM	67	R	
PROCERR_BADRANGEA	67	ReadOrWrite	19
PROCERR_BADRANGEB	67	Readout	60
PROCERR_FILEOPENA	67	Readout time	55
PROCERR_FILEOPENB	67	Redraw	
PROCERR_FILEOPENOUTPUT	67	DocFile	22
PROCERR_NOERROR	67	DocWindow	26
PROCERR_OUTFILECONFLICTA	67	frame	25
PROCERR_OUTFILECONFLICTB	67	References browser	7, 9, 10
Process		Registration	7, 93
PITG	46	Registry	7
SPT_PROCESS settings	43	Remove spectrograph	44
Process status	55	ResEdit.exe	91
PROCESS_FAILURE	68	Reset	
ProcessManager	32	PITG	47
ProcFileAccessType	67	Resource editor	91, 93
ProcFilter	32, 33, 34	Reverse	59
ProcHistogram	32, 33, 34	RichEditCtrl	40
ProcInput	31	ROIrect	17, 18
ProcLUT	32, 33, 34	example code	17
ProcMath	32, 33	ROIs	17, 18, 58
ProcOrient	32, 33, 34	drawing	27
ProcSaveOpts	67	GetROI	18
ProcSobel	32, 33, 34	getting/setting	26
Progress Indicator	37	number in header	60
Project References	92	number stored	55
PTG		ROIrect	18
pulser settings	80	SetROI	18
trigger count type	78	show box	62
PTG_EXT_TRIG_IN	77	X coordinate units	18
Pulser		X coordinates	18
bracketing type	77	X group size	18
download processing	83	Y coordinate units	18
error codes	77	Y coordinates	18
example code	47	Y group size	18
external trigger coupling	77	zoom	26
external trigger slop	79	Roper Scientific	
external trigger termination	79	Snap-In library	92
sequential increment types	78	WinX/32 2.0 Type Library	7, 9
settings	80	Rotate	59
trigger count type	78	RS170	55
trigger delay origin	77	Run	
triggering modes	79	Non-arithmetic/logical process	34
Pulser object functions	45	Run button	7
Pulser types		S	
DG535	80	Save	
none	80	spectrograph settings	43
PG200	80	Save DocFile	22
PTG	80	SaveAll	37
PutFrame	20, 21		
PutPixel	20, 21		
PutStrip	20, 21		

SaveAs		Skip blocks	
DocFile	22	minimum grouping	60
PrintWindow	38	minimum number of	54
SC_ARRANGE	39	Skips	60
SC_CLOSE	39	min block size	54
SC_HSCROLL	39	min number of blocks	54
SC_MAXIMIZE	39	valuestrip	62
SC_MINIMIZE	39	Slit location	86, 87
SC_MOVE	39	SLIT_LOC_FR_ENTR	86, 87
SC_NEXTWINDOW	39	SLIT_LOC_FR_EXIT	86, 87
SC_PREVWINDOW	39	SLIT_LOC_SD_ENTR	86, 87
SC_RESTORE	39	SLIT_LOC_SD_EXIT	86, 87
SC_SIZE	39	SLITLOCATIONS	86
SC_VSCROLL	39	SLOW_SCAN_MODE	85
Scaling offset	61	Snap-Ins	7, 13, 91
Screen saver option	56	compile project	93
