AUTOMATION MANUAL

FOR

WAVE EXPERT SERIES OSCILLOSCOPE

May, 2008

LeCroy Corporation

700 Chestnut Ridge Road
Chestnut Ridge, NY 10977-6499
Tel: (845) 578 6020, Fax: (845) 578 5985

Internet: www.lecroy.com

916198 Rev A

CHAPTER 1: ABOUT AUTOMATION

CHAPTER 2: ABOUT RESULTS

APPENDIX A: DCOM

APPENDIX B: GLOSSARY

About this Manual

This manual is a reference guide to the “Automation” capabilities of LeCroy’s X-Stream™ DSOs.

The manual includes a complete list of all instrument controls that are available to a controlling application. In contrast to previously available remote control possibilities for LeCroy instruments, Automation enables the controlling application to run on the instrument itself.

- Part One, “About Automation,” covers the principles of control via Automation and offers practical examples. It also presents the overall hierarchy of controls accessible via Automation.
- Part Two, “Control Reference,” presents each of the controls in detail, including examples of use in many cases.
- Part Three, “Math/Measure Control Reference,” presents controls available in each of the multitude of the Math/Measure ‘processors’ available in X-Stream DSOs.

BLANK PAGE

ABOUT AUTOMATION

This section presents an overview of the Automation capabilities of X-Stream DSOs along with an overview of the technology itself.

CHAPTER ONE: *Overview*

In this chapter learn about

- ***Microsoft's COM-based "OLE Automation"***
- ***How to create simple applications in Visual Basic to control the instrument***
- ***How to use X-Stream Browser to interactively control the instrument***

OVERVIEW OF AUTOMATION

In addition to supporting the familiar ASCII-based remote commands that have been used to control all LeCroy DSOs for many years, all of the Windows-based “X-Stream” instruments fully support control by Automation interfaces based on Microsoft’s Component Object Model (COM). Using COM, the controlling application can run directly on the instrument without requiring an external controller; or, alternatively, it can run using Microsoft’s distributed COM standard (DCOM) on a networked computer.

Standards

Automation (formerly referred to as “OLE Automation”) is a Microsoft technology that is primarily used to enable cross-application macro programming. It is based upon the Component Object Model (COM), which is similar in nature to CORBA, which is more commonly found in the UNIX world.

An application that “exposes Automation Objects” is referred to as an “Automation Server.” Automation Objects expose “Automation Interfaces” to the controlling “Automation Client.” The oscilloscope application on LeCroy Windows-based oscilloscopes is an Automation Server, and can be controlled by Automation Clients. This manual describes the Automation objects and interfaces exposed by the oscilloscope application in detail.

It is important to note that Automation itself is not language dependent; it can be used from any programming language that supports COM. This manual, however, concentrates mainly on the use of Automation from the Visual Basic Script (VBScript) language for several reasons, including the fact that it is one of the easiest to use. Also, it is the language that X-Stream instruments use for setup files (more on this later). In addition, the VBScript interpreter is installed by default on all X-Stream instruments and, therefore, is available without your having to purchase any additional software.

Compatibility with Other LeCroy Scopes

Throughout LeCroy's history, we have striven to maximize compatibility, and this policy remains in force. However, due to the fact that the underlying technologies used by Automation require the 32-bit Windows operating system, and that this system is available only on our X-Stream instruments, Automation is not available on the older LeCroy oscilloscope families.

Automation and IEEE 488.2 Remote Control – How Do They Compare ?

Automation does not replace the IEEE 488.2-based remote command set, which is also supported by X-Stream instruments (and will continue to be). Instead, it augments it and allows a new class of application to be created that can run on the DSO itself.

Automation, however, can be considered as the “Native Language,” or “Mother Tongue” of X-Stream instruments. All of the instrument’s controls and features are available to the Automation Client.

Automation controls generally are more granular than 488.2 remote commands. That is, many 488.2 remote commands set more than one control at the same time; whereas, via Automation, this is not the case.

The following table summarizes the differences between the two remote control possibilities:

	IEEE 488.2 Control	Automation Control
Physical Transport	GPIB, TCP/IP over Ethernet	Inter-process using COM, inter-PC using DCOM (TCP/IP)
Textual parsing of instrument responses required	Yes. All instrument responses need 'parsing' to extract useful information.	No. Each element in the Automation hierarchy appears as a "variable" to the Automation client.
Compatibility with previous LeCroy DSOs	Very good. In most cases remote control applications written for older DSOs will work without modification.	None. Automation is a new standard first introduced with LeCroy's X-Stream DSOs.
Ability to control the oscilloscope application from "inside the box"	Yes, by using the TCP/IP (VICP) protocol to talk to the "local host"	Yes, natively
Ease of Use	Not trivial. It's easier using a tool such as ActiveDSO* that hides some of the complexities.	Very easy to use from scripting languages and office productivity tools
Format of Waveform results	Binary or ASCII. Both require parsing before use.	Waveforms are presented as arrays of floating point values.
Control from MS Office suite	Possible via ActiveDSO* utility	Yes, natively (see examples later in this manual)

* ActiveDSO is a Active-X based driver for LeCroy oscilloscopes

General Characteristics

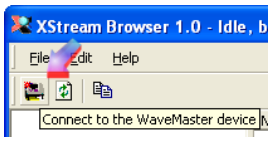
- When an application is running locally on the instrument and requests a connection to the DSO via Automation (for example, by using **CreateObject("LeCroy.XStreamDSO")** from Visual Basic), one of two things will happen. If the X-Stream DSO application is already running, the object returned will be a "pointer" to the running application. If the DSO application is not running, it will be started. It is not possible to run two simultaneous instances of the X-Stream DSO application.
- More than one simultaneous connection to the instrument via Automation will be accepted, but simultaneous connections are not recommended.
- When the final client has been disconnected from the instrument (server), the X-Stream DSO application will remain running and will accept further client connections.
- Operations that cause Modal Dialogs to appear in the instrument's display will, by default, disrupt access from Automation. This behavior can be changed using the **app.SystemControl.ModalDialogTimeout** and **app.SystemControl.EnableMessageBox** controls. Refer to the description of each of these controls in the reference section for more information.
- The instrument application can be minimized in order to allow the controlling application to take over the display and touch panel by means of the **app.Minimize** control. It can also be resized and repositioned on the display by means of the **app.Top**, **app.Left**, **app.Bottom**, **app.Right** controls.

INTRODUCTION TO THE X-STREAM BROWSER

The easiest way to get up and running with Automation, and also to visualize the “X-Stream Object Model” is to use the X-Stream browser tool, which is pre-installed on all instruments.

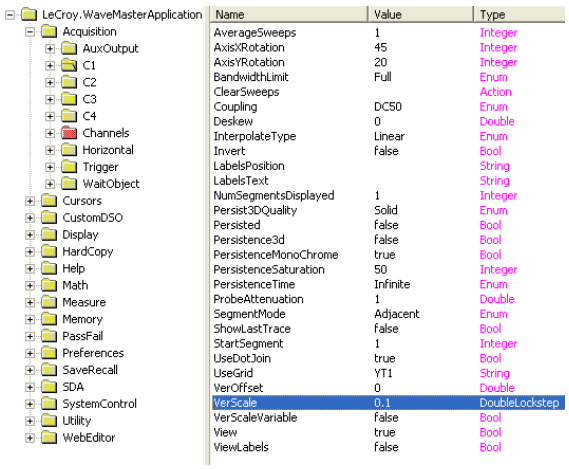
To launch the tool, first minimize the instrument application (**File->Minimize**), then double-click on the X-Stream Browser icon on the desktop:

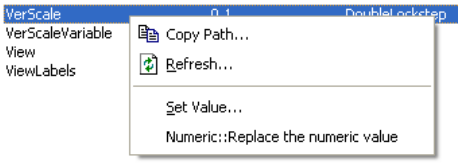
To connect to the running DSO application, click the **Connect** toolbar icon:

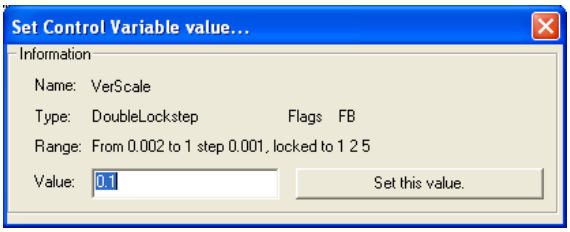


Upon connection, the root of the object hierarchy is shown in a layout similar to that presented when a file system is browsed using Windows Explorer.

As a quick demonstration of how the X-Stream Browser can be used, open the **Acquisition** folder and then click on the **C1** folder. Find the item **VerScale** (Volts/Div) in the right-hand window and right-click on it, then select the **Set Value** menu option. Use the Copy Path… option to easily copy to the control variable’s “path”.

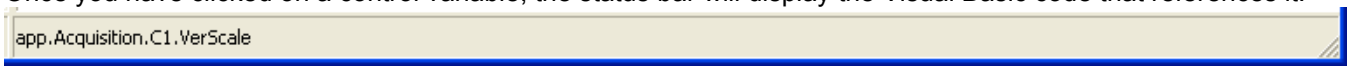






Enter a new value for **VerScale** and click the **Set this value** button. Restore the DSO’s X-Stream window and note that the V/Div value for C1 should have changed to reflect the entered value.

Once you have clicked on a control variable, the status bar will display the Visual Basic code that references it:



Most of the “folders” in the tree hierarchy in X-Stream Browser areyellow; these correspond to **Objects.** The object hierarchy is tiered; for example, the Acquisition is comprised of a variety of objects, including C1, C2, Horizontal, etc. Some Folders are pink; these are **Collections**. The subfolders of collections are referenced by indexing the collection name with the subfolder name. Lastly, the “Results” folders, which are dark gray. Results folders are objects, but contain items that are slightly different in nature to items in other folders.

The items in the right-hand window are referred to as **Control Variables**. These are explained in further detail in the section “Control Variables Explained” later in this chapter.

STEP-BY-STEP INTRODUCTION TO AUTOMATION USING VBScript

This section of the manual presents a walk-through of how to create a simple remote control application, which will run on the instrument, from scratch. It doesn't rely upon any 3rd-party development tools, since it uses Windows' built-in text editor (Notepad) and the Visual Basic Script interpreter (VBScript), which is also installed on all instruments.

1. Use the **File→Windowed** menu option to place the DSO application into windowed mode. This allows the windows start-bar to be accessed.
2. Open Windows **Notepad** via Start->Programs->Accessories->Notepad
   a. Write the following text into the editor:

```
Set app = CreateObject("LeCroy.XStreamDSO")
app.AutoSetup
app.Display.GridMode = "Quad"
myVerScale = app.Acquisition.C1.VerScale
MsgBox myVerScale
```

3. Save the file to drive **D:** and name it **Exercise1.vbs**. Leave Notepad open, we'll need it again.
4. Open Windows Explorer, via **Start→Programs→Accessories→Windows Explorer**.
5. Navigate to drive **D:** and double-click on **Exercise1.vbs**.
6. That's it. If these steps were followed correctly, you should hear relays clicking while the scope performs an auto-setup operation and enters its quad-grid display mode.

So, what did this "program" actually do?

- The **CreateObject** statement.

  ```
  Set app = CreateObject("LeCroy.XStreamDSO")
  ```

  CreateObject is the Visual Basic function that creates an instance of a COM Server (a.k.a. ActiveX Control). The argument **"LeCroy.XStreamDSO"** refers to our DSO application. Once it has instantiated (connected to) our DSO application we need some kind of 'handle' (pointer) to it so that we can use it later to communicate with the instrument. **CreateObject** returns a handle to us, which we store in the **app** variable.

  NOTE: Only a single instance of the X-Stream DSO software can run on a system at one time. If the DSO software is already running when CreateObject is called, a handle to that running instance is returned. If the DSO software is not running, it will be started.

- The **app.AutoSetup** statement.

  ```
  app.AutoSetup
  ```

  Using the **app** handle, this line of code calls the **AutoSetup** method, which performs the same task as the front-panel Auto-Setup button. Documentation for this method can be found later in the reference section.

- The **app.Display.GridMode = "Quad"** statement.

  ```
  app.Display.GridMode = "Quad"
  ```

  Using the **app** handle, this line of code sets the **GridMode** control of the **Display** system to the value "Quad". It's important to note that the controls are arranged in a hierarchy, with each 'level' of the hierarchy delimited with a decimal point (**.**).

- The **myVerScale = app.Acquisition.C1.VerScale** statement.

  ```
  myVerScale = app.Acquisition.C1.VerScale
  ```

  Instead of setting the value of a control, this line of code retrieves the current value of a control, in this case the Vertical Scale (Volts/Div) of Channel 1. The value returned is stored within the variable **myVerScale**.

NOTE: In Visual Basic Script it is not necessary to "Dimension" variables before using them (for example, using statements like "Dim myVerScale as Double").

- The **MsgBox myVerScale** statement

  ```
  MsgBox myVerScale
  ```

  This line of code does not communicate with the scope at all, but calls the standard Visual Basic Script function **MsgBox**. This function displays a dialog containing the value of the variable following "MsgBox". In our case the value of Channel 1's vertical scale, and waits for the **OK** button to be clicked.

  Documentation about the MsgBox function can be found in Microsoft's Visual Basic Scripting documentation at www.microsoft.com\scripting.

Another point that should be mentioned here is something that is used extensively in Setup files created by the instrument: the ability to use "abbreviations" to simplify programs. Following is an example in which a shorthand method is used to replace some rather long-winded code. It is also important to note that this also enhances performance. For example, the "lookup" of the object `app.Acquisition.C1` occurs only once in the modified code, but three times in the original code.

Instead of:

```
app.Acquisition.C1.VerScale = 0.5
app.Acquisition.C1.VerOffset = 0.1
app.Acquisition.C1.Coupling = "DC50"
```

The following may be used:

```
set myChannel1 = app.Acquisition.C1
myChannel1.VerScale = 0.5
myChannel1.VerOffset = 0.1
myChannel1.Coupling = "DC50"
```

WHERE IS AUTOMATION USED?

Automation is used in several places in the X-Stream based instrument.

- Instrument Setups (Panel Files)
- Custom Math/Measurements
- CustomDSO, User Interface customization
- Control from external applications (COM/DCOM)

Each of these uses is described in more detail in the following sections.

SETUPS (PANEL FILES) ARE PROGRAMS!

Setup files, which are used to save and recall the state of the instrument between runs, are traditionally binary files with an internal structure that is neither documented nor obvious to the user.

In X-Stream DSOs, however, this is no longer the case. Setups are ASCII text files that contain a complete Visual Basic Script "program" that, when "executed," will restore the instrument to a previously recorded state. **In effect, each time a panel is saved, the instrument effectively writes you a program that, when executed, returns the instrument to the saved state.**

Due to the fact that these setups are already programs, they are a great way to get started quickly with Automation. As an example, try saving a setup into a file and examine it using a text-editor, as follows:

1. Touch **File** in the menu bar, then **Save Setup** from the drop-down menu.
2. Touch the **Browse** button and specify drive **D:\Setups** as the location to save the .lss (LeCroy setup script) file. (This is the factory default directory for setup files)
3. Touch **Save Now!** 

4. Minimize the application using the **Minimize** option from the **File** menu
5. Open Microsoft's Notepad application from the **Accessories** program folder (**Start→Programs→Accessories**).
6. Open the file saved above. You will see a Visual Basic Script program that begins like this:

```
' XStreamDSO ConfigurationVBScript ...
' LECROY,WM8300,WM000001,0.0.0
' Thursday, February 20, 2003 11:26:55 AM

On Error Resume Next
set XStreamDSO = CreateObject("LeCroy.XStreamDSO")
XStreamDSO.RecallingSetup = True

' AladdinPersona ...
XStreamDSO.HideClock = False
XStreamDSO.TouchScreenEnable = True

…
```

Since the entire state of the instrument, including all controls for all installed software options, is saved, this panel may look fairly complex. But don't let this fool you; the basic concept is, in fact, fairly simple.

As a quick example of how setups can be used as the starting point for controlling applications, scroll down to the end of the file and add the following code (shown in bold-type) to the file.

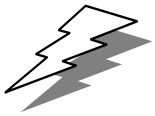
```
' Place any custom VBScript code after this point
'
' Perform an Auto Setup
XStreamDSO.AutoSetup
```

Add this code

When this setup is recalled, the complete state of the instrument will be restored, followed by an Auto-Setup operation.

Obviously this is a fairly trivial “application,” but it is easy to imagine how automated testing could be performed with the introduction of loops and conditional execution.

NOTE: Setup files stored by the instrument have file extension “.lss” (LeCroy Setup Script). These files are syntactically identical to Microsoft Visual Basic Script (VBScript) files, which have a “.vbs” extension.



TIP: A simple alternative to recalling the panel into the instrument is to execute it, either by double-clicking on the .lss file in Windows Explorer, or by executing it from the command line.

CUSTOM MATH AND MEASUREMENTS

Custom Math and Measurements can be coded using VBScript in instruments equipped with the XDEV and/or XMAP software options. Using Automation control of the instrument, decisions can be made during custom processing that reconfigure the DSO.

When you are developing custom processing routines using the reference section of this manual, **app.Acquisition.Cx.Out.Result** may be used as a comprehensive reference to the **Result Object**, which is used to describe waveform data (InResult, InResult1, InResult2, OutResult).

For more detail about this capability, see the “Customization” section of the on-line Help Manual.

CustomDSO

CustomDSO enables customization of the instrument’s UI in instruments equipped with the XDEV and/or XMAP options. Two modes of operation are supported: Basic mode and Plug-in mode.

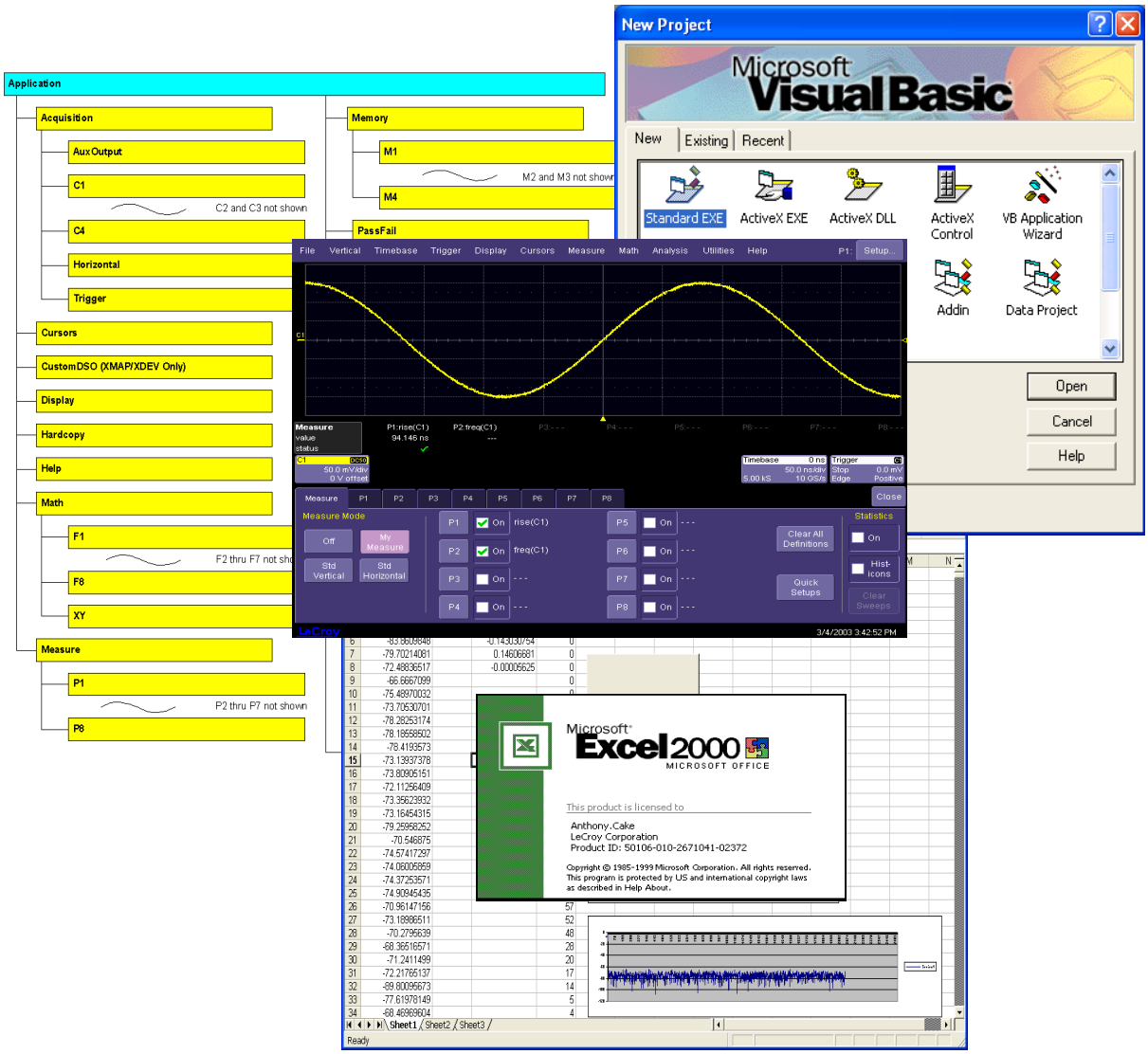
In Basic mode a Visual Basic Script (VBScript) program can be assigned to each of 8 buttons that can, optionally, appear at the bottom of the instrument’s display. By means of Automation, each of these may further reconfigure all 8 buttons, which would allow simple menu hierarchies to be generated.

In Plug-in mode an ActiveX control, created in any of a number of programming languages, can be inserted into the instrument’s menu system. Once “embedded,” this Plug-in can take full control of the DSO using Automation.

Full documentation on CustomDSO is available in the CustomDSO section of the on-line Help Manual.

Control from External Applications

Control of an X-Stream based instrument by Automation is possible from most modern programming languages (interpreted and/or compiled), and also from the “macro” capability of office productivity suites such as Microsoft Office.



From Visual Basic

From Visual Basic the **CreateObject** method is used to create the connection to an instrument by Automation.

The following code example creates this connection and sets up some of the instrument's controls:

```
' Connect to the X-Stream DSO
Dim app as Object
Set app = CreateObject("LeCroy.XStreamDSO")

' Setup Vertical and Horizontal settings
app.Acquisition.C1.VerScale = 0.5
app.Acquisition.C1.VerOffset = 0.25

app.Acquisition.Horizontal.HorScale = 0.000001

' Disconnect from the DSO
Set app = Nothing
```

From MATLAB

The syntax for controlling the scope MATLAB uses the **actxserver** keyword to connect to the instrument.

The following code example creates this connection, enables variable vertical scale, reads the vertical scale value for C1, and increases it by a factor of 1.5, arms the scope, waits for the acquisition to be complete, and then reads out the value and mean of parameter P1.

```
% Connect to oscilloscope via DCOM at IP address 172.28.9.31
DSO = actxserver('LeCroy.XStreamDSO', '127.0.0.1')

% Readback instrument ID
get(DSO.Item('InstrumentID'),'Value')

% Create references to the Acquisition, C1, Measure and P1 objects for convenience
acq = DSO.Object.Item('Acquisition')
c1 = acq.Object.Item('C1')
meas = DSO.Object.Item('Measure')
p1 = meas.Object.Item('P1')

% Enable variable vertical scale for channel 1
set(c1,'VerScaleVariable',-1)

% Read the vertical scale for C1 and increase it by a factor of 1.5, re-read and display new scale
verscale = get(c1.Item('VerScale'), 'Value')
set(c1, 'VerScale', 1.5 * verscale)
verscale = get(c1.Item('VerScale'), 'Value')

% Invoke the Acquire method, and force a trigger if the acquisition isn't complete within 3 seconds.
invoke(acq, 'Acquire', 3, 1)

% Readout the value and mean of measurement P1 (requires P1 to be setup and displayed)
p1_val= get(p1.Out.Result,'Value')
p1_mean = get(p1.Object.Item('Mean').Result,'Value')
```

Creating references to objects at each level in the oscilloscope application's object hierarchy is highly recommended in order to keep the code cleaner and easier to debug.

NOTE: Don't confuse the control of the instrument from MATLAB (MATLAB "drives") with the use of MATLAB from within a custom processing function (the instrument "drives").

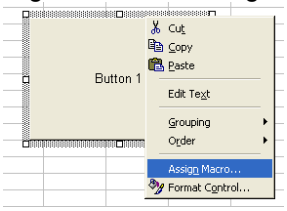
From MS Office (Excel)

Using Automation, control of the instrument from Microsoft Excel is very similar to control from Visual Basic. This is because the “macro language” used in the office suite is Visual Basic for Applications, a lightweight version of Visual Basic.

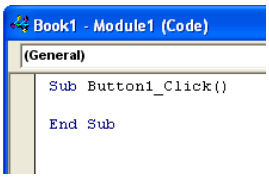
The following example shows how to add a button to an Excel spreadsheet that connects to, and controls, the instrument on which Excel is running. Note that this example was generated using Excel 2000, other versions of Excel support similar functionality, but the specific sequence of commands may be slightly different:

1. Enable the Forms toolbar using the **View→Toolbars→Forms** menu option.

2. Click the button icon  on the toolbar and draw a button on the spreadsheet. The button will be labeled “Button 1” by default.
3. Right-click on the edge of the button and select **Assign Macro** from the drop-down menu.



4. Select the **New** button from the **Assign Macro** dialog; the macro editor will appear:



5. Type the following code into the subroutine:

```
Set app = CreateObject("LeCroy.XStreamDSO")
app.AutoSetup
Set app = Nothing
```

Clicking on the newly created button will now execute this code segment, which connects to the DSO and performs an Auto Setup.

CONTROL VARIABLES EXPLAINED

Traditionally, properties presented to an Automation Client are simple “variables” with types such as Integer (int), String (BSTR), Floating Point (single, double), etc.

Control variables in X-Stream are an extension of the traditional Automation pattern, without affecting how they appear to most Automation clients (see section below on early/late bound clients).

As an example of what this enables, consider the following:

Take a control such as **VerScale** (Volts/Div), this may be set and queried in Visual Basic as follows:

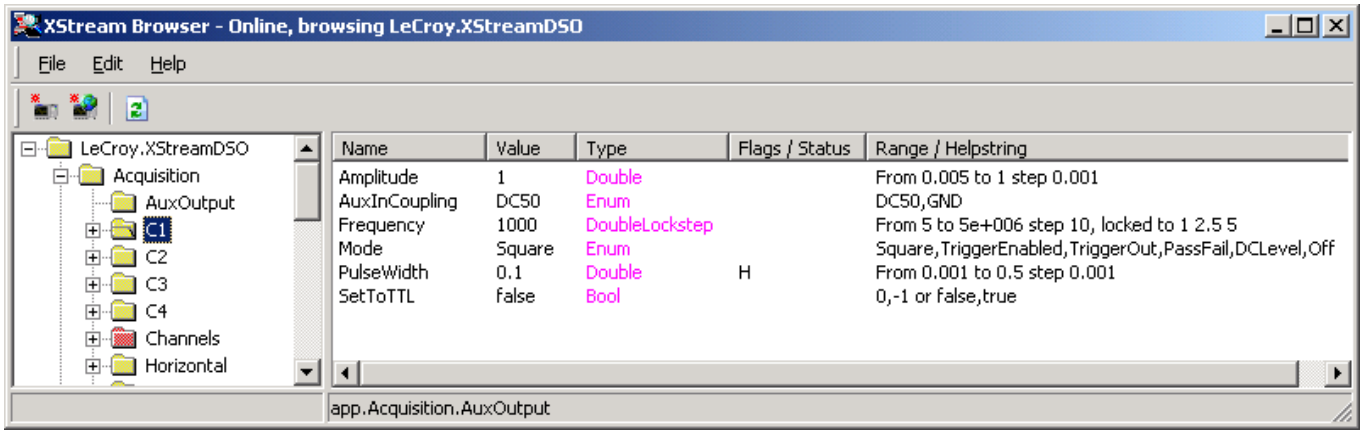
```
app.Acquisition.C1.VerScale = 0.5
CurrentValue = app.Acquisition.C1.VerScale
```

In addition, the following is supported:

```
minValue = app.Acquisition.C1.VerScale.GetMinValue
maxValue = app.Acquisition.C1.VerScale.GetMaxValue
```

This enables an Automation Client to not only set and query the current value of a control, but also to query its limits. This is useful in the generation of instrument-independent applications, or applications that present scope controls in a graphical user interface in which limits are required.

Various types of control variables are supported. The **Type** column in the X-Stream browser shows the control type:



The type designations are also given in the reference section of this manual, and are defined as follows:

Integer	32-bit Integer Value
Double	Double-precision floating point value
DoubleLockstep	Double-precision floating point value locked to a non-linear (e.g., 1, 2, 5) sequence.
Enum	List Value (e.g., “Orange,” “Apple,” “Pear”)
String	String value
Bool	Boolean Value { True, False }, { 0, -1 }
Action	Action (no arguments or value)

The properties and methods available for each control are type-specific. Listed below are the most commonly used:

TYPE	PROPERTIES
Integer	VARIANT Value Value(VARIANT) int GetAdaptedValue SetRequestedValue(int) int GetRequestedValue int GetDefaultValue int GetGrain int GetMax int GetMin Increment(int)
Double	VARIANT Value Value(VARIANT) double GetAdaptedValue SetRequestedValue(double) double GetRequestedValue double GetDefaultValue double GetGrainValue double GetMaxValue double GetMinValue Increment(int)
DoubleLockstep	See Double
Enum	VARIANT Value Value(VARIANT) int GetAdaptedValue SetRequestedValue(int) int GetRequestedValue int GetDefaultValue int GetMax int GetMin int GetNumberOfValueStates Increment(int) BSTR GetRangeStringScreen BSTR GetRangeStringRemote
String	VARIANT Value Value(VARIANT) int GetMaxLength BSTR GetRequestedValue BSTR GetAdaptedValue SetRequestedValue(BSTR)
Bool	VARIANT Value Value(VARIANT) BOOL GetAdaptedValue BOOL GetRequestedValue BOOL GetDefaultValue Set Clear
Action	ActNow

ACCESSING WAVEFORM/MEASUREMENT RESULTS

Waveforms

Waveform data is exposed by a 'Result' object, which appears at various places in the object hierarchy depending upon which waveform is to be accessed. Some examples follow:

```
app.Acquisition.C1.Out.Result
app.Math.F1.Out.Result
app.Memory.M1.Out.Result
```

Waveform data is exposed as a simple array, no deciphering of proprietary binary formats is performed, as was necessary in the past. An example of how it is used follows.

The example is coded as an Excel macro, and should be assigned to a button as described earlier. The macro reads the number of samples in the waveform and places it in cell B1 of the Excel spreadsheet. It then reads all available sample data values and copies them into cells in the first column of the spreadsheet (A1...Axx).

```
Sub Button1_Click()
    ' Connect to the DSO
    Set app = CreateObject("LeCroy.XStreamDSO")

    ' Query the number of samples in C1 and store in cell "B1"
    numSamples = app.Acquisition.C1.Out.Result.Samples
    Cells(1, 2).Value = numSamples

    ' Access the waveform data array, and fill the first column
    ' of the spreadsheet with it
    wave = app.Acquisition.C1.Out.Result.DataArray
    For i = 0 To numSamples - 1
        Cells(i + 1, 1).Value = wave(i)
    Next i
End Sub
```

NOTE: Ensure that the record length is < 64kSamples, since Excel has a limit on the number of rows in a spreadsheet. Ideally, you should start experimenting with short (500 point) records.

Measurements

Measurement results are read in the same way as Waveforms. The following example, when copied into an Excel macro, will enable Standard Vertical parameters. It will then transfer the eight parameter values into the spreadsheet (cells C1…C8):

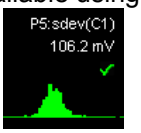
```
Sub Button1_Click()
    ' Connect to the DSO
    Set app = CreateObject("LeCroy.XStreamDSO")

    ' Enable Standard Vertical Parameters
    app.Measure.MeasureMode = "StdVertical"
```

```
    ' Transfer the 8 parameter values into the spreadsheet
    'Note: the number of configurable measurements your scope may be different
    Cells(1, 3).Value = app.Measure.P1.Out.Result.Value
    Cells(2, 3).Value = app.Measure.P2.Out.Result.Value
    Cells(3, 3).Value = app.Measure.P3.Out.Result.Value
    Cells(4, 3).Value = app.Measure.P4.Out.Result.Value
    Cells(5, 3).Value = app.Measure.P5.Out.Result.Value
    Cells(6, 3).Value = app.Measure.P6.Out.Result.Value
    Cells(7, 3).Value = app.Measure.P7.Out.Result.Value
    Cells(8, 3).Value = app.Measure.P8.Out.Result.Value

    Set app = Nothing
End Sub
```

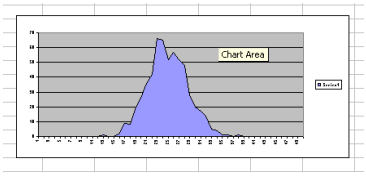
Statistics are also available for each parameter using two different kinds of calls:

```
result = app.Measure.P1.mean.Result.Value
result = app.Measure.P1.max.Result.Value
result = app.Measure.P1.min.Result.Value
result = app.Measure.P1.num.Result.Value
result = app.Measure.P1.sdev.Result.Value

result = app.Measure.P1.Statistics("mean").Result.Value
result = app.Measure.P1.Statistics("max").Result.Value
result = app.Measure.P1.Statistics("min").Result.Value
result = app.Measure.P1.Statistics("num").Result.Value
result = app.Measure.P1.Statistics("sdev").Result.Value
```

In addition, the data used to display the Histicon is available using the “histo” statistic:




```
Sub Button1_Click()
    Set app = CreateObject("LeCroy.XStreamDSO")
    bins = app.Measure.P5.Statistics("histo").Result.BinPopulations
    numBins = app.Measure.P5.Statistics("histo").Result.bins
    For i = 0 To numBins - 1
        Cells(i + 1, 4).Value = bins(i)
    Next I

    Set App = Nothing
End Sub
```



Result Status

The waveform result object described above includes a status property (bit-field) that reflects the current status of the trace. This includes both 'warning' and 'error' conditions, as described below.

Description	Value	Bit #
LEC_Valid	0x0	N/A
LEC_Invalid	0x0000000000001	0
LEC_Overflow	0x0000000000002	1
LEC_Underflow	0x0000000000004	2
LEC_ContainsUndefinedValues	0x0000000000008	3
LEC_LessThan	0x0000000000010	4
LEC_GreaterThan	0x0000000000020	5
LEC_NotAPulse	0x0000000000040	6
LEC_NotCyclic	0x0000000000080	7
LEC_Averaged	0x0000000000100	8
LEC_UnlockedPLL	0x0000000000200	9
LEC_OtherError	0x0000000000400	10
LEC_OtherWarning	0x0000000000800	11
LEC_OtherInfo	0x0000000001000	12
LEC_InputsIncompatible	0x0000100000000	28
LEC_AlgorithmLimitsReached	0x0000200000000	29
LEC_BadDefinition	0x0000400000000	30
LEC_TooFewData	0x0000800000000	31
LEC_TooManyData	0x0001000000000	32
LEC_UniformHorizIntervalRequired	0x0002000000000	33
LEC_BadUnits	0x0004000000000	34
LEC_DataRangeTooLow	0x0008000000000	35
LEC_DataUndersampled	0x0010000000000	36
LEC_PoorStatistics	0x0020000000000	37
LEC_SlowTransitionTime	0x0040000000000	38
LEC_DataResampled	0x0080000000000	39
LEC_DataInterpolated	0x0100000000000	40
LEC_MeasurementScaleImprecise	0x0200000000000	41
LEC_NoDataAvailable	0x0400000000000	42
LEC_SomeCummulatedResultsInvalid	0x0800000000000	43
LEC_InsufficientMemory	0x1000000000000	44
LEC_ChannelNotActive	0x2000000000000	45
LEC_UseStatusDescription	0x4000000000000	46

SYNCHRONIZATION

Synchronization or, more specifically, knowing when to read results, is critical when working with a digital oscilloscope by remote control (it is just as important by IEEE488.2 control as by Automation). This is especially true when working with an oscilloscope that uses a multithreaded architecture.

A classic problem seen in the majority of custom applications that control LeCroy (or other) DSOs is that the scope is left to free-run in Auto-trigger mode while simultaneously (and asynchronously) results are queried.

While working with the instrument via the Automation interface, there are a few techniques that can be used to guarantee the synchronization and consistency of results, whether they be waveform or parameter measurements.

The following example demonstrates a useful technique for ensuring synchronization. This example runs as an Excel macro:

```
Sub Button1_Click()
    ' Connect to the DSO
    Set app = CreateObject("LeCroy.XStreamDSO")

    ' Enable Standard Vertical parameters
    app.Measure.MeasureMode = "StdVertical"

    ' Stop the free-running trigger and take a single acquisition
    ' Use a 10 second timeout in the case that the acquisition is not complete.
    app.Acquisition.TriggerMode = "Stopped"
    app.Acquisition.Acquire (10, True)

    ' Read the first parameter value and transfer into the spreadsheet
    Cells(1, 3).Value = app.Measure.P1.Out.Result.Value

End Sub
```

The **Acquire** method arms the acquisition system and waits for a user-specified time for a trigger. The second argument, a Boolean, specifies whether or not to force a trigger before returning if a trigger doesn't arrive within the allotted time period. The method also returns a Boolean value signifying whether or not a trigger arrived. See the reference section for more information on this useful method.

Another scenario where synchronization is necessary is between changing settings and reading results, even when no acquisition took place. For this the **WaitUntilIdle** method is used. This method is "blocking" and will not return control until the setup request has completed.