Scrollbar positions	27	create button on form	10
ScrollTo	27	library	92
SendMessage	36	register	93
Sequential frames	56	Visual Basic	7
SEQUENTIAL_PULSE	79	Visual Basic limitations	7
Set Param	21	Source compensation	55
SetCalib	23	SPE document type	59
SetCalibration	22	SPEC_DLL_TYPES	69
DocWindow	27	SPEC_ERROR_CODE	68
DocWindows	28	SPEC_GEN_ERROR	68
SetFont	38	SPEC_STEP_ERRORCODE	68
SetFrame	21	Spectrograph	
SetGateLive		Acton spectrograph types	85
PITG	47	manufacturer	
SetKernel	32	MNFCTR_ID_ACTON	85
SetParam	12, 17, 24, 25, 26, 27, 28, 33	MNFCTR_ID_PI	85
configurable process	33	MNFCTR_ID_RS	85
DocWindow	25	NUM_MANUFACTURERS	85
ExpSetup	16	Spectrograph DLL types	
PITG	46	ACTON_TYPE	69
SpectroObj	42	GENERIC_SPECT_1_TYPE	69
SetPosition	13	GENERIC_SPECT_2_TYPE	69
DocWindow	25	SPEX_TYPE	69
PrintWindow	38	Spectrograph error codes	
SetProcParam	32	CREATE_COMM_FAIL	68
SetROI	26	CREATE_SPECT_FAIL	68
SetROIPos	27	INIT_COMM_FAIL	68
SetSize	39	NO_IO_ACCESS	68
ShowDemoBox	11, 35	NO_IO_DEVICE	68
ShowFileOpen	37	NO_SPEC_COMMUN	68
ShowFileSaveAs	37	NO_SPEC_ERROR	68
ShowProgress	37	PROCESS_FAILURE	68
ShowWindow	36	SPEC_GEN_ERROR	68
Shutter		WRONG_GRATING	68
compensation	60	WRONG_SLIT_WIDTH	68
flatfield mode	49	WRONG_WAVELENGTH	68
mode	49	Spectrograph parameters (general)	See SPT_...
type	60	Spectrograph Processes	75

SpectroObj		SPT_DEF_DETECTORANGLE	70
example code	42, 43	SPT_DEF_FOCALLENGTH	70
GetEnumPara	42	SPT_DEF_GRAT_BACKLASH_REQ	70
GetEnumString	42	SPT_DEF_GRAT_BACKLASHSTEPS	70
GetParam	42	SPT_DEF_GRAT_MAX_SPEED	70
IsAvail	43	SPT_DEF_GRAT_MIN_SPEED	70
Move	43	SPT_DEF_GRAT_SPEED	70
Process	43	SPT_DEF_GRAT_STEPSPERUNIT	70
SetParam	42	SPT_DEF_GRAT_USERNAME	70
SpectroObjMgr		SPT_DEF_GRATINGSPERTURRET	70
Add	44	SPT_DEF_GROOVES	70
example code	42, 43	SPT_DEF_INCLUSIONANGLE	70
GetAt	44	SPT_DEF_MAGNIFICATION	70
Remove	44	SPT_DEF_PIXELWIDTH	70
Spex spectrograph		SPT_DEF_PORTADDRESS	70
MNFCTR_ID_ISA	87	SPT_DEF_PORTNAME	70
MNFCTR_ID_JY	87	SPT_DEF_PORTTYPE	70
MNFCTR_ID_SPEX	87	SPT_DEF_REFGRATING	70
NUM_MANUFACTURERS	87	SPT_DEF_SLIT_BACKLASH_REQ	70
Spex spectrometers supported		SPT_DEF_SLIT_BACKLASHSTEPS	70
NUM_SPECTROGRAPHS	88	SPT_DEF_SLIT_MAX_SPEED	70
SPEX_270M	88	SPT_DEF_SLIT_MIN_SPEED	70