*Note that the **Acquire** method is equivalent to setting the Trigger Mode to "Single", then executing **WaitUntilIdle.***

An example of **WaitUntilIdle** usage follows. Note that WaitUnilIdle is not for use when in NORMAL or AUTO trigger mode.

```
Sub Button1_Click()
    ' Connect to the DSO
    Set app = CreateObject("LeCroy.XStreamDSO")

    ' Enable Standard Vertical parameters
    app.Measure.MeasureMode = "StdVertical"

    ' Wait for the change to take place for a max. of 5 seconds
    app.WaitUntilIdle(5)

    ' Read the value of measurement P1 (pkpk) and transfer into the spreadsheet
    Cells(1, 2).Value = app.Measure.P1.Out.Result.Value

    ' Enable Standard Horizontal parameters
    app.Measure.MeasureMode = "StdHorizontal"

    ' Wait for the change to take place for a max. of 5 seconds
    app.WaitUntilIdle(5)

    ' Read the value of measurement P1 (rise time) and transfer into the
spreadsheet
    Cells(1, 3).Value = app.Measure.P1.Out.Result.Value
End Sub
```

NOTE: In almost all remote control applications, it is HIGHLY RECOMMENDED that you STOP acquisitions before accessing result data. Most remote control problems are caused by failure to follow this practice.

GOOD PRACTICES

- Using the **app.SetToDefaultSetup** action, restore the instrument to its default state before setting the controls required by an application. This eliminates any dependency on the previous configuration of the instrument. LeCroy strives to ensure that the default state of the instrument is constant from one software release to the next.
- Synchronization is an important concept that needs to be understood before you work with an X-Stream DSO via Automation. Attempting to read results while acquisitions are in progress could cause inconsistent results.
- Use the X-Stream Browser while developing Automation applications. This tool is guaranteed to show the up-to-date status of the Automation hierarchy since it retrieves it from a running instrument. It is also a very quick and easy way to exercise controls in real-time without your having to write a single line of code.
- When using a result object, verify that the status is valid to ensure that the acquisition and/or processing was valid.

EXAMPLES

Following are fairly complete examples of automating an X-Stream DSO, including configuration, acquisition, and reading results. Examples are given both as Excel macros, and as Visual Basic Scripts, which can run without Excel being loaded on the instrument.

Example 1: Excel Macro to Perform FFT of C1

```
Sub Button1_Click()
    ' Connect to the DSO
    Set app = CreateObject("LeCroy.XStreamDSO")

    ' Restore the instrument to its default state
    app.SetToDefaultSetup

    ' Stop acquisitions during setup
    app.Acquisition.TriggerMode = "Stopped"

    ' Turn C2 off (default state leaves C1 and C2 On)
    app.Acquisition.C2.View = False

    ' Configure F1=FFT(C1), using a Blackman-Harris filter
    app.Math.F1.View = True
    app.Math.F1.Source1 = "C1"
    app.Math.F1.Operator1 = "FFT"
    app.Math.F1.Operator1Setup.Window = "BlackmanHarris"

    ' Take a single acquisition, force after 2 seconds if it doesn't trigger
    app.Acquisition.Acquire 2, True

    ' Read out the FFT
    ' Query the number of samples in F1 and store in cell "B1"
    numSamples = app.Math.F1.Out.Result.Samples
    Cells(1, 2).Value = numSamples

    ' Access the waveform data array, and fill the first column
```

```
    ' of the spreadsheet with it
    wave = app.Math.F1.Out.Result.DataArray
    For i = 0 To numSamples - 1
        Cells(i + 1, 1).Value = wave(i)
    Next

End Sub
```

Example 2: VBScript Program to Perform FFT of C1 and Store Results in Text File

This example requires no additional software to be installed on the instrument, since it relies upon the built-in Visual Basic Script interpreter. The example is very similar to the previous Excel example, the most notable difference being the use of a standard system ActiveX control, **"Scripting.FileSystemObject"**, to enable the creation of files containing waveform data in ASCII format.

```
' VBScript example
' Configure the DSO to perform an FFT on Channel 1 and store
' the resulting data in a text file in ASCII format

' Connect to the DSO
Set app = CreateObject("LeCroy.XStreamDSO")

' Restore the instrument to its default state
app.SetToDefaultSetup

' Stop acquisitions during setup
app.Acquisition.TriggerMode = "Stopped"

' Turn C2 off (default state leaves C1 and C2 On)
app.Acquisition.C2.View = False

' Configure F1=FFT(C1), using a Blackman-Harris filter
app.Math.F1.View = True
app.Math.F1.Source1 = "C1"
app.Math.F1.Operator1 = "FFT"
app.Math.F1.Operator1Setup.Window = "BlackmanHarris"

' Take a single acquisition, force after 2 seconds if it doesn't trigger
app.Acquisition.Acquire 2, True

' Readout the FFT
numSamples = app.Math.F1.Out.Result.Samples

Set fso = CreateObject("Scripting.FileSystemObject")
Set MyFile= fso.CreateTextFile("c:\XStreamFFT.txt", True)

' Write the FFT power spectrum into the file, sample by sample
wave = app.Math.F1.Out.Result.DataArray
For i = 0 To numSamples - 1
    MyFile.WriteLine(wave(i))
Next

' Clean up
MyFile.Close
Set fso = Nothing
Set app = Nothing
```

Example 3: Script to Measure the Rise Time of the Signal on C1 and Display It in a Popup Window

This example configures the DSO to measure the rise time of the signal on C2, take a single acquisition, and present the results in a popup dialog. The example requires no additional software to be installed on the instrument, since it relies on the built-in Visual Basic Script interpreter.

```
' VBScript example
' Configure the DSO to measure the rise time of the signal
' on Channel 1 and display it in a popup message box.

' Connect to the DSO
Set app = CreateObject("LeCroy.XStreamDSO")

' Restore the instrument to its default state
app.SetToDefaultSetup

' Stop acquisitions during setup
app.Acquisition.TriggerMode = "Stopped"

' Turn C2 off (default state leaves C1 and C2 On)
app.Acquisition.C2.View = False

' Configure P1=rise(C1)
app.Measure.MeasureMode = "MyMeasure"
app.Measure.P1.View = True
app.Measure.P1.ParamEngine = "rise"

' Take a single acquisition, force after 2 seconds if it doesn't trigger
app.Acquisition.Acquire 2, True

' Present the rise time in a popup message box
MsgBox app.Measure.P1.Out.Result.Value & "s", vbOKOnly, "Rise time of C1"

' Clean up
Set app = Nothing
```

EARLY AND LATE BINDING

The COM standard on which Automation is built supports two kinds of “binding” between client and server: early (static), and late (dynamic, dispatch). Static binding usually involves a type library and is used primarily by compiled languages such as C++. In this case, function entry points are resolved at compile time. Dynamic binding (also known as late binding) involves resolving method and property calls at run time, as opposed to compile time.

The Automation interfaces in X-Stream based DSOs use primarily the latter: Dynamic binding. From many programming languages (VB, VBScript, etc.) this is transparent. But when you are developing applications in C++, which does not provide late-binding natively, the use of a “helper” class is required. This is demonstrated below:

```
#include "stdafx.h"

#include "AtlBase.h"
CComModule _Module;
#include "AtlCom.h"

CComPtr<IDispatch> spDso;
CComDispatchDriver ddDso;       // dispatch ptr. to root of object model (app)

int main(int argc, char* argv[])
{
      printf("Hello X-Stream World!\n");

      ::CoInitialize(NULL);

      HRESULT hr = spDso.CoCreateInstance(L"LeCroy.XStreamDSO");
      if(SUCCEEDED(hr))
      {
              ddDso = spDso;

             // perform an Auto-Setup (app.Autosetup)
                hr = ddDso.Invoke0(L"AutoSetup");

             // retrieve a Dispatch ptr. to the app.Display object
               CComVariant displayPtr;
               hr = ddDso.GetPropertyByName(L"Display", &displayPtr);
                CComDispatchDriver ddDisplay(displayPtr.pdispVal);

            // enter Dual-grid mode (app.Display.GridMode = "Dual")
               hr = ddDisplay.PutPropertyByName(L"GridMode", &CComVariant("Dual"));
      }

      return 0;
}
```

VBS REMOTE COMMAND

For users who wish to harness the power of Automation control of an instrument, but are currently using “traditional” remote commands via GPIB or the network (using the VICP protocol), there is a solution. This is primarily of interest in controlling the advanced features of X-Stream DSOs, which are not supported by a traditional remote command.

X-Stream instruments, in addition to supporting LeCroy’s standard remote command set, also support a new command/query called **VBS**[?]. This command may be used in traditional remote control applications to access Automation controls.

This example shows two methods for setting the V/Div of Channel 1, the former using a traditional remote command, **VDIV**, and the latter using an Automation control via the new remote command, **VBS**. These two commands are equivalent:

```
C1:VDIV 0.5

VBS 'app.Acquisition.C1.VerScale = 0.5'
```

In its query form the following are equivalent:

```
C1:VDIV?

VBS? 'return = app.Acquisition.C1.VerScale'
```

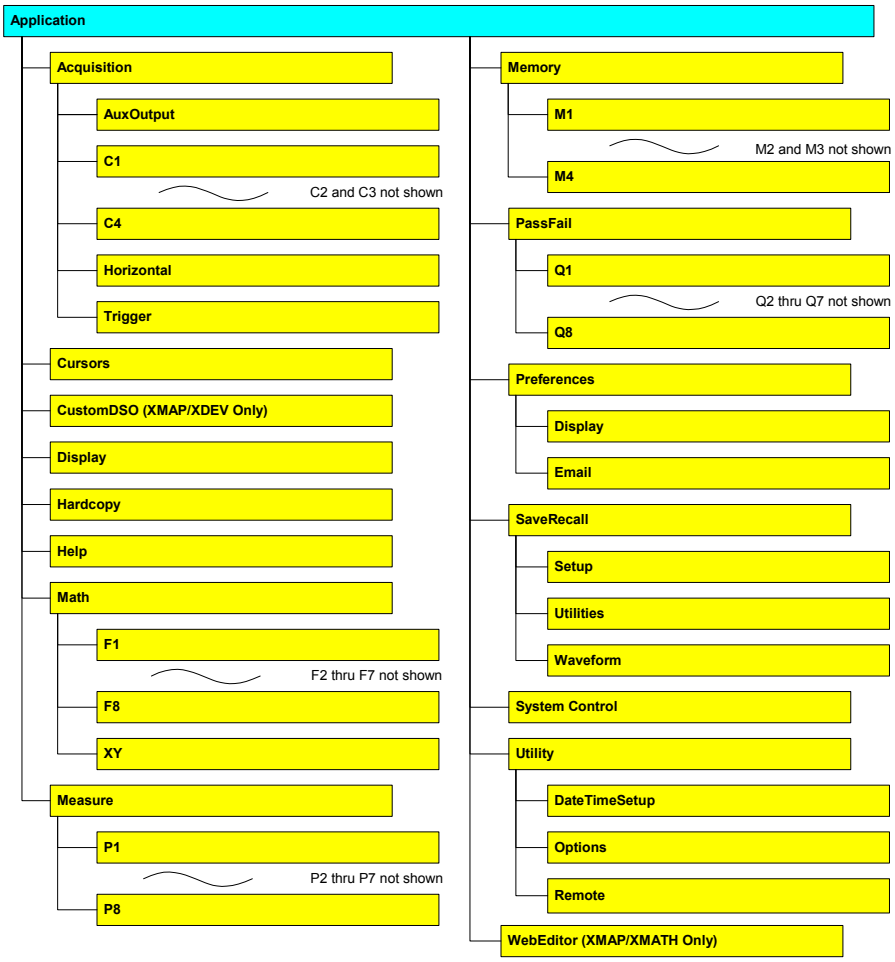
A couple of points to note here are that the **app** variable is pre-defined and refers to the root of the Automation hierarchy. Also note that for the query form the **return =** is important, it indicates which value you wish to return to the caller.

The VBS[?] Command/Query is documented in more detail in the *Remote Control Manual* for X-Stream DSOs.

X-STREAM DSO OBJECTS

The object hierarchy exposed by X-Stream based instruments is rooted at the Application object. This is the object returned when the **CreateObject("LeCroy.XStreamDSO")** method is executed in Visual Basic. All major instrument subsystems are available from this object, and many of these subsystems themselves are broken down further. Note that to simplify this figure only the first and last of the collections of Channels, Memories, Math, and Measurements are shown.

The reference section of this manual describes the controls presented by each of these objects.



NOTE: The root of the object hierarchies of some software options are not shown in this diagram.

ABOUT RESULTS

PART TWO: ***About Results***

In this chapter learn about

- ***X-Stream DSO's rich set of Result Interfaces***
- ***How to read waveforms (and other results) from the X-Stream DSO, and write them back into the instrument***

INTRODUCTION

This section presents details on the rich 'Result' interfaces of the X-Stream DSO. These interfaces contain more than just the basic results, such as waveform data and measurement values; it includes information about horizontal and vertical resolution, event times, number of sweeps, histogram peaks, etc.

The properties that are available for readout depend on the type of result. The 7 different Results Interfaces are:

- Digital
- Histogram
- Param
- Persist
- Table
- Waveform
- XY

The following section describes each interface, followed be a detailed explanation of each property.

Interface Descriptions

Digital

The Digital interface is found when using one of the mixed-signal oscilloscope options to generate digital (bi-state) waveforms. These waveforms are contained in the **Digitalx** trace name, where x = 1, 2, 3 etc. Each Digitalx trace is a bus that may contain multiple digital lines, with a maximum number dependent on the capabilities of the mixed-signal oscilloscope option. The key property in the Digital interface is the **DataArray**, which is a 2D array containing the value of each sample of each digital line in the bus. Example paths to Digital Result interfaces:

```
app.LogicAnalyzer.Digital1.Out.Result
app.LogicAnalyzer.Digital2.Out.Result
```

Histogram

The Histogram interface is used for all histogram traces. Histograms can be configured on most oscilloscopes as a math function (Fx). Histograms are also used in various analysis modes, especially on Serial Data Analyzers, where the HTIE trace is the source data for the algorithm to calculate Total jitter (Tj). The key property is **BinPopulations**, which is a 1D array of the bin populations. Example paths to HTIE Result interfaces:

```
app.Math.F1.Out.Result  (assuming that F1 is setup as a Histogram)
app.SDA.Htie.Out.Result
```

Param

The Parameter interface is used in the Measurement and Pass/Fail systems to hold both individual results and result arrays. The key properties for the Parameter interface are **Value** and **ValueArray**. Value holds the last measurement or P/F result, while ValueArray holds all measurements taken for the last acquisition. Example paths to Param Result interfaces:

```
app.Measure.P1.Out.Result
app.Measure.P1.Sdev.Result
app.Measure.Measure("P1").Out.Result
```

Persist

The Persist interface is most often found when using software options that create eye diagrams or other waveshapes that are the result of overlaying multiple waveforms. (Note that this does not apply to the Persistence display mode.) The key property in the Persist interface is the **DataArray**, which is a 2D array containing the population of each pixel of the persistance map. Example paths to Persist Result interfaces:

```
app.SDA.Eye.Out.Result
app.SDA.Eye2.Out.Result
```

Table

Table Results interfaces are used primarily used in conjunction with the various low-speed serial decoding options, such as CAN, I2C, SPI, UART, etc. When the table is displayed, the user has access to information about the messages decoded for the last acquisition. Retrieving data from the Table interface involves some knowledge of the table's structure, in terms of the columns that are displayed and the cell types. The **CellType** and **CallValue** are the key properties to use to retrieve the table's data. See the information in the **CellValue** property for more information. Example paths to Table Result interfaces:

```
app.SerialDecode.Decode1.Out.Result.
app.SerialDecode.Objects("Decode2").Out.Result
```

Waveform

Waveform Results interfaces can be found wherever there is basic Voltage vs. Time waveforms. Examples include channel (Cx), memory (Mx), math function (Fx) and zoom (Zx) traces, as well as others that are associated with specific analysis packages, such as the Power waveform used in the PMA2 package. In general, these waveforms are usual voltage vs time, unless a current probe or other method of changing the vertical units (e.g. the Rescale Math function) is employed. The key property in the Waveform interface is the **DataArray**, which is a 1D array containing the value of each sample in the waveform. Example paths to Waveform Result interfaces:

```
app.Acquisition.C1.Out.Result
app.Acquisition.Channels("C1").Out.Result
app.Math.F2.Out.Result
app.Zoom.Z3.Out.Result
app.SDA.Btub.out.Result
app.SDA.TIETrend.Out.Result
app.PMA.Pwr.Out.Result
```

(Note that traces like F2 and Z3 can hold waveforms like histograms, which do not use the Waveform interface.)

XY

The XY interface is used on XY waveforms. Users can create an XY waveform by selecting this option in the Display Setup… screen. XY waveforms allow you to see how the signal on one channel moves in relation to another. The key property is the DataArray, which is a 2D array that returns the voltages on the X and Y sources for each XY pair. Example path to the XY Result interface:

```
app.Math.XY.Out.Result
```

OVERVIEW OF RESULTS

Variable Types

The examples in this section use the following standards:

- **Integer** is 16 bit signed integer
- **Long** is 32 bit signed integer
- **Double** is 8byte floating point type
- **String** is an array of characters
- **Object** is a object with its own interface.

Where a variable is dimensioned without indicating a variable type, the variable is a variant.

Use of Variants

Several properties, most significantly the **DataArray** object, is a variant containing either a one or two dimensional array. Dimension the target variable as a variant type, and assign the DataArray property to it. See the DataArray property for examples of its use within each of the Result interfaces.

Using Variables to Reference Objects within the XStream Hierarchy

One way to increase the readability and simplicity of your source code is to use variables to reference objects that are at each level of the XStream Hierarchy. For example, here is the code to readout the Sweeps property for channel 1:

```
Dim numSweeps as Long
numSweeps = app.Acquisition.C1.Out.Result.Sweeps
```

If you are reading out more than just the Sweeps property, you might find it easier to read the following :

```
Dim C1Res as Object
Dim C1 as Object
Dim numSweeps as Long
Dim HorizontalOffset As Double
Dim HorizontalPerStep As Double
Dim VerticalOffset As Double
Dim VerticalPerStep As Double
Dim wform

C1 = app.Acquisition.C1
C1.AverageSweeps = 200
numSweeps = C1Res.Sweeps
HorizontalOffset = C1Res.HorizontalOffset
HorizontalPerStep = C1Res.HorizontalPerStep
VerticalOffset = C1Res.VerticalOffset
VerticalPerStep = C1Res. VerticalPerStep
```

The examples in this chapter do not make use of this approach in order to show the "full story", but it is recommended. The panel setup files make heavy use of using variables to objects, making them far more readable and editable:

```
Set Acquisition = XStreamDSO.Acquisition
Set C1 = Acquisition.C1
' C1 ...
C1.View = True
C1.UseGrid = "YT1"
C1.Persisted = False
C1.PersistenceSaturation = 50
C1.PersistenceMonoChrome = True
C1.Persistence3d = False
C1.ShowLastTrace = False
C1.PersistenceTime = "Infinite"
C1.Persist3DQuality = "Solid"
C1.AxisXRotation = 45
C1.AxisYRotation = 20
C1.LabelsText = ""
C1.LabelsPosition = ""
C1.ViewLabels = False
C1.SegmentMode = "Adjacent"
C1.ProbeAttenuation = 1
C1.VerScale = 0.05
C1.VerScaleVariable = False
C1.VerOffset = 0
C1.Coupling = "DC50"
C1.BandwidthLimit = "Full"
C1.AverageSweeps = 1
C1.InterpolateType = "Linear"
C1.Invert = False
C1.Deskew = 0
C1.NumSegmentsDisplayed = 1
C1.StartSegment = 1
```

DESCRIPTION OF RESULT PROPERTIES

Base

Applies to: Histogram

Description: Horizontal coordinate of the leftmost of the two most populated histogram peaks. It is equivalent to the paramater **hbase.**

Example:

```
Dim hbase as Double
hbase = app.Math.F1.Out.Result.Base
```

BinPopulations

Applies to: Histogram

Description: VARIANT array containing the populations of each bin. Bin 0 is the first bin. Index the array to retrieve the number of counts in a given bin.