SPEX232_RETRO	88	SPT_DEF_SLIT_SPEED	70
TRIAX_180	88	SPT_DEF_SLIT_STEPSPERUNIT	70
TRIAX_190	88	SPT_DFLT_BIN_FLAG	70
TRIAX_320	88	SPT_DLL_FILE_VERSION	70
SPEX_270M	88	SPT_DLL_PRODUCT_VERSION	70
SPEX_TYPE	69	SPT_ENDING_WAVELENGTH	70
SPEX232_RETRO	88	SPT_ERROR_CODE	70
SPT_ACCESS_ALL_GRAT_SPEEDS	69	SPT_EXCITATION	71
SPT_ACTIVE_GRAT_POS	69	SPT_FORCEMOVE	71
SPT_ACTIVE_GRATING_NUM	69	SPT_GENERIC_CHECKBOX	71
SPT_ACTIVE_MIRROR_LOC	69	SPT_GENERIC_CHECKBOX_TXT	71
SPT_ACTIVE_MIRROR_POS	69	SPT_GENERIC_COMBOBOX	71
SPT_ACTIVE_SLIT_LOC	69	SPT_GENERIC_COMBOBOX_TXT	71
SPT_ACTIVE_SLIT_POS	69	SPT_GENERIC_DLG_INDEX	71
SPT_ACTIVE_TURRET_NUM	69	SPT_GENERIC_DLG_MAIN_TITLE	71
SPT_ADJUST_AVAIL	69	SPT_GENERIC_EDITBOX	71
SPT_ARRAYSIZE	69	SPT_GENERIC_EDITBOX_TXT	71
SPT_AUTOCALIBSUPPORTED	69	SPT_GENERIC_RADIOBUTTON	71
SPT_BAUDRATE	69	SPT_GENERIC_RADIOBUTTON_GRP	71
SPT_CALC_WL_INCENTERWL	69	SPT_GENERIC_RADIOBUTTON_TXT	71
SPT_CALC_WL_INDIST	69	SPT_GENERIC_SPINBUTTON	71
SPT_CALC_WL_OUTWL	69	SPT_GLUE_BIN_SIZE	71
SPT_CALIBUPDATE	69	SPT_GLUE_FILE_NAME	71
SPT_CMD	42, 69	SPT_GPIB_IO_SUPPORTED	71
SPT_COEFF	69	SPT_GRAT_ADJUST	71
SPT_COUNTERSUPPORTED	69	SPT_GRAT_BACKLASH_REQ	71
SPT_CTSFLOW	69	SPT_GRAT_CENTERADJ	71
SPT_CUR_GRATING	69	SPT_GRAT_DETECT_ANGLE	71
SPT_CUR_POSITION	69	SPT_GRAT_FOCAL_LEN	71
SPT_CURRGRATING	69	SPT_GRAT_GROOVES	71
SPT_CURRTURRET	70	SPT_GRAT_HI_CAL_WAVE	71
SPT_CUSTOM_NAME	70	SPT_GRAT_HI_POS_WAVE	71
SPT_DATABITS	70	SPT_GRAT_INCL_ANGLE	71

SPT_GRAT_LINEAR_ADJ	71	SPT_MIRROR_NEWPOSITION	73
SPT_GRAT_LO_CAL_WAVE	71	SPT_MODELNAME	73
SPT_GRAT_LO_POS_WAVE	71	SPT_MOTOR	73
SPT_GRAT_MAG	71	SPT_NEED_GRATING_SPEED_SET	73
SPT_GRAT_MOVE_MODE	71	SPT_NEED_SLIT_SPEED_SET	73
SPT_GRAT_OFFSET	71	SPT_NEW_GRATING	73
SPT_GRAT_REF_GRATING	72	SPT_NEW_MIRROR_INDEX	73
SPT_GRAT_REF_WAVE	72	SPT_NEW_POSITION	73
SPT_GRAT_SLOPE_ADJ	72	SPT_NEW_SLIT_INDEX	73
SPT_GRAT_STEPSPUNIT	72	SPT_NMTOSTEPS_STEPS	73
SPT_GRAT_STEPSUNIT_UNIT	72	SPT_NMTOSTEPS_WL	73
SPT_GRAT_TYPE	72	SPT_NUM_SETTABLE_GRAT_SPEEDS	73
SPT_GRAT_USERNAME	72	SPT_NUM_SETTABLE_SLIT_SPEEDS	73
SPT_GRATING_SPEED	72	SPT_NUM_SLITS	73
SPT_GRATING_SPEED_INDEX	72	SPT_NUM_SPECT_AVAIL	73
SPT_GRATING_SPEED_MAX	72	SPT_NUM_STEPS	74
SPT_GRATING_SPEED_MIN	72	SPT_NUMCOEFFS	74
SPT_GRATING_SPEED_NAME	72	SPT_NUMMIRRORS	74
SPT_GRATING_SPEED_UNITS	72	SPT_NUMSTAGES	74
SPT_GRATINGBACKLASHSTEPS	72	SPT_NUMTURRETS	74
SPT_GRATINGSPERTURRET	72	SPT_OFFSET_AVAIL	74
SPT_GRATINGWARNING	72	SPT_OFFSETADJUSTSUPPORTED	74
SPT_INST_CUR_GRAT_NUM	72	SPT_ORDER	74
SPT_INST_CUR_GRAT_POS	72	SPT_PARITY	74
SPT_INST_CURR_TURRET	72	SPT_PIXEL_WIDTH	74
SPT_INST_GRAT_GROOVES	72	SPT_PORT_ADDRESS	74
SPT_INST_GRAT_SPEED	72	SPT_PORT_NAME	74
SPT_INST_GRAT_USERNAME	72	SPT_PORT_TYPE	74
SPT_INST_MIRROR_LOCATION	72	SPT_POSITIONMEMORY	74
SPT_INST_MIRROR_POSITION	72	SPT_POSITIONSUPPORTED	74
SPT_INST_MODELNAME	72	SPT_PROCESS	75
SPT_INST_NUM_MIRRORS	72	SPT_PROCESS_MOVE_STATUS	74
SPT_INST_NUM_SLITS	72	SPT_RESET	74
SPT_INST_NUMTURRETS	72	SPT_RESET_TIMEOUT_MS	74
SPT_INST_SERIALNUMBER	72	SPT_ROI_CENTER	74
SPT_INST_SLIT_LOCATION	72	SPT_ROI_OFFSET	74
SPT_INST_SLIT_POSITION	72	SPT_SERIAL_IO_SUPPORTED	74
SPT_INST_SLIT_SPEED	72	SPT_SERIALNUMBER	74
SPT_INSTCOMMNAME	73	SPT_SLAVE_CNT	74
SPT_INSTNAME	73	SPT_SLIT_BACKLASH	74
SPT_INSTUSERNAME	73	SPT_SLIT_BACKLASH_REQ	74
SPT_LASERWARNING	73	SPT_SLIT_CURPOSITION	74
SPT_LINEARADJUSTSUPPORTED	73	SPT_SLIT_ENABLED	74
SPT_MANUFACTURE_NAME	73	SPT_SLIT_LOCATION	74
SPT_MARK_EXCITATION	73	SPT_SLIT_NEWPOSITION	74
SPT_MAX_GRATINGS_PER_TURRET	73	SPT_SLIT_POS_UNITS	74
SPT_MAX_NUM_MIRRORS	73	SPT_SLIT_SPEED	74
SPT_MAX_NUM_SLITS	73	SPT_SLIT_SPEED_INDEX	75
SPT_MAX_NUM_TURRETS	73	SPT_SLIT_SPEED_MAX	75
SPT_MAX_SLIP_POS	73	SPT_SLIT_SPEED_MIN	75
SPT_MIN_OVERLAP	73	SPT_SLIT_SPEED_NAME	75
SPT_MIN_SLIP_POS	73	SPT_SLIT_SPEED_UNITS	75
SPT_MIRROR_CURPOSITION	73	SPT_SLIT_STEPSPUNIT	75
SPT_MIRROR_ENABLED	73	SPT_SLIT_STEPSUNIT_UNIT	75
SPT_MIRROR_LOCATION	73	SPT_SPECTS_SUPPORTED	75

SPT_SPEED_FOR_EACH_GRATING	75	Status	
SPT_SPEED_FOR_EACH_SLIT	75	controller	55
SPT_STARTING_CENTER_WAVE	75	process	55
SPT_STARTING_WAVELENGTH	75	Status Bar	36
SPT_STEP_ERROR_CODE	75	Status Bar message	37
SPT_STEP_SIZE	75	StatusBarMsg	37
SPT_STOPBITS	75	STEP_COMMANDERROR	68
SPT_TRIMTOPPRECISION	75	STEP_CONNECTERROR	68
SPT_TYPE	75	STEP_ERROR	68
SPT_USERNAME	75	STEP_NEEDRESET	68
SPT_VALIDITYFLAG	75	STEP_READY	68
SPT_WAVE_ORIENTED	75	STEP_RESET	68
SPT_WL_PRECISION	75	STEP_WAITING	68
SPTP_ADD_MIRROR	75	Stepper motor error codes	<i>See</i> STEP_...