Example:

```
'Read out BinPopulations by indexing the array using the
'FirstPopulatedBin and LastPopulatedBin properties as boundaries
'Also shows code for determining coordinate of bin centers

Const STARTROW = 5
Dim i As Integer
Dim BinPops
Dim FirstPopulatedBin, LastPopulatedBin As Integer
Dim OffsetAtLeftEdge As Double
Dim BinWidth As Double

BinPops = app.Math.F1.Out.Result.BinPopulations
FirstPopulatedBin = app.Math.F1.Out.Result.FirstPopulatedBin
LastPopulatedBin = app.Math.F1.Out.Result.LastPopulatedBin

OffsetAtLeftEdge = app.Math.F1.Out.Result.OffsetAtLeftEdge
BinWidth = app.Math.F1.Out.Result.BinWidth

For i = FirstPopulatedBin To LastPopulatedBin
    Cells(i - FirstPopulatedBin + STARTROW + 1, 7) = i
    'Calculate Bin Center
    Cells(i - FirstPopulatedBin + STARTROW + 1, 8) = OffsetAtLeftEdge +
(BinWidth / 2) + BinWidth * i
    Cells(i - FirstPopulatedBin + STARTROW + 1, 9) = BinPops(i)
Next
```

Bins

Applies to: Histogram

Description: Number of bins in the histogram.

Example:

```
Dim Bins as Integer
Bins = app.Math.F1.Out.Result.Bins
```

BinWidth

Applies to: Histogram

Description: Width of each bin in the histogram.

Example:

```
Dim BinWidth as double
BinWidth = app.Math.F1.Out.Result.BinWidth
```

BusName

Applies to: Digital

Description: Name of the bus, which can be configured via the user interface or by the automation property **app.LogicAnalyzer.Digital1.BusName**

Example:

```
Dim BusName As String
BusName = app.LogicAnalyzer.Digital1.Out.Result.BusName
```

CellType

Applies to: Table

Description: Datatype for the selected cell in a table.This is used in order to properly readout the CellValue. .

Arguments:

Row: 0-based index of the row containing the cell. Use the **Rows** property to determine the maximum number of rows.

Column: 0-based index of the column containing the cell. Use the **Columns** property to determine the maximum number of columns.

Result type: Enumerated, with the following definitions:

0: Variant
2: Param result interface
(other values exist, but do not apply to the Table interface)

Example:

```
'Returns CellType for  row 1, column 1
Dim CellType As Integer
CellType = app.SerialDecode.Decode1.Out.Result.CellType(1, 1)
```

CellValue

Applies to: Table

Description: The value for the selected cell in a table. The method for reading back CellValue depends on the cell's CellType property. For cells where CellType = 0 (variant), then CellValue contains the cell's value. For CellType = 2, CellValue becomes a Param result interface with its own value property. See the example for more information.

Arguments:

Row: 0-based index of the row containing the cell

Column: 0-based index of the column containing the cell

Example:

```
Dim CellValue

' When the CellType for the cell = 0, read out the value directly, as shown
below.
' This call often returns "Idx", which is the header for column 0
CellValue = app.SerialDecode.Decode1.Out.Result.CellValue(0, 0)

' The code below demonstrates how to readback the
' cell value when the CellType for the cell = 2:

Dim DecoderTableResult As Object
Set DecoderTableResult = app.SerialDecode.Decode1.Out.Result
Dim CellValueOb As Object
```

```
Set CellValueOb = DecoderTableResult.CellValue(1, 2)
CellValue = CellValueOb.Value
```

Scopes that have a serial decoder option include a script for outputting the table of decoded values. These can be found on the scope within the D:\Applications folder.

Columns

Applies to: Persist, Table

Description: Number of columns in the DataArray. Typically equal to 1000 for Persist waveforms. For Tables, the result can vary depending on the application.

Example:

```
Dim Columns As Integer

'Persist example
Columns = app.SDA.Eye.Out.Result.Columns

'Table example
Columns = app.SerialDecode.Decode1.Out.Result.Columns
```

DataArray

Applies to: Digital, Persist, Waveform, XY

General Description: Variant array containing data for a trace. The implementation depends on the type of waveform; see below for further details.

XY Interface:

Arguments (optional):

arrayValuesScaled as Boolean

Default value:TRUE, data is scaled

Determines whether returned values are scaled or raw. Passing TRUE indicates scaled data; FALSE indicates raw data. Scaled values are double precision; raw values are 16-bit signed integers. See the properties XVerticalPerStep, XVerticalResolution and XVerticalOffset properties (and the corresponding Y coordinate) for more information

numSamples as Long

Default value: -1, retrieve all data

Number of samples to retreived

startIndex as Long

Default value: 0, first sample

Index of the first sample to be retreived.

sparsingFactor as Long

Default value: 1, no sparsing

Determines the sparsing factor to use. For example, a sparsing factor of 5 means to retreive every 5th point. (Note: Use the math function called Sparse in order to configure sparsing offset)

Example:

```
'Reads values to use for DataArray arguments from a spreadsheet, and then
loops through
'the retrieved array, saving the data to the spreadsheet along with sample
index and sample time.
Const ROWOFFSET = 10
Const COLOFFSET = 7
```

```
Dim HorizontalPerStep As Double
Dim HorizontalOffset As Double
Dim ScaleArray As Boolean
Dim nSamples As Long
Dim startIndex As Long
Dim sparseFactor As Long
Dim XYwform

ScaleArray = Cells(9, 5)
nSamples = Cells(10, 5)
startIndex = Cells(11, 5)
sparseFactor = Cells(12, 5)
XYwform = app.Math.XY.Out.Result.DataArray '(ScaleArray, nSamples,
startIndex, sparseFactor)
HorizontalPerStep = app.Math.XY.Out.Result.HorizontalPerStep
HorizontalOffset = app.Math.XY.Out.Result.HorizontalOffset

For i = 0 To UBound(XYwform)
    For j = 0 To 1
        Cells(i + ROWOFFSET, COLOFFSET) = i
        'Calculate sample time
        Cells(i + ROWOFFSET, COLOFFSET + 1) = HorizontalPerStep * i +
HorizontalOffset 'Sample
        Cells(i + ROWOFFSET, COLOFFSET + 2) = XYwform(i, 0)   'X coordinate
        Cells(i + ROWOFFSET, COLOFFSET + 3) = XYwform(i, 1)   'Y coordinate
    Next j
Next i
```

Waveform Interface:

Arguments (optional):

arrayValuesScaled as Boolean

Default value:TRUE, data is scaled

Determines whether returned values are scaled or raw. Passing TRUE indicates scaled data; FALSE indicates raw data. Scaled values are double precision; raw values are 16-bit signed integers. (See the properties VerticalPerStep, VerticalResolution and VerticalOffset properties for more information)

numSamples as Long

Default value: -1, retrieve all data

Number of samples to retreived

startIndex as Long

Default value: 0, first sample

Index of the first sample to be retreived.

sparsingFactor as Long

Default value: 1, no sparsing

Determines the sparsing factor to use. For example, a sparsing factor of 5 means to retreive every 5th point. (Note: Use the math function called Sparse in order to configure sparsing offset)

Example:

```
'Reads values to use for DataArray arguments from a spreadsheet, and then
loops through
'the retrieved array, saving the data to the spreadsheet along with sample
index and sample time.
```

```
Const ROWOFFSET = 10
Const COLOFFSET = 7
Dim HorizontalPerStep As Double
Dim HorizontalOffset As Double
Dim ScaleArray As Boolean
Dim nSamples As Long
Dim startIndex As Long
Dim sparseFactor As Long
Dim wform

ScaleArray = Cells(9, 5)
nSamples = Cells(10, 5)
startIndex = Cells(11, 5)
sparseFactor = Cells(12, 5)

wform = app.Acquisition.C1.Out.Result.DataArray(ScaleArray, nSamples,
startIndex, sparseFactor)
HorizontalPerStep = app.Acquisition.C1.Out.Result.HorizontalPerStep
HorizontalOffset = app.Acquisition.C1.Out.Result.HorizontalOffset

For i = 0 To UBound(wform)
        Cells(i + ROWOFFSET, COLOFFSET) = i
        Cells(i + ROWOFFSET, COLOFFSET +1) = HorizontalPerStep * i +
HorizontalOffset
        Cells(i + ROWOFFSET, COLOFFSET+2) = wform(i)
Next i
```

Persist Interface:

Description:

Retrieves a 2D array containing hits in each cell of of the rectangle selected via the input arguments.

Arguments (optional):

numColumns as Long

Default value: -1, retrieve all columns

Number of columns to retrieve.

numRows as Long

Default value: -1, retrieve all rows

Number of rows to retrieve.

startColumn as Long

Default value: 0, first column

Index of the first column to retrieve. (0-based)

startRow as Long

Default value: 0, first row

Index of the first row to retrieve. (0-based)

Example:

```
'Reads values to use for DataArray arguments from a spreadsheet, and then
loops through
'the retrieved array, saving the data to the spreadsheet.

Const ROWOFFSET = 10
```

```
Const COLOFFSET = 7
Dim numColumns As Long
Dim numRows As Long
Dim startColumn As Long
Dim startRow As Long
Dim wform

numColumns = Cells(9, 5)
numRows = Cells(10, 5)
startColumn = Cells(11, 5)
startRow = Cells(12, 5)
wform = app.SDA.Eye.Out.Result.DataArray(numColumns, numRows, startColumn,
startRow)
For i = 0 To numRows - 1
    For j = 0 To numColumns - 1
        Cells(i + ROWOFFSET, j + COLOFFSET) = wform(i, j)
    Next j
Next i
```

Digital Interface:

Description:

Retrieves a 2D array containing the state of each sample on each line. The value 1 is returned if he sample was above the threshhold, 0 if below.

Arguments (optional):

numSamples as Long

Default value: -1, retrieve all data

Number of samples to retreived

numLines as Long

Default value: -1, retrieve all lines.

Number of digital lines to retrieve.

startIndex as Long

Default value: 0, first index

Index of the first sample to retrieve. (0-based)

startLine as Long

Default value: 0, first digital line

Index of the first digital line to retrieve (0-based). The user selects which lines to enable; only data from enabled lines are included in the DataArray

Example:

```
'Reads values to use for DataArray arguments from a spreadsheet, and then
loops through
'the retrieved array, saving the data to the spreadsheet.

Const ROWOFFSET = 10
Const COLOFFSET = 7
Dim nSamples As Long
Dim nLines As Long
Dim startIndex As Long
Dim startLine As Long
Dim wform
```

```
nSamples = Cells(9, 5)
nLines = Cells(10, 5)
startIndex = Cells(11, 5)
startLine = Cells(12, 5)

wform = app.LogicAnalyzer.Digital1.Out.Result.DataArray(nSamples, nLines,
startIndex, startLine)

'Saves the retrieved samples to a spreadsheet, with each digital line in a
column.
For j = 0 To UBound(wform, 2)    ' index out lines
    For i = 0 To UBound(wform, 1)  ' index out samples in line
        Cells(i + ROWOFFSET, j + COLOFFSET) = wform(i, j)
    Next
Next
```

ExtendedStatus

Applies to: All Result interfaces.

Description: Reserved for future use.

Example: N/A

FirstEventTime

Applies to: All Result interfaces.

Description: Absolute trigger time of the acquisition, or the time of the first trigger in an acquisition that utilizes multiple sweeps. Examples of multi-sweep acquisitions are Sequence Mode and when a source channel is configured for pre-processor averaging.

Times are returned encoded as a currency value (VT_CY) within a variant, this allows the use of the full 64-bit resolution of the timestamp value. Values are referenced to 1st Jan 2000, with 1ns resolution. Note that VT_CY values are stored as 64-bit (8 byte) two's complement integers, scaled by 10,000 to give a fixed-point number with 15 digits to the left of the decimal point, and 4 digits after. See the example for details on decoding **FirstEventTime** as well as other properties that return times such as **LastEventTime** and **UpdateTime.**

Example:

```
Dim FirstEventTime
Dim TimeString as String
FirstEventTime = app.Acquisition.C1.Out.Result.FirstEventTime
timeString = DecodeTimeProperty(FirstEventTime)
```

What follows is a function for decoding the FirstEventTime, LastEventTime and UpdateTime properties.

```
Function DecodeTimeProperty(ByVal EventTime) As String
'  Decodes result returned by FirstEventTime, LastEventTime and UpdateTime
properties

    Dim days, thedate, hours, thehour, minutes, theminute, seconds
    Dim outstring As String
    days = EventTime / (86400 * 100000#) '  use 1e5 since the data returned
is scaled by 10,000
    thedate = DateAdd("d", days, "1-Jan-00")
    hours = (days - Int(days)) * 24   'calculate remaining hours
    thehour = Int(hours)   'truncate to get hour in the day
    minutes = (hours - thehour) * 60 'calculate minutes since hour began
    theminute = Int(minutes)  'truncate to get minute value
    seconds = (minutes - theminute) * 60 'calculate number of seconds
    'Format output string
    DecodeTimeProperty = thedate & " " & thehour & ":" & theminute & ":" &
seconds
```

```
End Function
```

FirstPopulatedBin

Applies to: Histogram

Description: Index of the first populated bin. (The first bin of the histogram is bin 0).

Example:

```
Dim FirstPopulatedBin as Integer
FirstPopulatedBin = app.Math.F1.Out.Result.FirstPopulatedBin
```

HorizontalFrameStart

Applies to: Digital, Histogram, Persist, Waveform, XY

Description: Coordinate of the left edge of the graticule containing the trace relative to the trigger time, which is at time = 0.

Example:

```
Dim XFrameStart As Double
X  FrameStart = app.Acquisition.C1.Out.Result.HorizontalFrameStart
```

HorizontalFrameStop

Applies to: Digital, Histogram, Persist, Waveform, XY

Description: Coordinate of the right edge of the graticule containing the trace relative to the trigger time, which is at time = 0.

Example:

```
Dim XFrameStop As Double
XFrameStop = app.Acquisition.C1.Out.Result.HorizontalFrameStop
```

HorizontalOffset

Applies to: Digital, Histogram, Persist, Waveform, XY

Description: Time of the first sample. Note that the first sample is typically just off screen, to the left of the grid. Use this value when calculating the time for each sample.

Example:

```
'Waveform Interface example:
Dim HorizontalOffset As Double
HorizontalOffset = app.Acquisition.C1.Out.Result.HorizontalOffset

Dim HorizontalPerStep As Double
HorizontalPerStep = app.Acquisition.C1.Out.Result.HorizontalPerStep

'Example of calculating sample times:
Dim SampleTime as Double
Dim i as Long
For i = 0 to numPoints-1
      SampleTime = HorizontalPerStep * i + HorizontalOffset
Next
```

For a more complete example, see the example under Waveform in **DataArray** as well as the function **GetDataArray_Click**

HorizontalPerColumn

Applies to: Persist

Description: Difference in the horizontal coordinate of adjacent columns (typically in seconds). Use **HorizontalPerColumn** to determine the timing information for each column of the DataArray.

Example:

```
Dim HorizontalPerColumn As Double
HorizontalPerColumn = app.SDA.Eye.Out.Result.HorizontalPerColumn
```

HorizontalPerStep

Applies to: Digital, Histogram, Persist, Waveform, XY

Description: Time between samples, otherwise known as the sample interval. This is the inverse of the SampleRate.

Example:

```
Dim HorizontalPerStep As Double
HorizontalPerStep = app.Acquisition.C1.Out.Result.HorizontalPerStep
```

HorizontalResolution

Applies to: Digital, Histogram, Persist, Waveform, XY

Description: Resolution of the readout of horizontal values. Not directly related to the sample period.

Example:

```
Dim HorizontalResolution As Double
HorizontalResolution = app.Acquisition.C1.Out.Result.HorizontalResolution
```

HorizontalUnits

Applies to: Digital, Histogram, Persist, Waveform, XY

Description: Resolution of the readout of horizontal values. Not directly related to the sample period.

Example:

```
Dim HorizontalUnits As String
HorizontalUnits = app.Acquisition.C1.Out.Result.HorizontalUnits
```

HorizontalVarianceArray

Applies to: Waveform, XY

Description: Variant array with the variances from the nominal sample times. Used on WaveExpert oscilloscopes

Example:

```
Dim HorizontalUnits As Double
HorizontalUnits = app.Acquisition.C1.Out.Result.HorizontalUnits
```

HorizontalVariances

Applies to: Waveform, XY

Description: Number of elements in **HorizontalVarianceArray**

Example:

```
Dim HorizontalVarianceArray
HorizontalVarianceArray =
app.Acquisition.C1.Out.Result.HorizontalVarianceArray
```

IndexOfFirstSampleInFrame

Applies to: Waveform

Description: Index of the first sample that appears in the graticule. This is typically index 1, since index 0 is off-grid to the left. When traces are zoomed

Example:

```
Dim IndexOfFirstSampleInFrame as Long
IndexOfFirstSampleInFrame =
app.Acquisition.C1.Out.Result.IndexOfFirstSampleInFrame
```

LastEventTime

Applies to: All Result interfaces.

Description:

Time of the last contributing event in a set. Useful only when the result includes data produced by a sequence acquisition, or a cumulative operation such as averaging. Times are returned encoded as a currency value (VT_CY) within a variant; see description for **FirstEventTime** for decoding details.

Example:

```
Dim LastEventTime
Dim TimeString as String
LastEventTime = app.Acquisition.C1.Out.Result.FirstEventTime
TimeString = DecodeTimeProperty(LastEventTime)
```

See **FirstEventTime** for the code for the function DecodeTimeProperty, which is the example of decoding **LastEventTime.**

LastPopulatedBin

Applies to: Histogram

Description: Index of the last populated bin. (The first bin of the histogram is bin 0).

Example:

```
Dim LastPopulatedBin as Long
LastPopulatedBin = app.Math.F1.Out.Result.LastPopulatedBin
```

Levels

Applies to: Digital

Description: Returns the value 2, corresponding to the number of levels possible.

Example:

```
Dim Levels As Integer
Levels = app.LogicAnalyzer.Digital1.Out.Result.Levels
```

LineAliasName

Applies to: Digital

Description: Alias of the line name based on the input index. The alias for the line is displayed on the right side of the grid, and have values of the form "D0", "D1", etc.

Example:

```
Dim LineAliasName0 'As String
LineAliasName0 = app.LogicAnalyzer.Digital1.Out.Result.LineAliasName(0)
```

LineName

Applies to: Digital

Description: Name of the line. This can be modified via the user interface or via the property **app.LogicAnalyzer.Digital1.LineNames**

Example:

```
Dim LineName0  as String
LineName0 = app.LogicAnalyzer.Digital1.Out.Result.LineName(0)
```

Lines

Applies to: Digital

Description: Number of lines in the digital bus.

Example:

```
Dim Lines  As Integer
Lines = app.LogicAnalyzer.Digital1.Out.Result.Lines
```

Max

Applies to: Histogram

Description: Horizontal coordinate of the left edge of the last populated bin. It is equivalent to the **hmax** parameter.