SPTP_ADD_SLIT	75	Stop	16
SPTP_CALC_WL	75	PITG	48
SPTP_CREATE_PORT	75	Strip skip value	62
SPTP_EVAL_STEP_STATUS	75	Strips/clean	55
SPTP_GET_SLOPEADJ_WL	75	SUMMATION	66
SPTP_INITIALIZE	76	SW_SHOWMAXIMIZED	36
SPTP_INST_GRATING_SPEED	76	sync	59
SPTP_INST_LOADCONFIGURATION	76	SYNC_TO_GATE	79
SPTP_INST_MODELNAME	76	SYNC_TO_TRIGGER	79
SPTP_INST_READCURPOSITIONS	76	Synchronous	59
SPTP_INST_RESETALLPOSITIONS	76	Syntax error	12
SPTP_INST_SERIALNUMBER	76	SysCmd	39
SPTP_INST_SLIT_SPEED	76	System menu actions	
SPTP_NM_TO_STEPS	76	SC_ARRANGE	39
SPTP_PROCESS_MOVE_STATUS	76	SC_CLOSE	39
SPTP_PROCESS_MOVES	76	SC_HSCROLL	39
SPTP_REMOVE_ALL_MIRRORS	76	SC_MAXIMIZE	39
SPTP_REMOVE_ALL_SLITS	76	SC_MINIMIZE	39
SPTP_RESET	76	SC_MOVE	39
SPTP_SET_GRATING	76	SC_NEXTWINDOW	39
SPTP_SET_GRATING_POS	76	SC_PREVWINDOW	39
SPTP_SET_GRATING_SPEED	76	SC_RESTORE	39
SPTP_SET_MIRROR_POS	76	SC_SIZE	39
SPTP_SET_SLIT_POS	76	SC_VSCROLL	39
SPTP_SET_SLIT_SPEED	76		
SPTP_SET_SLOPEADJ_WL	76	T	
SPTP_SET_TURRET	76	Technical support	7
SPTP_SETUP_OPTIONS	76	Temperature	
SPTP_START	76	detector actual	49
SPTP_STOP	76	lock/unlock	57
SPTP_TRIMTOPPRECISION	76	set	57
SRS_DG535	80	TG_CMD	46, 80
ST130	59	TG_PROCESS	83
ST133	49	TGC_ANTICIPATE_ACTIVE	80
Stand-alone programs	7, 13	TGC_ANTICIPATE_TIME	80
Start	13, 16, 18	TGC_AUX_DELAY	80
PITG	47	TGC_AUX_DELAY_UNITS	80
Start2	16	TGC_AUX_TRIGTOGATE	80
StartFocus	13, 16, 18	TGC_AUX_TRIGTOGATE_UNITS	80
StartFocus2	16	TGC_AUX_TRIGTOTRIG	80
		TGC_AUX_TRIGTOTRIG_UNITS	80

TGC_AUXTRIGSYNC_MODE	80	TGC_SWEEP_START_DELAY	82
TGC_BURST_ACTIVE	80	TGC_SWEEP_STEP_SIZE	82
TGC_BURST_COUNTS	80	TGC_SWEEP_STEP_SIZE_UNITS	82
TGC_BURST_PERIOD	80	TGC_TEMPERATURE	46, 82
TGC_BURST_PERIOD_UNITS	80	TGC_TERMINATION	82
TGC_CLB_ACTIVE	80	TGC_TRIG_FREQUENCY	82
TGC_COUPLING	80	TGC_TRIG_THRESHOLD	81
TGC_DEL_TRIGTOGATE	80	TGC_TRIGGER_COUNT	46, 82
TGC_DEL_TRIGTOGATE_UNITS	80	TGC_TRIGGER_COUNT_TYPE	82
TGC_DEL_TRIGTOTRIG	80	TGC_TRIGGER_MODE	82
TGC_DEL_TRIGTOTRIG_UNITS	80	TGC_USE_FIXED_GATE	82
TGC_DELTRIGSYNC_MODE	80	TGP_ANTICIPATE_TIME	83