Example:

```
Dim Max as double
Max = app.Math.F1.Out.Result.Max
```

MaxPopulation

Applies to: Histogram

Description: Mode of the histogram (population of the bin with the most hits).

Example:

```
Dim MaxPopulation as Long
MaxPopulation = app.Math.F1.Out.Result.MaxPopulation
```

MaxPopulationBin

Applies to: Histogram

Description: Index of the bin with the highest population. (The first bin of the histogram is bin 0).

Example:

```
Dim MaxPopulationBin as Integer
MaxPopulationBin = app.Math.F1.Out.Result.MaxPopulationBin
```

MaxPopulationInRectangle

Applies to: Persist

Description: Population of the largest element in the selected rectangle.

Arguments:

numColumns as Long

Default value: -1, use all columns

Number of columns to use in maximum population search.

numRows as Long

Default value: -1, use all rows

Number of rows to retrieve.

startColumn as Long

Default value: 0, first column

Index of the first column to use. (0-based)

startRow as Long

Default value: 0, first row

Index of the first row to use. (0-based)

Example:

```
'Example for determining the maximum population in the rectangle
Dim numColumns As Long
Dim numRows As Long
Dim startColumn As Long
Dim startRow As Long
Dim MaxPopulationInRectangle As Long
numColumns = Cells(9, 5)
numRows = Cells(10, 5)
startColumn = Cells(11, 5)
startRow = Cells(12, 5)
MaxPopulationInRectangle =
app.SDA.Eye.Out.Result.MaxPopulationInRectangle(numColumns,  _
   numRows, st artColumn, startRow)
'Use no arguments to select the full rectangle:
'MaxPopulationInRectangle = app.SDA.Eye.Out.Result.MaxPopulationInRectangle
```

Mean

Applies to: Histogram

Description: Returns the mean value of the histogam. It is equivalent to the **hmean** parameter.

Example:

```
Dim hmean as double
hmean = app.Math.F1.Out.Result.Mean
```

Min

Applies to: Histogram

Description: Horizontal coordinate of the left edge of the first populated bin. It is equivalent to the **hmin** parameter.

Example:

```
Dim hmin as double
hmin = app.Math.F1.Out.Result.Min
```

NumFrameDimensions

Applies to: All Result interfaces.

Description: Number of dimensions in the current result. For waveforms, this is typically = 2, (Voltage and time) dimensions). For X-Y mode the result is 3.

Example:

```
Dim NumFrameDimensions as Integer
NumFrameDimensions = app.Acquisition.C1.Out.Result.NumFrameDimensions
```

NumSamplesInFrame

Applies to: Waveform

Description: Number of samples within the graticule.

Example:

```
Dim NumSamplesInFrame as Long
NumSamplesInFrame = app.Acquisition.C1.Out.Result.NumSamplesInFrame
```

OffsetAtLeftEdge

Applies to: Histogram

Description: Horizontal coordinate of left edge of the first bin (either populated or unpopulated) displayed on the grid. Use **OffsetAtLeftEdge** as an offset for calculating the position of each bin.

Example:

```
'Read out BinPopulations by indexing the array using the
'FirstPopulatedBin and LastPopulatedBin properties as boundaries
'Also shows code for determining coordinate of bin centers

Const STARTROW = 5
Dim i As Integer
Dim BinPops
Dim FirstPopulatedBin, LastPopulatedBin As Integer
Dim OffsetAtLeftEdge As Double
Dim BinWidth As Double

BinPops = app.Math.F1.Out.Result.BinPopulations
FirstPopulatedBin = app.Math.F1.Out.Result.FirstPopulatedBin
LastPopulatedBin = app.Math.F1.Out.Result.LastPopulatedBin

OffsetAtLeftEdge = app.Math.F1.Out.Result.OffsetAtLeftEdge
```

```
BinWidth = app.Math.F1.Out.Result.BinWidth

For i = FirstPopulatedBin To LastPopulatedBin
    Cells(i - FirstPopulatedBin + STARTROW + 1, 7) = i
    'Calculate Bin Center
    Cells(i - FirstPopulatedBin + STARTROW + 1, 8) = OffsetAtLeftEdge +
(BinWidth / 2) + BinWidth * i
    Cells(i - FirstPopulatedBin + STARTROW + 1, 9) = BinPops(i)
Next
```

Peaks

Applies to: Histogram

Description: Number of peaks in the histogram.

Example:

```
Dim NumPeaks as Integer
NumPeaks = app.Math.F1.Out.Result.Peaks
```

PeakInfo

Applies to: Histogram

Description: VARIANT array returning information about the peak selected in the input argument. Sending 0 returns information for the entire histogram; input values from 1 to **Peaks**+1 returns information about the requested peak.

Argument (Optional):

peakIndex

Selects the peak of interest. 0 requests information about the entire histogram; >0 requests information about a single peak.

Default value: 0

Definition of output elements:

Index	Description
0	Mean
1	Sigma
2	Unused
3	Unused
4	FirstPopulatedBin
5	LastPopulatedBin
6	MaxPopulationBin
7	MaxPopulation
8	PopulationInside

Example:

```
'Reads out complete PeakInfo arrays for each peak.
Const STARTROW = 10
Const STARTCOL = 13
Dim PeakInfo
Dim i, j As Integer
Dim NumPeaks as Integer
NumPeaks = app.Math.F1.Out.Result.Peaks
For i = 0 To NumPeaks
    'Index out a peak (index 0 is the entire histogram)
    PeakInfo = app.Math.F1.Out.Result.PeakInfo(i)
    Cells(STARTROW, STARTCOL + i) = i
```

```
    'loop through each element of the selected PeakInfo array. There are 9
elements.
    For j = 0 To 8
        Cells(STARTROW + j + 1, STARTCOL + i) = PeakInfo(j)
    Next
Next
```

PopulationInside

Applies to: Histogram

Description: Population of the histogram for events contained within the number of bins configured.

Example:

```
Dim PopInside as Long
PopInside = app.Math.F1.Out.Result.PopulationInside
```

PopulationOfRectangle

Applies to: Persist

Description: Population of the selected rectangle.

Arguments:

numColumns as Long

Default value: -1, use all columns

Number of columns to use in maximum population search.

numRows as Long

Default value: -1, use all rows

Number of rows to retrieve.

startColumn as Long

Default value: 0, first column

Index of the first column to use. (0-based)

startRow as Long

Default value: 0, first row

Index of the first row to use. (0-based)

Example:

```
Dim numColumns As Long
Dim numRows As Long
Dim startColumn As Long
Dim startRow As Long
Dim PopulationOfRectangle As Long
numColumns = Cells(9, 5)
numRows = Cells(10, 5)
startColumn = Cells(11, 5)
startRow = Cells(12, 5)
numRows, startColumn, startRow)
PopulationOfRectangle =
app.SDA.Eye.Out.Result.PopulationOfRectangle(numColumns,   _     numRows,
startColumn, startRow)
PopulationOfRectangle = app.SDA.Eye.Out.Result.PopulationOfRectangle
```

PopulationOver

Applies to: Histogram

Description: Population of the events greater than the last bin.

Example:

```
Dim PopOver as Long
PopOver = app.Math.F1.Out.Result.PopulationOver
```

PopulationUnder

Applies to: Histogram

Description: Population of the events less than the first bin.

Example:

```
Dim PopUnder as Long
PopUnder = app.Math.F1.Out.Result.PopulationUnder
```

RMS

Applies to: Histogram

Description: RMS value of the histogram. Equivalent to the parameter **hrms.**

Example:

```
Dim hRMS as Double
hRMS = app.Math.F1.Out.Result.RMS
```

Rows

Applies to: Persist, Table

Description: Number of rows in the DataArray (Persist) or in the table. Typically equal to 256 for Persist waveforms. For Tables, the result can vary depending on the application.

Example:

```
'Persist example
Dim Rows As Integer
Rows = app.SDA.Eye.Out.Result.Rows
```

Samples

Applies to: Digital, Waveform, XY

Description: Number of samples in the waveform. For the Waveform and XY interface, this is typically the number of samples configured for the acquisition (e.g. 500) plus 2. The additional two points are off-grid, one on each side. For the Digital interface, this is the number of samples in each digital line + 1.

Example:

```
Dim Samples as Long
Samples = app.Acquisition.C1.Out.Result.Samples
```

Sdev

Applies to: Histogram

Description: Standard deviation of the histogram. Equivalent to the parameter **hsdev.**

Example:

```
Dim hsdev as Double
hsdev = app.Math.F1.Out.Result.Sdev
```

Status

Description: Result status. Status is returned as a decimal value, but should be treated like a register with the following bit values. See the example for decoding the Status result.

64-bit hexadecimal int value shown in XStreamBrowser	VT_DECIMAL value returned from Status property	StatusDescription	Description
0x1	0.0001	Invalid result	Indicate that this Result cannot be used for any calculation
0x2	0.0002	Data overflow	Result contains Overflow data or is calculated from a Result containing Overflows.
0x4	0.0004	Data underflow	Result contains Underflow data or is calculated from a Result containing Underflows
0x8	0.0008	Some values are undefined	Result contains some values or data that are undefined.
0x10	0.0016	Less than	Actual value is likely to be less than the calculated value. Mostly used for parameter measurements.
0x20	0.0032	Greater than	Actual value is likely to be greater than the calculated value. Mostly used for parameter measurements.
0x40	0.0064	Not a pulse	Analyzed signal is not recognized as a pulse; mostly used for pulse parameter measurements.
0x80	0.0128	Not cyclic	Contains some values measured on non-cyclic signal.
0x100	0.0256	Data averaged	Calculated value is the result of an average of multiple measurements.
0x200	0.0512	Unlocked PLL	Unlocked PLL
0x400	0.1024	Other error	Indicates some other error condition, usually accompanied by a processor-specific status description
0x800	0.2048	Other warning	Indicates some other warning condition, usually accompanied by a processor-specific status description
0x1000	0.4096	Other info	Indicates some other informative condition, usually accompanied by a processor-specific status description
0x2000	0.8192	Cumulative result	Cumulative processing somewhere in the chain

64-bit hexadecimal int value shown in XStreamBrowser	VT_DECIMAL value returned from Status property	StatusDescription	Description
0x4000 - 0x200000	1.6384 - 209.7152		reserved
0x400000	419.4304	Not an NRZ Eye	Not an NRZ Eye
0x800000	838.8608	Not an RZ Eye	Not an RZ Eye
0x1000000 - 0x80000000	1677.7216 - 214748.3648		reserved
0x100000000	429496.7296	Inputs incompatible	Some incompatibility exists between inputs (e.g. units, frame, sample rate)
0x200000000	858993.4592	Algorithm limits reached	Algorithm couldn't converge or some other algorithmic limitation reached.
0x400000000	1717986.918	Bad settings	Combination of settings isn't correct.
0x800000000	3435973.837	Too little data	Not enough data to accurately perform calculation.
0x1000000000	6871947.674	Too much data	Too muchdata to perform calculation.
0x2000000000	13743895.35	Requires uniform horz interval	Uniform horizontal interval required.
0x4000000000	27487790.69	Bad units	Incorrect or invalid units
0x8000000000	54975581.39	Data range too low	Range of data values is too close to the resolution (e.g. bad signal to noise ratio).
0x10000000000	109951162.8	Data undersampled	Sample rate too low
0x20000000000	219902325.6	Poor statistics	Too few samples, e.g. to populate a data histogram to have enough statistics
0x40000000000	439804651.1	Slow transition time	Transition time to slow, e.g. JTA time measurement accuracy for clock that requires fast edges.
0x80000000000	879609302.2	Data resampled	Data is resampled
0x100000000000	1759218604	Data interpolated	Data is interpolated; e.g. interpolating missing points in RIS
0x200000000000	3518437209	Measurement scale imprecise	If all the data falls in the same histogram bin, one line for trend, one pixel for XY, etc...
0x400000000000	7036874418	No data available	Data not available (e.g. processing inputs disconnected or clear sweeps on cumulative processing).
0x800000000000	14073748836	Some accumulated results invalid	One or more results or values within the accumulated result were invalid and were skipped.
0x1000000000000	28147497671	Insufficient Memory	Memory limitations reached, attempted allocations failed.

64-bit hexadecimal int value shown in XStreamBrowser	VT_DECIMAL value returned from Status property	StatusDescription	Description
0x2000000000000	56294995342	Channel not active	Source channel inactive.
0x4000000000000	1.1259E+11		use status description (custom string)
0x8000000000000 - 0x8000000000000000			reserved

StatusDescription

Description:

Example: Description for each status code. See the above table.

Sweeps

Applies to: Digital, Histogram, Persist, Waveform, XY

Description: Number of sweeps used for results that are based on multiple sweeps, such as averages, histograms, eye diagrams, etc.

Example:

```
Dim Sweeps as Long
Sweeps = app.Acquisition.C1.Out.Result.Sweeps
```

Top

Applies to: Histogram

Description: Horizontal coordinate of the rightmost of the two most populated histogram peaks. It is equivalent to the paramater **htop.**

Example:

```
Dim htop as Double
htop = app.Math.F1.Out.Result.Top
```

See the **HistogramProperties_Click** subroutine for a more complete example.

UniformInterval

Applies to: Digital

Description: Returns if the samples are evenly spaced in time. Currently always true.

Example:

```
Dim htop as Double
htop = app.Math.F1.Out.Result.Top
```

UpdateTime

Applies to: All Result interfaces.

Description: Indicates the time that particular result was most recently updated. Often, this property will show the same value as LastEventTime, but for traces such as memories (M1, M2, etc) this will be the time that the waveform was stored to the memory trace. Times are returned encoded as a currency value (VT_CY) within a variant; see description for **FirstEventTime** for decoding details.

Example:

```
Dim UpdateTime
Dim TimeString as String
UpdateTime = app.Memory.M1.Out.Result.UpdateTime
TimeString = DecodeTimeProperty(UpdateTime)
```

See **FirstEventTime** for the code for the function DecodeTimeProperty, which is the example of decoding **UpdateTime.**

Value

Applies to: Param

Description: Value of a parameter or of a statistic in the measurement system. When the scope has no result, it will show "---" in the measure table. When this is displayed, the Value property is not accessible. Error handling should be employed to catch errors that may occur. See the example below for more information.

Example:

```
' Readout all elements of the measurement parameter for P1.
' Use the On Error Resume Next statement to handle situations
' where a measurement is not available, such as attempting to
' readback the frequency of a DC level, or trying to read the Sdev
' statistic when there is only 1 measurement.  The scope will show
'  "---" when there is no result to readback.

On Error Resume Next
Dim Value, Mean, Min, Max, Sdev, Num As Double
Value = app.measure.p1.out.Result.Value
Mean = app.measure.p1.Mean.Result.Value
Min = app.measure.p1.Min.Result.Value
Max = app.measure.p1.Max.Result.Value
Sdev = app.measure.p1.Sdev.Result.Value
Num = app.measure.p1.Num.Result.Value
```

ValueArray

Applies to: Param

Description: Retrieves a 1D array of values, typically used to return measurement results for mult-values parameters such as Period, Frequency, TIE@level, etc. ValueArray can also return the start time, stop time and status of each measurement taken in the sweep.

Arguments:

numSamples as Long

Default value: -1, retrieve all data

Number of samples to retreived

startIndex as Long

Default value: 0, first sample

Index of the first sample to be retreived.

valueType as integer (enumerated)

Default value: 1

Accepted values:

1	Readback parameter values
2	Readback start time of measurement
4	Readback stop time of measurement
8	Readback status value. Use this to determine if edge measurements are of positive or negative polarity

Example:

```
' Readout all elements of the measurement parameter for P1.
' Use the On Error Resume Next statement to handle situations
' where no measurements were not taken. On Error Resume Next
' The scope will show'  "---" when there is no result to readback

Const ROWOFFSET = 16
Const COLOFFSET = 2
Dim numValues As Long
Dim startIndex As Long
Dim valueType As Integer
Dim valArray


numValues = Cells(9, 5)
startIndex = Cells(10, 5)

'Get measurement values
valArray = app.measure.p1.out.Result.ValueArray(numValues, startIndex, 1)
'Get measurement start times
startTimes = app.measure.p1.out.Result.ValueArray(numValues, startIndex, 2)
'Get measurement stop times
stopTimes = app.measure.p1.out.Result.ValueArray(numValues, startIndex, 4)
'Get measurement status values (typically edge polarity)
statusVals = app.measure.p1.out.Result.ValueArray(numValues, startIndex, 8)

Dim Value, Mean, Min, Max, Sdev, Num As Double
For i = 0 To UBound(valArray)
    Cells(i + ROWOFFSET, COLOFFSET) = i
    Cells(i + ROWOFFSET, COLOFFSET + 1) = valArray(i)
    Cells(i + ROWOFFSET, COLOFFSET + 2) = startTimes(i)
    Cells(i + ROWOFFSET, COLOFFSET + 3) = stopTimes(i)
    Cells(i + ROWOFFSET, COLOFFSET + 4) = statusVals(i)
Next
```

VerticalFrameStart

Applies to: Digital, Histogram, Persist, Waveform, XY

Description: Vertical coordinate of the bottom edge of the graticule containing the trace.

Example:

```
Dim YFrameStart As Double
YFrameStart = app.Acquisition.C1.Out.Result.VerticalFrameStart
```

VerticalFrameStop

Applies to: Digital, Histogram, Persist, Waveform, XY

Description: Vertical coordinate of the top edge of the graticule containing the trace.

Example:

```
Dim YFrameStop As Double
YFrameStop = app.Acquisition.C1.Out.Result.VerticalFrameStop
```

VerticalMaxPossible

Applies to: Waveform

Description: Returns the maximum voltage that can be returned.

Example:

```
Dim VerticalMaxPossible As Double
VerticalMaxPossible = app.Acquisition.C1.Out.Result.VerticalMaxPossible
```

VerticalMinPossible

Applies to: Waveform

Description: Returns the minimum voltage that can be returned.

Example:

```
Dim VerticalMinPossible As Double
VerticalMinPossible = app.Acquisition.C1.Out.Result.VerticalMinPossible
```

VerticalOffset

Applies to: Persist, Waveform

Description:

Potential difference between ground and the center of the screen. With **VerticalOffset** = +50 mV, the center of the screen represents -50 mV: with an offset of -21mV, the center represents +21 mV. (I.e. centerline voltage + **VerticalOffset** = 0)

Example:

```
Dim VerticalOffset As Double
VerticalOffset = app.Acquisition.C1.Out.Result.VerticalOffset
```

VerticalPerRow

Applies to: Persist

Description: Returns the increment of the vertical coordinate between rows (typically in volts). Use **VerticalPerRow** to determine the voltage information for each row of the **DataArray**.

Example:

```
Dim VerticalPerRow As Double
VerticalPerRow = app.SDA.Eye.Out.Result.VerticalPerRow
```

VerticalPerStep

Applies to: Waveform

Description: Smallest step size in the numerical values that can be read out from the **DataArray**, which utilzed 16 bit signed integer values. Note that the vertical range is almost exactly 65536 * **VerticalPerStep.** (See **VerticalMaxPossible** and **VerticalMinPossible.**)

Example:

```
Dim VerticalPerStep as Double
VerticalPerStep = app.Acquisition.C1.Out.Result.VerticalPerStep
```

VerticalResolution

Applies to: Param, Waveform

Description: Vertical resolution of the Y coordinate, which is the actual smallest difference that can be practically resolved. Using averaging can improve the resolution by the square root of the number of sweeps in the average. For example, if 16 averages are set via pre-processor averaging or by using the **Average** math function, the resolution is improved by a factor of 4. For 100 sweeps it improves by a factor of 10.

Example:

```
Dim VerticalResolution as Double
VerticalResolution = app.Acquisition.C1.Out.Result.VerticalResolution
```

VerticalUnits

Applies to: Histogram, Param, Persist, Waveform

Description: Units used for the vertical axis. Typically V for volts, but other units are possible, such as "A" for amps when using a current probe. When selecting to change units when using the Rescale math function, a variety of other chooses for units are available.

Example:

```
Dim VerticalUnits as String
VerticalUnits = app.Acquisition.C1.Out.Result.VerticalUnits
```

XFrameStart

Applies to: XY

Description: Coordinate of the left edge of the XY graticule.

Example:

```
Dim XFrameStart As Double
XFrameStart = app.Math.XY.Out.Result.XFrameStart
```

XFrameStop

Applies to: XY

Description: Coordinate of the right edge of the XY graticule.

Example:

```
Dim XFrameStop As Double
XFrameStop = app.Math.XY.Out.Result.XFrameStop
```

XMaxPossible

Applies to: XY

Description: Maximum possible value of the X coordinate

Example:

```
Dim XMaxPossible As Double
XMaxPossible = app.Math.XY.Out.Result.XMaxPossible
```

XMinPossible

Applies to: XY

Description: Minimum possible value of the X coordinate

Example:

```
Dim XMinPossible As Double
XMinPossible = app.Math.XY.Out.Result.XMinPossible
```

XOffset

Applies to: XY

Description: Potential difference between ground and the horizontal center of the XY grid. For example, with **VerticalOffset** of the X coordinate's source trace = +50 mV, the center of the screen represents -50 mV: with an offset of -21mV, the center represents +21 mV. (I.e. centerline voltage + **VerticalOffset** = 0)

Example:

```
Dim XOffset As Double
XOffset = app.Math.XY.Out.Result.XOffset
```

XPerStep

Applies to: XY

Description: Smallest step size in the numerical values that can be read out from the **DataArray** utilzing 16 bit signed integer values. Note that the X range is just about 65536 * **XPerStep.** (See **XMaxPossible** and **XMinPossible.**)

Example:

```
Dim XPerStep As Double
XPerStep = app.Math.XY.Out.Result.XPerStep
```

XResolution

Applies to: XY

Description: Vertical resolution of the X coordinate, which is the actual smallest difference that can be practically resolved. Using averaging can improve the resolution by the square root of the number of sweeps in the average. For example, if 16 averages are set via pre-processor averaging or by using the **Average** math function, the resolution is improved by a factor of 4. For 100 sweeps it improves by a factor of 10.

Example:

```
Dim XResolution As Double
XResolution = app.Math.XY.Out.Result.XResolution
```

XUnits

Applies to: XY

Description: Units of the X coordinate. Typically V for Volts, unless a current probe is in use or if the Rescale function is being used to change the trace's units.

Example:

```
Dim XUnits As String
XUnits = app.Math.XY.Out.Result.XUnits
```

YFrameStart

Applies to: XY

Description: Coordinate of the bottom edge of the XY graticule.

Example:

```
Dim YFrameStart As Double
YFrameStart = app.Math.XY.Out.Result.YFrameStart
```

YFrameStop

Applies to: XY

Description: Coordinate of the top edge of the XY graticule.

Example:

```
Dim YFrameStart As Double
YFrameStart = app.Math.XY.Out.Result.YFrameStart
```

YMaxPossible

Applies to: XY

Description: Minimum possible value of the Y coordinate.

Example:

```
Dim YMaxPossible As Double
YMaxPossible = app.Math.XY.Out.Result.YMaxPossible
```

YMinPossible

Applies to: XY

Description: Minimum possible value of the Y coordinate.

Example:

```
Dim YMinPossible As Double
YMinPossible = app.Math.XY.Out.Result.YMinPossible
```

YOffset

Applies to: XY

Description: Potential difference between ground and the vertical center of the XY grid. For example, with **VerticalOffset** of the Y coordinate's source trace = +50 mV, the center of the screen represents -50 mV: with an offset of -21mV, the center represents +21 mV. (I.e. centerline voltage + **VerticalOffset** = 0)

Example:

```
Dim YOffset As Double
YOffset = app.Math.XY.Out.Result.YOffset
```

YPerStep

Applies to: XY

Description: Smallest step size in the numerical values that can be read out from the **DataArray** utilzing 16 bit signed integer values. Note that the Y range is just about 65536 * **YPerStep.** (See **YMaxPossible** and **YMinPossible.**)

Example:

```
Dim YPerStep As Double
YPerStep = app.Math.XY.Out.Result.YPerStep
```

YResolution

Applies to: XY

Description: Vertical resolution of the Y coordinate, which is the actual smallest difference that can be practically resolved. Using averaging can improve the resolution by the square root of the number of sweeps in the average. For example, if 16 averages are set via pre-processor averaging or by using the **Average** math function, the resolution is improved by a factor of 4. For 100 sweeps it improves by a factor of 10.

Example:

```
Dim YResolution As Double
YResolution = app.Math.XY.Out.Result.YResolution
```

YUnits

Applies to: XY

Description: Units of the Y coordinate. Typically V for Volts, unless a current probe is in use or if the Rescale function is being used to change the trace's units.

Example:

```
Dim YUnits As String
YUnits = app.Math.XY.Out.Result.YUnits
```

LECROY.XSTREAMDSO.1

app

This is the root of the automation hierarchy, all other nodes are accessed from this point.

AutoSetup	Action
ClearSweeps	Action
Exit	Action
FirmwareVersion	String
Height	Property
HideClock	Bool
InstrumentID	String
InstrumentModel	String
Left	Property
Maximize	Action
Minimize	Action
OperationMode	Enum
Quit()	Method
ResetPreferences	Action
Restore	Action
SetToDefaultSetup	Action
Shutdown	Action
Sleep([in] double timeoutMilliseconds)	Method
Top	Property
TouchScreenEnable	Bool
WaitUntilIdle([in] double timeoutSeconds)	Method
Width	Property
Windowed	Action
WindowState	Property

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Popup a dialog containing the instrument model
MsgBox "Model is: " & app.InstrumentModel
```

InstrumentModel

String

Range Any number of characters

Description

Queries the model number of the instrument.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Present the model number of the instrument.
MsgBox app.InstrumentModel
```

InstrumentID *String*

Range Any number of characters

Description

Reads the complete ID of the instrument in the format: "LECROY,WM8500,WM000001,0.0.0", which includes the maker, the instrument model number, the serial number, and the version number.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Present the ID of the instrument.
MsgBox app.InstrumentID
```

FirmwareVersion *String*

Range Any number of characters

Description

Queries the firmware version of the instrument in the form - "1.0.0 (build 12345)"

Example

```
' Microsoft Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Query the firmware version number of the instrument.
MsgBox "Firmware Version is: " + app.FirmwareVersion
```

Shutdown *Action*

Description

Shuts down the instrument. It will prompt the user with an 'Are you sure?' dialog before shutting down. Note that until the user responds to the dialog, control via Automation will be blocked.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Shut down the instrument with a confirmation prompt.
app.Shutdown
```

Exit *Action*

Description

Equivalent to app.Quit() method.

Maximize *Action*

Description

Maximize the instrument window to fill the underlying desktop.
Equivalent to app.WindowState = 1

Minimize *Action*

Description

Minimizes the instrument window to reveal the underlying desktop. It will display a small window in the bottom right corner of the display, which when clicked will restore the window to full-screen mode. To programmatically restore the window refer to the app.WindowState control.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Minimize the instrument display.
app.Minimize
```

Windowed *Action*

Description

Places the instrument application in windowed mode (as opposed to full-screen mode). Places the application in the upper-part of the display screen with a sizable border.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the instrument display into the windowed mode.
app.Windowed
```

Restore *Action*

Description

Restore the instrument display to its position and size before the last minimize request.

SetToDefaultSetup *Action*

Description

Restores the instrument setup to its default state. Note that certain settings will not be restored to the default state. These are the user preferences, such as the current remote communications port, and the color settings, which may be reset, if required, using the ResetPreferences action.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Restore the instrument to its default state.
app.SetToDefaultSetup
```

AutoSetup *Action*

Description

Starts an AutoSetup operation. When input channels are visible, AutoSetup operates only on those visible channels. If no channels are visible, all channels are affected by AutoSetup. With more than one channel visible, the first visible channel in numerical order, that has a detectable signal applied to it, is automatically set up for edge triggering.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Start an Auto-Setup process.
app.AutoSetup
```

ClearSweeps *Action*

Description

Clears all accumulated sweeps for all subsystems. These include Channel Pre-Processing, Math, Measure, and Display Persistence. Note that subsystem-specific clear sweeps controls are also available. For the details please refer to the ClearSweeps control for each subsystem.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Clear all accumulated sweeps for all subsystems.
app.ClearSweeps
```

HideClock *Bool*

Description

Hides/Shows the clock that resides in the lower-right corner of the display of the instrument.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Hide the clock for 3 seconds.
app.HideClock = True
app.Sleep(3000)
app.HideClock = False
```

TouchScreenEnable *Bool*

Description

Sets/Queries the state of the touch-screen enable control. This is equivalent to the front-panel Touch Screen button.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Disable touch-screen if it is enabled.
if app.TouchScreenEnable = True then
   app.TouchScreenEnable = False
End if
```

ResetPreferences *Action*

Description

Resets all scope preferences to their default states. The set includes the current remote communications port, the color palette settings, etc. but does not include the main DSO controls such as V/Div, T/Div, etc. These main instrument controls can be reset using the SetToDefaultSetup control.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Reset all instrument preferences.
app.ResetPreferences
```

OperationMode *Enum*

Description

Sets the operating mode of the WaveExpert. Each mode affects the horizontal and vertical menus, as well as the display mode, as appropriate for the selected mode of operation.
Not present on other scope families.

Values

Eye	Eye Diagram mode. Sets sequential timebase, and infinite persistence.
Scope	Traditional 'Scope' mode.
TDR	Time Domain Reflectometry (TDR) mode.

Quit() *Method*

Description

Closes the instrument application. The instrument will prompt the user with an 'Are you sure?' dialog before closing down. Note that until the user responds to the dialog, control via Automation will be blocked.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Quit the instrument application with a confirmation prompt.
app.Quit
```

WindowState *Property*

Description

Sets/Queries the state of the PC window used by the instrument display.
0 windowed
1 full screen
2 minimized
Trying to set values greater than 2 or less than 0 will result in the value 0 (windowed) being set.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the instrument window state to windowed.
app.WindowState = 0
```

Width *Property*

Description

Sets/Queries the width in pixels of the instrument display on the PC screen.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the width of the instrument window to 800 pixels.
app.Width = 800
```

Height *Property*

Description

Sets/Queries the height in pixels of the instrument display on the PC screen.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the height of the instrument window to 400 pixels.
app.Height = 400
```

Top *Property*

Description

Sets/Queries the position in pixels of the top edge of the instrument display on the PC screen. The position is measured downwards from the top of the screen to the top of the instrument window.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the position of the top edge of the instrument window to 100 pixels.
app.Top = 100
```

Left *Property*

Description

Sets/Queries the position in pixels of the left edge of the instrument display on the PC screen. The position is measured from the left edge of the screen to the left edge of the instrument window.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")
' Set the position of the left edge of the instrument window to 100 pixels.
app.Left = 100
```

WaitUntilIdle([in] double timeoutSeconds) *Method*

Description

Waits until either the application is idle or the specified timeout expires, specified in seconds. This evaluates to True if the application completes before the timeout expires, and to False if a timeout occurs.
When Trigger mode is Auto or Run, the application is never Idle. In this case the call to WaitUntilIdle returns after the next acquisition and any configured processing.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Wait with a timeout of five seconds.
app.WaitUntilIdle(5)
```

Sleep([in] double timeoutMilliseconds) *Method*

Description

Causes the main execution thread of the instrument application to sleep for the specified time period, defined in milliseconds.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

MsgBox "Sleeping for 10 seconds..."
app.Sleep(10000)
MsgBox "Sleep finished"
```

ACQUISITION *app.Acquisition*

This group of variables controls the input channels C1, C2, C3 and C4, the timebase, the trigger, and the Aux Output.

Names of the form app.Acquisition.Channels.xxxx are aliases of simpler names which are described in this section of the manual. Examples of alias pairs are as follows -
app.Acquisition.Channels("Cx") is equivalent to app.Acquisition.Cx
app.Acquisition.Channels(1) is equivalent to app.Acquisition.C1
app.Acquisition.Channels("Cx").Out.Result is equivalent to app.Acquisition.Cx.Out.Result

Acquire([in] double timeoutSeconds, [in] long bForceTriggerOnTimeout)	Method
Calibrate	Action
CalNeeded	Integer
ClearSweeps	Action
HorOffset	Double
TriggerMode	Enum

TriggerMode *Enum*

Description

Sets/Queries the trigger mode, using values from the following list -
Auto, Norm, Normal, Single, Stopped.

Auto: After a timeout, if a real hardware trigger is not received, then force a trigger so there are automatically lots of updates.

Normal: Accepts triggers as rapidly as the system permits, but likewise will wait "forever" for a trigger, without updating anything.

Single: Arm the acquisition system to acquire once, and do not rearm automatically after. Once a trigger is received and the data processed, the instrument finishes in the "Stopped" state.

Stop: Finishes the current acquisition and does not re-arm.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Place the instrument in stopped mode and take one acquisition.
app.Acquisition.TriggerMode = "Stopped"
app.Acquisition.Acquire(5)
```

Values

Auto	Auto-trigger
Normal	Normal Trigger
Single	Single Trigger
Stopped	No trigger possible, Stopped

HorOffset *Double*

Range From -2.72e-008 to -1.72e-008 step 2e-011

Description

same as "app.Acquisition.Horizontal.HorOffset.cvar"

Calibrate *Action*

Description

Initiates a full calibration of the acquisition system of the instrument.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Start a calibration.
app.Acquisition.Calibrate
```

CalNeeded *Integer*

Range From -2147483648 to 2147483647 step 1

Description

Query: Indicates whether calibration is required or not.

Based on hexadecimal bit value, it provides following information:
0x00000001: Front end calibration is required
0x00000002: Digitizers delay matching is required
0x00000004: Digitizers gain matching is required
0x00000008: Trigger level calibration is required
0xFFFFFFFF(-1): All of above calibrations are required

ClearSweeps *Action*

Description

Resets any accumulated average data or persistence data for channel waveforms (C1..C4). Valid only when one or more channels have waveform averaging or persistence enabled in their pre-processing settings. Note that an average may be reset on an individual basis using app.Acquisition.Cx.ClearSweeps control.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Clear accumulated sweeps for channels C1...C4
app.Acquisition.ClearSweeps

' Clear accumulated sweeps for only C1
app.Acquisition.C1.ClearSweeps
```

Acquire([in] double timeoutSeconds, [in] long bForceTriggerOnTimeout) *Method*

Description

Action/Query. Takes a single acquisition. The first of the two arguments specifies a timeout; the second, which is optional, specifies whether or not to force a trigger when the timeout occurs. Evaluates to True if a trigger occurred, or False if a timeout occurred.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")
' Start an acquisition, wait for up to 5 seconds for a trigger
' event, force a software trigger if a hardware trigger is not
' detected before the 5 second timeout expires.
triggerDetected = app.Acquisition.Acquire(5, true)
```

CHANNELS *app.Acquisition.Channels*

This group of variables controls the acquisition channels C1, C2, C3 and C4.

Names of the form app.Acquisition.Channels.xxxx are aliases of simpler names which are described in the section of the

manual which is devoted to app.Acquisition. Examples of alias pairs are as follows -
app.Acquisition.Channels("Cx") is equivalent to app.Acquisition.Cx
app.Acquisition.Channels(1) is equivalent to app.Acquisition.C1
app.Acquisition.Channels("Cx").Out.Result is equivalent to app.Acquisition.Cx.Out.Result

Example

```
Set app = CreateObject("LeCroy.XStreamDSO")

For X = 1 To 4
    app.Acquisition.Channels(X).VerScale = 0.2
Next
```

CX

app.Acquisition.Cx

This group of variables controls the input channels C1, C2, C3 and C4.

Names of the form app.Acquisition.Channels.xxxx are aliases of simpler names which are described in the section of the manual which is devoted to app.Acquisition. Examples of alias pairs are as follows -
app.Acquisition.Channels("Cx") is equivalent to app.Acquisition.Cx
app.Acquisition.Channels("Cx").Out.Result is equivalent to app.Acquisition.Cx.Out.Result

AverageSweeps	Integer
CalibrateScope	Action
ClearSweeps	Action
DarkCalLevel	Double
Deskew	Double
FindScale	Action
InterpolateType	Enum
Invert	Bool
LabelsPosition	String
LabelsText	String
MaskOn	Bool
Persisted	Bool
PersistenceSaturation	Integer
PersistenceTime	Enum
PodModel	String
PodType	Enum
ProbeAttenuation	Double
ProbeName	String
ShowLastTrace	Bool
TDREdgeTime	Double
TDRMeasurementType	Enum
TDROn	Bool
TDRPolarity	Enum
TDRZ0	Double
UseGrid	String
VerOffset	Double
VerScale	DoubleLockstep
VerScaleVariable	Bool

View	Bool
ViewLabels	Bool
XTolerance	Integer
YTolerance	Integer

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Setup Channel C1
app.Acquisition.C1.VerScale = 0.5
app.Acquisition.C1.VerOffset = 0.0
app.Acquisition.C1.Coupling = "DC50"

' Setup Channel C2
app.Acquisition.C2.VerScale = 0.1
```

View *Bool*

Description

Sets/Queries the channel's 'Viewed' state. When True the channel waveform is displayed on one of the display graticules. Note that even when a channel is not visible it may be used as a source for Math, Measure, etc.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Make channel C3 visible.
app.Acquisition.C3.View = True
```

ClearSweeps *Action*

Description

Clears all accumulated average data and persistence data for this channel. See app.Acquisition.ClearSweeps for a control that clears accumulated data for channels 1..4, or app.ClearSweeps for a control that clears accumulated data for all subsystems (including Math/Measure/Display, etc.)