TGC_DISPLAY_COMMAND_SENT	81	TGP_BURST_ACTIVE	83
TGC_ERROR_CODE	81	TGP_GEN_SEQUENCE_ARRAY	83
TGC_EXT_SYNC_TO_CONT	81	TGP_PRELOAD_SEQUENCE	83
TGC_EXT_SYNC_TO_CONT_UNITS	81	TGP_RESET_TRIGGER_COUNT	83
TGC_FRONT_PANEL_VIEWRATE	81	TGP_SEQ_ADD	83
TGC_GATINGMODE	81	TGP_SEQ_CLEAR	83
TGC_NUM_COMMAND_REPEATS	81	TGP_SEQ_SKIP	83
TGC_ON_CHIP_ACCUM	81	TGP_SEQ_START	83
TGC_PULSE_DELAY	81	TGP_SEQ_STOP	83
TGC_PULSE_DELAY_FROM	81	TGP_SET_AUX_TRIGTOGATE	83
TGC_PULSE_DELAY_UNITS	81	TGP_SET_AUX_TRIGTOTRIG	83
TGC_PULSE_WIDTH	81	TGP_SET_BURST	83
TGC_PULSE_WIDTH_UNITS	81	TGP_SET_DEL_TRIGTOGATE	83
TGC_PULSING_MODE	81	TGP_SET_DEL_TRIGTOTRIG	83
TGC_SEQ_ARRAY_UNITS	47, 81	TGP_SET_FRONT_PANEL_VIEWRATE	83
TGC_SEQ_AUX_ARRAY_HANDLE	81	TGP_SET_MAIN_TRIGGER	83
TGC_SEQ_DELAY_ARRAY_HANDLE	81	TGP_SET_SWEEP	83
TGC_SEQ_DELAY_END	81	TGP_SET_TIMING_PATTERN	83
TGC_SEQ_DELAY_END_UNITS	81	TGP_SET_TRIGGER_MODE	83
TGC_SEQ_DELAY_START	81	THRESHOLD	66
TGC_SEQ_DELAY_START_UNITS	81	Threshold parameters	66
TGC_SEQ_FAST_AMPLITUDE	81	Threshold/clipping parameters	65
TGC_SEQ_FAST_TIMECONSTANT	81	Thresholding	
TGC_SEQ_FAST_TIMECONSTANTUNITS	81	maximum	57
TGC_SEQ_FX_DELAY_INC	81	minimum	57
TGC_SEQ_FX_DELAY_INC_UNITS	81	THRESHVAL	66
TGC_SEQ_FX_WIDTH_INC	81	TIFF document type	59
TGC_SEQ_FX_WIDTH_INC_UNITS	81	Tile	28
TGC_SEQ_INC_TYPE	82	Timing	60
TGC_SEQ_NUM_SHOTS	82	Toolbar	91
TGC_SEQ_ONCHIP_ACCUM_ARRAY_HANDLE	82	bitmaps	93
TGC_SEQ_SLOW_AMPLITUDE	82	Toolbar button	92
TGC_SEQ_SLOW_TIMECONSTANT	82	Toolbar buttons	7, 91
TGC_SEQ_SLOW_TIMECONSTANTUNITS	82	Tooltip	92
TGC_SEQ_WIDTH_ARRAY_HANDLE	82	Transfer speed	54
TGC_SEQ_WIDTH_END	82	Transformation	
TGC_SEQ_WIDTH_END_UNITS	82	flip	59
TGC_SEQ_WIDTH_START	82	reverse	59
TGC_SEQ_WIDTH_START_UNITS	82	rotate	59
TGC_SLOPE	82	TRIAX_180	88
TGC_SWEEP_NUM_OF_STEPS	82	TRIAX_190	88
TGC_SWEEP_START_DEL_UNITS	82	TRIAX_320	88
		TRIGCNT_MAIN	78

TRIGCNT_MAIN_BURST	78	Windows system menu actions	39
Trigger mode	60	WinSpec/32	
TTL control lines	57	program	7
Type Library	7	version	35
U		WinView/32	
UnZoom	26	program	7
Update	57	version	35