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Reset channel C1
app.Acquisition.C1.ClearSweeps

' Reset channels C1..C4
app.Acquisition.ClearSweeps
```

UseGrid *String*

Range Any number of characters

Description

Sets/Queries the graticule on which the trace is displayed. Typical values include:
YT1..YT8: one of the YT graticules used in Single, Dual, Quad, and Octal display modes.
NotOnGrid: not displayed on any graticule.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Switch to dual grid mode, place C1 on the lower graticule
' and C2 on the upper graticule.
app.Display.GridMode = "Dual"
app.Acquisition.C1.UseGrid = "YT2"
app.Acquisition.C2.UseGrid = "YT1"
```

Persisted *Bool*

Description

Sets/Queries the persisted state of the waveform. If the Display.LockPersistence control is set to 'AllLocked' then the persisted state of all displayed waveforms will be the same. If the Display.LockPersistence control is set to 'PerTrace' then the persisted state of each waveform may be independently controlled.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set persistence on for trace C1
app.Display.LockPersistence = "PerTrace"
app.Acquisition.C1.Persisted = True
```

PersistenceSaturation *Integer*

Range From 0 to 100 step 1

Description

Sets/Queries the saturation threshold for persisted waveforms.
All information at this level or above will be recorded with the same color or intensity.
See the general description above for a discussion of the locked and unlocked persistence modes.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the persistence saturation level for trace C1.
app.Acquisition.C1.PersistenceSaturation = 60
```

ShowLastTrace *Bool*

Description

Sets/Queries the state of the Show Last Trace control. If True then when this trace is displayed in persistence mode the last acquired waveform will be superimposed on the accumulating persistence map.
See the general description above for a discussion of the locked and unlocked persistence modes.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Makes the last acquired trace invisible for the
' persistence trace of channel C1.
app.Acquisition.C1.ShowLastTrace = False
```

PersistenceTime *Enum*

Description

Sets/Queries the state of the Persistence Time control. Controls the persistence decay time for this trace. See the general description above for a discussion of the locked and unlocked persistence modes.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the persistence time for the persistence trace of channel C1 to 10
seconds.
app.Acquisition.C1.PersistenceTime = "10s"
```

Values

0.5s	
10s	
1s	
20s	
2s	
5s	
Infinite	

LabelsText *String*

Range Any number of characters

LabelsPosition *String*

Range Any number of characters

Description

Sets / Queries the horizontal position of the label attached to the acquisition trace Cx. The unit of measurement is the unit of the horizontal scale. The measurement is made from the trigger point. Note that this control is a string, not a numeric value. This allows multiple labels to be positioned, as shown in the example below.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Add a couple of labels to trace C1, one at 0ns, and one at 55ns
app.SetToDefaultSetup
app.Acquisition.C1.ViewLabels = True
app.Acquisition.C1.LabelsPosition = "0.0,55e-9"
app.Acquisition.C1.LabelsText = "Hello,World"
```

ViewLabels *Bool*

Description

Sets/Queries whether the user-defined labels for the trace are visible.
See Also: LabelsPosition and LabelsText controls.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Show the user-defined label for trace C2.
app.Acquisition.C2.ViewLabels = True
```

FindScale *Action*

Description

Starts FindScale operation for this chanel. This operation will adjust channel's v/div and offset control such that the signal is visible on the screen with in +/- 3 div.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

'Find vertical scale of channel 1
app.Acquisition.C1.FindScale
```

ProbeAttenuation ***Double***

Range From 1e-006 to 10000 step 1e-006

Description

Sets/Queries the probe attenuation. The probe attenuation is the factor by which the signal is made smaller, for example, 10 means that the probe divides by 10, and is referred to as a ÷10 probe. Note that certain passive probes may be marked as 'x10', even though they actually divide the input signal by a factor of 10.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

'  Set the probe attenuation for channel C1 to 100
app.Acquisition.C1.ProbeAttenuation = 100
```

ProbeName ***String***

Range Any number of characters

Description

Queries the name of connected probe.

VerScale ***DoubleLockstep***

Range From 0.001 to 1 step 0.0005, locked to 1 2 5, fine grain allowed=true, on=false

Description

Sets/Queries the vertical scale (in Volts/Division) of an input channel. When variable gain (VerScaleVariable control) is disabled, the control will clip values to a 1..2..5 sequence. When variable gain is enabled, the setting resolution lies in the range 1% to 2%, depending upon the numerical value.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set C1 to a scale of 250mV/Div in Variable Scale mode
app.Acquisition.C1.VerScaleVariable = True
app.Acquisition.C1.VerScale = 0.25
```

VerScaleVariable *Bool*

Description

Sets/Queries the state of the variable vertical scale control for channel Cx. When the variable scale is enabled, the setting resolution lies in the range 1% to 2%, depending on the numerical value. If a knowledge of the exact value is important, the value should be read back after a setting has been made.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the variable vertical scale for C1 to On.
app.Acquisition.C1.VerScaleVariable = True
```

VerOffset *Double*

Range From -1 to 1 step 0.001

Description

Sets/Queries the vertical offset of input channel Cx. The setting resolution in volts lies in the range 0.25% to 0.5%, depending on the numerical value.
Note that the available offset range is dependent upon the current V/Div setting, and also the instrument model.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the vertical offset for C1 to 10 mV.
app.Acquisition.C1.VerOffset = 0.01
```

AverageSweeps *Integer*

Range From 1 to 1000000 step 1

Description

Sets/Queries the number of averaging sweeps for input channel Cx. This is distinct from the math function app.Math.Fx. If the number of sweeps is 1 (the default value), the data will not be averaged.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the number of sweeps for channel C1 to 25.
app.Acquisition.C1.AverageSweeps = 25
```

InterpolateType *Enum*

Description

Sets/Queries the type of interpolation used for input channel Cx. Note that Sinx/x interpolation increases the size of the trace by a factor of 10, beware when using this option with long records.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

'  Set the interpolation for channel C3 to (sin x)/x
app.Acquisition.C3.InterpolateType = "Sinxx"
```

Values

Linear	Linear interpolation
Sinxx	Sinx/x interpolation

Invert *Bool*

Description

Sets/Queries whether input channel Cx is inverted.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set channel C2 to be inverted.
app.Acquisition.C2.Invert = True
```

Deskew *Double*

Range From -9e-009 to 9e-009 step 1e-012

Description

Sets/Queries the deskew of input channel Cx to produce a required alignment with another trace.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")
```

DarkCalLevel *Double*

Range From -1 to 1 step 1e-007

Description

Set/Query the Dark Cal Level, the residual power measured by the optical head with no input applied (dark input).
Used only by the extinction ratio measurement.
Units are Micro-Watts (uW)

CalibrateScope *Action*

Description

This control will initiate DC Calibration of modules. Disconnect the inputs from all plugged in Modules before starting calibration process.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Start calibration of modules
app.Acquisition.C1.CalibrateScope
```

TDROn *Bool*

Description

This control is applicable in TDR operating mode only. This control turns ON or OFF TDR pulse for given channel.

TDRPolarity *Enum*

Description

This control is applicable in TDR operating mode only. This variable controls polarity of TDR pulse.

Values

Neg	Negative Pulse
Pos	Positive Pulse

TDREdgeTime *Double*

Range From 2e-011 to 2e-010 step 1e-012

Description

This control is used for reducing edge time of TDR pulse. The reduced edge time is achieved by using software low pass Filter. Note: The default TDR edge time is 20 pS.

TDRMeasurementType *Enum*

Description

This control is applicable in TDR operating mode only. It defines unit of TDR signal.

Values

Ohm	
Percent	
Reactance	
Volt	

TDRZ0 *Double*

Range From 0 to 1000 step 0.1

Description

Characteristic Impedance for TDR medium

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the characteristic impedance C1 to 50 ohms.
app.Acquisition.C1.TDRZ0 = "50"
```

MaskOn *Bool*

Description

In Eye mode operation, this control turns on and off mask for given channel.

XTolerance *Integer*

Range From 0 to 100 step 1

Description

In Eye mode operation, this variable controls X-axis(horizontal) tolerance during Mask testing

YTolerance *Integer*

Range From 0 to 100 step 1

Description

In Eye mode operation, this variable controls Y-axis(vertical) tolerance during Mask testing

PodModel *String*

Range Any number of characters

Description

This control reflects the model number of the Module.

PodType *Enum*

Description

This control reflects the type of the Module.

Values

Electrical	The module has an electrical signal input
ElectricalTDR	The module has TDR capabilities
None	
Optical	The module has an optical signal input

RESULT

app.Acquisition.Cx.Out.Result

Properties of the type xxxx.Out.Result.yyyy are those of the last completed acquisition. They are not affected if other controls are changed after that acquisition was completed. This distinction between "Out.Result" properties and other controls is most important when the trigger mode is Single or Stopped. You should treat "Out.Result" properties as read-only.

Several of these properties mention the 'frame', this is the term used to describe the visible portion of the trace, which is generally smaller than the acquired waveform. The frame could be used for example to display a 500pt. window onto a 1Mpt. Trace, or vertically it could be used to show the 'center' 10mV of a 100mV pk trace.

For a full overview of the properties of waveform (or other) results, please see Chapter 1.

HORIZONTAL

app.Acquisition.Horizontal

This group of variables controls the timebase, the sampling, and the trigger delay.

AcquisitionDuration	Double
CISStrobeFreq	Double
FindBitRate	Action
FPGAStatus	Register
HorOffset	Double
HorScale	DoubleLockstep
HorScaleVariable	Bool
HorUnits	String
MaxSamples	DoubleLockstep
NumPoints	Integer

OperationMode	Enum
SampleMode	Enum
SamplingRate	Double
SignalBitRate	Double
SignalStandardUI	Enum
TimePerPoint	Double
Units	Enum

HorScale *DoubleLockstep*

Range From 1e-012 to 0.001 step 1e-011, locked to 1 2 5, fine grain allowed=true, on=false

Description

Sets/Queries the horizontal scale in time per division.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the horizontal acquisition scale to 200 ns/div.
app.Acquisition.Horizontal.HorScale = 2.0e-7
```

HorOffset *Double*

Range From -1.00122e-005 to -1.22e-008 step 2e-011

Description

Sets/Queries the horizontal position of the trigger time, relative to the origin set by HorOffsetOrigin, in seconds. Positive to the right, negative to the left. The setting resolution is about 1% to 2&.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the horizontal trigger offset to 200 ns.
app.Acquisition.Horizontal.HorOffset = 2.0e-7
```

HorUnits *String*

Range Any number of characters

Description

Queries the units in which the horizontal scale is measured.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Obtain the units of the horizontal scale.
HorizUnit = app.Acquisition.Horizontal.HorUnits
```

NumPoints *Integer*

Range From 2 to 512000000 step 1

Description

Queries the number of samples in the current setting of the acquisition memory. For sequence mode, this refers to the number if samples per segment, not to the number in the complete set.
Use MaxSamples to limit the number of samples acquired.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Obtain the number of points being used in the acquisition memory.
NumberOfPoints = app.Acquisition.Horizontal.NumPoints
MsgBox NumberOfPoints
```

TimePerPoint *Double*

Range From 5e-015 to 1e+012 step 1e-013

Description

Queries the time interval between successive samples in the acquisition.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Obtain the time per point of the acquisition.
timePerPt = app.Acquisition.Horizontal.TimePerPoint
MsgBox timePerPt
```

SampleMode *Enum*

Description

Sets/Queries the mode of acquisition as real-time or sequence or random interleaved sampling. Note that RIS mode and sequence mode are not available over the entire range of time-bases, and are not available simultaneously.
WaveExpert differences: CIS and SEQ are the only timebase modes.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the mode of acquisition to random interleaved sampling.
app.Acquisition.Horizontal.SampleMode = "RIS"
' WaveExpert example
app.Acquisition.Horizontal.SampleMode = "CIS"
```

Values

SEQ	

AcquisitionDuration *Double*

Range From 1e-012 to 1e+012 step 1e-015

Description

Queries the duration of the last completed acquisition. The result may depend on the spacing of the triggers in sequence mode, and it may depend on the number of averages when a channel is in averaging mode.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Obtain the duration of the last completed acquistion.
AcqDuration = app.Acquisition.Horizontal.AcquisitionDuration
MsgBox AcqDuration
```

MaxSamples *DoubleLockstep*

Range From 500 to 16000 step 10, locked to 1 2.5 5, fine grain allowed=true, on=false

Description

Sets/Queries the maximum permissible number of samples to be used in the acquisition memories. At the faster sample rates, the actual number used may be less than this maximum.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the available memory length per channel to 500
app.Acquisition.Horizontal.MaxSamples = 500
```

SamplingRate *Double*

Range From 500 to 1e+013 step (2 digits)

Description

Queries the sampling rate. This is the effective sampling rate of the traces, rather than the sample rate of the ADCs.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Inspect the effective sampling rate of the signal.
SamplingRate = app.Acquisition.Horizontal.SamplingRate
```

HorScaleVariable *Bool*

Description

Sets/Queries time base variable step On/Off

OperationMode *Enum*

Description

Sets/Queries the operation mode. In Sampling scopes, the scope can be configured for Scope, Eye or TDR modes of operation.

Values

Eye	Eye mode will continuously accumulate data for persistence type of display
Scope	Scope mode will run like a real time scope.
TDR	TDR mode should be used for TDR based measurements

Units *Enum*

Description

This control is applicable in Eye mode of operation only. It controls the timebase units of the output waveform.
1) When selected as Time, the horizontal unit will be in seconds
2) When selected as Bits, the horizontal unit will be in Uis. In this configuration, the unit interval of the output will be calculated using value of SignalBitrate control.

Values

Time	Horizontal unit in seconds

SignalStandardUI *Enum*

Description

This control selects signal standard. The value of signal bit rate will be updated on selection of new standard.

Values

1000BaseCX	
1000BaseLX	
1000BaseSX	
10G10.4	
10G9.9	
10GBASELX4	
10GbEFEC	
10GFCX4	
10XFC	
10XFCFEC	
DVI	
FBDIMM3.2Gbs	
FBDIMM4.0Gbs	
FBDIMM4.8Gbs	
FC1063	
FC133	
FC2125	
FC266	
FC4250	
FC531	
FEC10.664	
FEC10.709	
FEC12.5	
FSB533Mhz	
FSB667Mhz	
FSB800Mhz	
IEEE1394b	
Infini2.5Gbs	
InfiniBand	
OC1	
OC12	
OC192	
OC3	
OC48	
OC768	
PCIe1.0a	
PCIe1.0aMobile	
PCIe1.1	
PCIeCompliance	

PCIeCompliance1.1	
PCIeCompliance2.0	
RapidIOLPSerial	
RapidIOParallel	
SerialATA1.0a	
SerialAttachedSCSI	
STM16	
STM1Optical	
STM4Optical	
STS1Eye	
STS3	
USB2.0	
VSR5	
XAUI	

SignalBitRate ***Double***

Range From 1000 to 5e+011 step 1000

Description

Sets/Queries value of signal bit rate. When the Signal Standard is set to Custom, this controls sets the bit-rate value.

FindBitRate ***Action***

Description

Starts an Find Bit-rate operation. This operation operates only on the first visible channel. If no channels are visible, it operates on the first channel with sampling module.

CISStrobeFreq ***Double***

Range From 1 to 100 step 1e-012

Description

Queries actual sampling rate of the CIS time base. The value reflects hardeware sampling rate.

FPGAStatus ***Register***

Range Register 16 bit

Description

This control is intended for internal use only. Do not use this control.

Calibrate	Action
CalibratePod1	Bool
CalibratePod2	Bool
CalibratePod3	Bool
CalibratePod4	Bool
CancelCalibration	Action
PodsCalStartWizard	Action

PodsCalStartWizard *Action*

Description

This control initiates calibration wizards.
Note: This controls is intended for On screen Wizard only. It involves operator's selections. Do not use for fully Automated system.

Calibrate *Action*

Description

This control starts Calibration of selected modules.
Note: This controls is intended for On screen Wizard only. It involves operator's selections. Do not use for fully Automated system.

CancelCalibration *Action*

Description

This control cancels active calibration process.
Note: This controls is intended for On screen Wizard only. It involves operator's selections. Do not use for fully Automated system.

CalibratePod1 *Bool*

Description

This control enables/disables calibration for channel 1 Module.
Note: This controls is intended for On screen Wizard only. It involves operator's selections. Do not use for fully Automated system.

CalibratePod2 *Bool*

Description

This control enables/disables calibration for channel 2 Module.
Note: This controls is intended for On screen Wizard only. It involves operator's selections. Do not use for fully Automated system.

CalibratePod3 *Bool*

Description

This control enables/disables calibration for channel 3 Module.
Note: This controls is intended for On screen Wizard only. It involves operator's selections. Do not use for fully Automated system.

CalibratePod4 *Bool*

Description

This control enables/disables calibration for channel 4 Module.
Note: This controls is intended for On screen Wizard only. It involves operator's selections. Do not use for fully Automated system.

TRIGGER *app.Acquisition.Trigger*

This group of cvars controls all aspects of the trigger, except for trigger delay, which is in Acquisition.Horizontal.

Names of the form app.Acquisition.Trigger.Sources.xxxx are aliases of simpler names which are described in this section of the manual. Examples of alias pairs are as follows -
app.Acquisition.Trigger.Sources("Cx") is equivalent to app.Acquisition.Trigger.Cx
app.Acquisition.Trigger.Sources("Ext") is equivalent to app.Acquisition.Trigger.Ext
app.Acquisition.Trigger.Sources("Line") is equivalent to app.Acquisition.Trigger.Line
Please see under Acquisition.Channels("Cx") for a programming example.

Source	Enum
Type	Enum

Source *Enum*

Description

Sets/Queries the trigger source.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the trigger source to external.
```

```
app.Acquisition.Trigger.Source = "Ext"
```

Values

Prescaler	
Trigger	

Type *Enum*

Description

Sets/Queries the trigger type (mode).

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the trigger type to glitch.
app.Acquisition.Trigger.Type = "Glitch"
```

Values

EDGE	

CURSORS *app.Cursors*

This set of variables controls the cursor system.