DocFile	22	WinX/32	7, 9, 11, 91, 92
DocWindow	25, 26	COM objects	36
UsageFlag	23	functions	15
USE_EXTERNAL_TRIG	79	hide/show application	36
USE_INTERNAL_TRIG	79	identify running program	35
USE_SINGLESHOT	79	WinX/32 2.0 Type Library	7, 9
USE_SINGLESHOT_INTERNAL	79	WinX/32 Automation Object Model	15
User comment field	60	WinX32App	10
V		CloseOpenDocs	36
ValidRange	17, 19	CountOpenDocs	36
VARIANT_BOOL	24	example code	35, 36
VB	<i>See</i> Visual Basic	GetAppRect	36
VICCD	59	GetAppWnd	36
Videotype		Hide	36
NTSC (1)	58	InfoBox	36
NTSC (2)	58	LoadFactoryDefaults	37
View menu	9	Quit	37
Visual Basic	7, 9, 21	SaveAll	37
Enterprise edition	7	ShowDemoBox	35
executable	7, 13	ShowFileOpen	37
executable to Snap-In conversion	8, 92	ShowFileSaveAs	37
Professional edition	7	ShowProgress	37
Run button	7	Status Bar	36
Snap-In Wizard	8	StatusBarMsg	37
snap-ins	7	WINX32Lib	16, 20, 29
syntax	12	WinXColorTable	
Tools menu	91	Blue(<i>i</i>)	29
versions	7	example code	29
W		Green(<i>i</i>)	29
WaitForExperiment	18	Red(<i>i</i>)	29
Wavelength	24, 61	Val(<i>i</i>)	29
Wavenumbers	23, 24	Wizards	
Window		PI Snap-In Generator	91
activate	27	Visual Basic Snap-In	8
brightness	62	WM_SYSCOMMAND	39
close	26	Word	7, 8
contrast	62	WRONG_GRATING	68
cursor	27	WRONG_SLIT_WIDTH	68
cursor coordinates	27	WRONG_WAVELENGTH	68
get handle	27	X	
redraw	26	X-axis	
ROI	26	axis on/off	62
scrollbar positions	27	calibration	22, 28
update	62	getting	22
X-axis information	27	label	61, 62
zoom	26	label on/off	62
		setting	22

X-axis calibration	27	Y-axis	
XCALIBUNIT	23	axis on/off	62
RELWAVENUM	24	label	63
XW_ABSWAVENUM	24	label on/off	62
XW_DATA	23	YCROSS	66
XW_EVOLTS	24	Y-dimension	
XW_SYSTEM	23	detector	58, 61
XCROSS	66	stored data	61
X-dimension of detector	61	YGroup	18
XGroup	18	YT data file	58
XUnits	18	YUnits	18
XW_ABSWAVENUM	24	Z	
XW_DATA	23	Z-axis	
XW_EVOLTS	24	interactive adjustment	62
XW_RELWAVENUM	24	offset	63
XW_SYSTEM	23	ZCROSS	66
XW_WAVELENGTH	23	Zero-based index	23
Y		Zoom, fixed	61
Y cross section	62	ZoomIn	26
		ZoomOut	26