Track	Bool
Type	Enum
View	Bool
XPos1	Double
XPos2	Double
YPos1	Double
YPos2	Double

View *Bool*

Description

Sets/Queries visibility of the cursors.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Make the cursors visible.
app.Cursors.View = "On"
```

Type *Enum*

Description

Sets/Queries the currently selected type of cursor.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the cursor type to vertical relative.
app.Cursors.View = "On"
app.Cursors.Type = "VertRel"
```

Values

HorizAbs	Single cursor, position specified in time
HorizRel	Dual cursors, positions specified in time
VertAbs	Single cursor, position specified in divisions vertically
VertRel	Dual cursors, positions specified in divisions vertically

Track *Bool*

Description

Sets/Queries the state of tracking of a pair of cursors. If tracking is enabled then when the first cursor is moved, the second will track at a constant distance from it.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set cursors tracking on.
app.Cursors.Track = True
```

XPos1 *Double*

Range From -1.79769e+308 to 1.79769e+308 step 0

Description

Sets/Queries the horizontal position of the first cursor, in the units of the horizontal variable.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the horizontal position of the first cursor to 50 ns.
app.Cursors.XPos1 = 50e-9
```

XPos2 *Double*

Range From -1.79769e+308 to 1.79769e+308 step 0

Description

Sets/Queries the horizontal position of the second cursor, in the units of the horizontal variable.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the horizontal position of the second cursor to 4.5 ms.
app.Cursors.XPos2 = 4.5e-3
```

YPos1 *Double*

Range From -3.99 to 3.99 step 0.01

Description

Sets/Queries the vertical position of the first cursor, in graticule divisions.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the vertical position of the first cursor.
app.Cursors.YPos1 = 3.4
```

YPos2 *Double*

Range From -3.99 to 3.99 step 0.01

Description

Sets/Queries the vertical position of the second cursor, in graticule divisions.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the vertical position of the second cursor.
app.Cursors.YPos2 = 2.1
```

DISPLAY *app.Display*

This set of variables controls the properties of the screen display of the instrument.

AxisLabels	Bool
Brightness	Integer
C1Color	Color
C1PrintColor	Color
C2Color	Color
C2PrintColor	Color
C3Color	Color
C3PrintColor	Color

C4Color	Color
C4PrintColor	Color
ClearSweeps	Action
DisplayMode	Enum
F1Color	Color
F1PrintColor	Color
F2Color	Color
F2PrintColor	Color
F3Color	Color
F3PrintColor	Color
F4Color	Color
F4PrintColor	Color
F5Color	Color
F5PrintColor	Color
F6Color	Color
F6PrintColor	Color
F7Color	Color
F7PrintColor	Color
F8Color	Color
F8PrintColor	Color
FactoryDefault	Action
GridIntensity	Integer
GridMode	Enum
GridOnTop	Bool
M1Color	Color
M1PrintColor	Color
M2Color	Color
M2PrintColor	Color
M3Color	Color
M3PrintColor	Color
M4Color	Color
M4PrintColor	Color
Persisted	Bool
Persistence3d	Bool
PersistenceLastTrace	Bool
PersistenceSaturation	Integer
PersistenceStyle	Enum
PersistenceTime	Enum
PreviewPrintColors	Action
ResetAll	Action
TraceStyle	Enum

GridMode *Enum*

Description

Sets/Queries the grid mode. The commands "Single" and "Dual", for example, set the grid mode until

countermanded. "Auto" allows the instrument to set the grid mode most suitable for the current number of visible traces.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Enter Octal grid mode
app.Display.GridMode = "Octal"
```

Values

Auto	Automatically choose grid mode, one trace per grid
Dual	Dual grid mode
Octal	Octal grid mode
Quad	Quad grid mode
Single	Single grid mode
XY	XY grid mode
XYDual	XY + Dual grid mode
XYSingle	XY + Single grid mode

TraceStyle *Enum*

Description

Sets/Queries the style in which traces are drawn.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Read the state of the persistence mode.
TraceStyle = app.Display.TraceStyle
```

Values

Line	Connect adjacent samples with straight lines
Points	Show only the sample points

GridIntensity *Integer*

Range From 0 to 100 step 1

Description

Sets/Queries the grid intensity as a percentage of the maximum value, with a resolution of 1%.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the grid intensity to 60% of the maximum.
app.Display.GridIntensity = 60
```

Brightness *Integer*

Range From 30 to 100 step 1

Description

LCD Display Brightness, only used on older WaveRunner 6000 DSOs.

GridOnTop *Bool*

Description

Sets/Queries whether the grid lines lie over the traces or vice versa.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the grid lines to be over the trace lines.
app.Display.GridOnTop = True
```

AxisLabels *Bool*

Description

Sets/Queries the visibility of the labels that show the horizontal and vertical limits of each grid.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Show the axis labels.
app.Display.AxisLabels = True
```

DisplayMode *Enum*

Description

Sets/Queries the display mode as either "Scope", showing the normal instrument screen, or "WebEdit", showing the web processor editing panel. Note that WebEdit mode is available only with certain software options, including XMATH and XMAP.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Switch to WebEdit mode
app.Display.DisplayMode = "WebEdit"
```

Values

Scope	

PersistenceStyle *Enum*

Description

Sets/Queries the type of persistence trace displayed.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the persistence style to color graded.
app.Display.PersistenceStyle = "ColorGraded"
```

Values

Analog	
ColorGraded	

Persisted *Bool*

Description

Sets/Queries whether persistence mode is in use. If the previously set persistence mode is per trace, the persisted cvar will be set as true by this command, even if none of the traces has been set to persistence mode.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Read the state of persistence mode.
Persist = app.Display.Persisted
```

Persistence3d *Bool*

Description

Sets/Queries whether the persistence 3-D mode is activated.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the 3-D display to off.
app.Display.Persistence3d = False
```

PersistenceSaturation *Integer*

Range From 0 to 100 step 1

Description

Sets/Queries the population level, relative to the maximum possible level, at which the persistence traces reach maximum intensity, and above which there are no further changes in color or intensity.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the persistence saturation level to 60%.
app.Display.PersistenceSaturation = 60
```

PersistenceTime *Enum*

Description

Sets/Queries decay time for trace persistence, expressed as a number of seconds, or as "infinity".

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the persistence time to 10 seconds.
app.Display.PersistenceTime = "10s"
```

Values

0.5s	
10s	
1s	
20s	
2s	
5s	
Infinite	

PersistenceLastTrace *Bool*

Description

Sets/Queries whether the last created trace is shown over the persistence trace.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the persistence display to show the last trace
' on top of the persistence trace.
app.Display.PersistenceLastTrace = True
```

ClearSweeps *Action*

Description

Initiates the Clear Sweeps operation. Clears history only for persistence traces, see the main Clear Sweeps control 'app.ClearSweeps', or the ClearSweeps control in other subsystems for other options.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Initiate a clear sweeps action for persistence traces.
app.Display.ClearSweeps
```

ResetAll *Action*

Description

Turns off persistence on any traces where it has been set on.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Reset all persistence traces to non-persisted mode.
app.Display.ResetAll
```

C1Color *Color*

Range From 0 to 16777215

Description

Sets/Queries the color of trace C1, using a number in the range 0 to FFFFFF in hexadecimal. The possible colors are made from any combination of the primary colors, which are set in hexadecimal as Blue = &HFF0000, Green = &HFF00, Red = &HFF. The value may be entered in decimal or in hexadecimal, though hexadecimal is usually more convenient. Note that if the intensity of a color is to be reduced or increased by a numerical factor, an AND operation must be used afterwards, to prevent corruption of other primary colors.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

Red = &Hff: Green = &H80: Blue = &H00

' Set the color of channel C1 trace to orange
app.Display.C1Color = (Blue * &H10000) + (Green * &H100) + Red
```

C1PrintColor *Color*

Range From 0 to 16777215

Description

Sets/Queries the color, in the printing palette, of trace C1, using a number in the range 0 to FFFFFF in hexadecimal. The primary colors are Blue = &HFF0000, Green = &HFF00, Red = &HFF in hexadecimal.
The value may be entered in decimal or in hexadecimal.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

Red = &Hff: Green = &H80: Blue = &H00

' Set the color of channel C1 trace to orange for printing.
app.Display.C1PrintColor = (Blue * &H10000) + (Green * &H100) + Red
```

C2Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

C2PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

C3Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

C3PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

C4Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

C4PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

F1Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

F1PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

F2Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

F2PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

F3Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

F3PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

F4Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

F4PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

F5Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

F5PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

F6Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

F6PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

F7Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

F7PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

F8Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

F8PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

M1Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

M1PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

M2Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

M2PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

M3Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

M3PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

M4Color *Color*

Range From 0 to 16777215

Description

Please see C1Color.

M4PrintColor *Color*

Range From 0 to 16777215

Description

Please see C1Printcolor.

FactoryDefault *Action*

Description

Restores the display of the instrument to the factory default settings

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Restore the display to the factory pre-set state.
app.Display.FactoryDefault
```

PreviewPrintColors *Action*

Description

Show the instrument display in the current color scheme selected for printing.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Show the current color scheme selected for printing.
app.Display.PreviewPrintColors
```

HARDCOPY *app.HardCopy*

This set of variables controls the transfer of information about the screen display to destinations such as such as disc files, internal memories, printers and remote computers.

Destination	Enum
EMailMessage	String
GridAreaOnly	Bool
HardcopyArea	Enum
Orientation	Enum
PreferredFilename	String
Print	Action
SelectedPrinter	Enum
StripChart	Bool
StripChartFactor	Enum
UseColor	Enum

Print *Action*

Description

Initiates a hard copy.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Initiate a hard copy.
app.Hardcopy.Print
```

UseColor *Enum*

Description

Defines the color scheme to be used when printing.

Values

BW	Optimized for black and white printers
Print	Use print colors (white background)
Std	As presented on DSO display

PreferredFilename *String*

Range Any number of characters

Description

Sets/Queries the preferred file name to use for hard copy.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the preferred filename to PrintFile.
app.Hardcopy.PreferredFilename = "PrintFile"
```

EMailMessage *String*

Range Any number of characters

Description

Sets/Queries the e-mail message.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Create the e-mail message - "Results for run 89".
app.Hardcopy.EMailMessage = "Results for run 89"
```

Orientation *Enum*

Description

Sets/Queries the orientation for hard copy to landscape.
Valid only when emitting to a printer as opposed to a file, the clipboard, or an E-Mail.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the orientation for hardcopy to landscape.
app.Hardcopy.Orientation = "Landscape"
```

Values

Landscape	
Portrait	

Destination *Enum*

Description

Sets/Queries the destination for hard copy.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the destination for hard copy to e-mail.
app.Hardcopy.Destination = "EMail"
```

Values

Clipboard	Send to clipboard for pasting into other applications
EMail	Send image in an E-Mail
File	Store image in a file
Printer	Print to a local, or networked printer
Remote	Special case used for remote printing, not usually used

SelectedPrinter *Enum*

Description

Sets/Queries the selection of the printer for hard copy. Note that whitespace and punctuation are removed from the string.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Select BarbondaleTintJet as the printer for hardcopy
app.Hardcopy.SelectedPrinter = "BarbondaleTintJet"
```

Values

HardcopyArea *Enum*

Description

Sets/Queries the area of the screen to be included in a hard copy.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Select the DSO screen area for hard copy.
app.Hardcopy.HardCopyArea = "DSOWindow"
```

Values

DSOWindow	Include only the DSO window
FullScreen	Include the full display screen
GridAreaOnly	Include the grid area only (doesn't include menus)

GridAreaOnly *Bool*

Description

Sets/Queries whether hard copy is of grid area only.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Read the status of Grid Area Only.
GridArea = app.Hardcopy.GridAreaOnly
```

StripChart *Bool*

Description

Sets/Queries the status of strip chart mode of printing. Valid only when emitting to the internal printer.

StripChartFactor *Enum*

Description

Sets/Queries the scale factor for strip chart printing. Valid only when emitting to the internal printer.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the strip chart scale to 5 cm/division.
app.Hardcopy.StripChartFactor = "5cmdiv"
```

Values

100cmdiv	
10cmdiv	
1cmdiv	
200cmdiv	
20cmdiv	
2cmdiv	
50cmdiv	
5cmdiv	

LABNOTEBOOK

app.LabNotebook

Provides access to the 'LabNotebook' feature. This allows the entire scope state (Waveforms, Setups, Display Images) to be stored, annotated, recalled, emailed, etc.

AttachFilesToEMail	Bool
BackupDatabase	Action
BackupFilename	String
BackupFolder	FileName
BackupToFolder	Action
BackupToMemoryStick	Action
ClearFilter	Action
CompactDatabase	Action
ConnectToFPHardCopy	Bool
CreateReport	Action
DeleteAll	Action
DeleteRecord	Action
EMailRecord	Action
FilterRecords	Action
FlashBackToRecord	Action
Format	Enum
HardcopyArea	Enum
InternalView	Action
MyLabNotebookMD	FileName
NextRecord	Action
PreviousRecord	Action
PrintRecord	Action
PromptBeforeSaving	Bool

RecordList	Enum
ReportLogo	FileName
ReportsDirectory	FileName
Save	Action
ScribbleBeforeSaving	Bool
StartNew	Action
UseDefaultLogo	Bool
UseDefaultTemplate	Bool
UsePrintColor	Bool
ViewRecord	Action
XSLTemplate	FileName

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Save the current state of the DSO into the Notebook
app.LabNotebook.ScribbleBeforeSaving = False
app.LabNotebook.PromptBeforeSaving = False
app.LabNotebook.Save

' Create a PDF report, and store it in the root of drive
C:app.LabNotebook.ReportsDirectory = "C:\"
app.LabNotebook.Format = "PDF"
app.LabNotebook.CreateReport

' Send the report in an email
app.Preferences.Email.Mode = "SMTP"
app.Preferences.Email.DefaultRecipient = "somebody@somewhere.com"
app.LabNotebook.EMailRecord
```

RecordList *Enum*

Description

The list of entries in the Lab Notebook, named using a GUID.

Values

NextRecord *Action*

Description

Move to (select) the next entry in the notebook.

PreviousRecord *Action*

Description

Move to (select) the previous entry in the notebook.

Save *Action*

Description

Initiate the creation of a new Lab Notebook entry.

FilterRecords *Action*

Description

Popup a dialog proposing various filtering methods, including date, and/or keword based filters.

ClearFilter *Action*

Description

Clear the NoteBook entry filter.

InternalView *Action*

Description

View the selected Lab Notebook entry within the DSO's graticule area.
Contrast with the 'ViewRecord' control, which presents the selected Lab Notebook entry in an external HTML browser.

CreateReport *Action*

Description

Create a report (PDF/RTF/HTML) of the currently selected notebook entry.

FlashBackToRecord *Action*

Description

Restore (FlashBack) the scope to the state that it was in when the current lab notebook entry was saved.
This may include the setup, and active waveforms.

DeleteRecord *Action*

Description

Delete the currently selected LabNotebook record.
Note that this will popup a modal dialog requesting confirmation.

DeleteAll *Action*

Description

Delete all LabNotebook entries.
Note that this action will popup a modal dialog, requesting confirmation.

Format *Enum*

Description

File Format in which exported reports are saved.

Values

HTML	
PDF	Adobe Acrobat file
RTF	Rich-text file (MS Wordpad, Word, etc.)

ViewRecord *Action*

Description

View the selected Lab Notebook entry in an external HTML browser.
Contrast with the 'ViewInternal' control, which presents the selected Lab Notebook entry within the DSO's graticule area.

PrintRecord *Action*

Description

Print the selected Lab Notebook entry. This action will present a popup allowing the target printer to be selected .

EMailRecord *Action*

Description

Email the currently selected record, in the selected format (PDF/RTF/HTML), to the recipient specified in the email setup (app.Preferences.Email).

PromptBeforeSaving *Bool*

Description

If true, the DSO will prompt the interactive user for a summary, and description, before the notebook entry is created.

ScribbleBeforeSaving *Bool*

Description

If True, the DSO will allow the interactive user to 'scribble' (annotate) the report page before it is saved.

ConnectToFPHardCopy *Bool*

Description

If True, the front-panel 'Print Screen' button is overridden to create a LabNotebook entry instead of its normal function.

UsePrintColor *Bool*

Description

If True, 'print colors' are used when storing an image of the display. These use a white background, as opposed to black, to save toner/ink.

HardcopyArea *Enum*

Description

Defines the region of the display that is stored when creating a new entry in the notebook.

Values

DSOWindow	Contents of DSO window, incl. dialog + menu bar
FullScreen	Entire windows display area
GridAreaOnly	Grid area only

AttachFilesToEMail *Bool*

Description

If true, the DSO Setup, and all enabled waveforms will be attached to any emailed report.

ReportsDirectory *FileName*

Range Any number of characters

Description

The directory in which Lab Notebook reports are created.

XSLTemplate *FileName*

Range Any number of characters

Description

Filename of the XSL template used in creating reports from Lab Notebook pages.

ReportLogo *FileName*

Range Any number of characters

Description

Contains the full pathname of the logo which will appear on Lab Notebook pages.

UseDefaultLogo *Bool*

Description

If True, the default logo is used on Lab Notebook pages. If False, the logo specifeid by the ReportLogo control is used instead.

UseDefaultTemplate *Bool*

Description

If True, the default xsl template is used when creating reports from LabNotebook pages.
If False, the template file specified by the XSLTemplate control is used instead.

CompactDatabase *Action*

Description

Compact the LabNotebook database. Useful if entries have been deleted from the database, to reclaim disk space.

MyLabNotebookMD *FileName*

Range Any number of characters

Description

Filename of the currently active Lab Notebook database.

StartNew *Action*

Description

Start a new Lab Notebook. This action will prompt the interactive user for the filename of the new Lab Notebook database file.

BackupDatabase *Action*

Description

Backup the current LabNotebook database. Note that this control will present a modal dialog, prompting for the backup filename and folder.
Use the BackupToFolder control to skip the dialog.

BackupToFolder *Action*

Description

Create a backup of the current LabNotebook database into the file specified by the BackupFolder/BackupFilename controls.

BackupToMemoryStick *Action*

Description

Create a backup of the current LabNotebook database into a file on an attached memory stick.

BackupFilename *String*

Range Any number of characters

Description

Contains the filename into which the LabNotebook is stored, when the BackupDatabase request is made.

BackupFolder *FileName*

Range Any number of characters

Description

Contains the folder into which the LabNotebook is stored, when the BackupDatabase request is made.

MATH *app.Math*

Variables of the form app.Math.xxxx control the mathematical functions F1 through F8.

Names of the form app.Math.Functions("Fx").xxxx are aliases of simpler names which are described in this section of the manual. Examples of alias pairs are as follows -

app.Math.Functions("Fx") is equivalent to app.Math.Fx
app.Math.Functions("Fx").Out.Result is equivalent to app.Math.Fx.Out.Result
app.Math.Functions("Fx").Zoom is equivalent to app.Math.Zoom.Fx

Please see under Acquisition.Channels for a programming example.

ClearSweeps	Action
ResetAll	Action
ShowZoomMenu	Action

ClearSweeps *Action*

Description

Clear sweeps for history functions such as average, histogram and trend. See also the general 'app.ClearSweeps' control which clears accumulated data for all subsystems, including persistence, measurement statistics, etc.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Clear sweeps for all history functions.
app.Math.ClearSweeps
```

ResetAll *Action*

Description

Reset the math subsystem to its default state.
All currently selected math operators, and other settings will be lost.

ShowZoomMenu *Action*

Description

Present the Zoom setup Menu (to setup the Zoom (Z) traces).

FUNCTIONS *app.Math.Functions*

Names of the form app.Math.Functions("Fx").xxxx are aliases of simpler names which are described in the section of the manual which is devoted to app.Math. Examples of alias pairs are as follows -

app.Math.Functions("Fx") is equivalent to app.Math.Fx
app.Math.Functions("Fx").Out.Result is equivalent to app.Math.Fx.Out.Result
app.Math.Functions("Fx").Zoom is equivalent to app.Math.Zoom.Fx

Please see under Acquisition.Channels for a programming example.

FX *app.Math.Fx*

This set of variables controls the math functions F1 through F8.

ClearSweeps	Action
DoResetZoom	Action
DoStoreToMemoryTrace	Action
Equation	String
LabelsPosition	String
LabelsText	String
MathMode	Enum

Operator1	Enum
Persisted	Bool
PersistenceSaturation	Integer
PersistenceTime	Enum
ShowLastTrace	Bool
Source1	Enum
UseGrid	String
View	Bool
ViewLabels	Bool

ClearSweeps *Action*

Description

Clears accumulated data for a single function trace.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Reset accumulation for trace F1
app.Math.F1.ClearSweeps
```

UseGrid *String*

Range Any number of characters

Description

Sets/Queries the grid in use for the math trace Fx.
See also app.Acquisition.Cx.UseGrid.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Place math trace F3 on grid YT4.
app.Math.F3.UseGrid = "YT4"
```

Persisted *Bool*

Description

Sets/Queries the persisted state of the function waveform. If the Display.LockPersistence control is set to 'AllLocked' then the persisted state of all displayed waveforms will be the same. If the Display.LockPersistence control is set to 'PerTrace' then the persisted state of each waveform may be independently controlled.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set persistence on for trace F3.
app.Math.F3.Persisted = True
```

PersistenceSaturation *Integer*

Range From 0 to 100 step 1

Description

Sets/Queries the saturation threshold for persisted waveforms. All information at this level or above will be recorded with the same color or intensity. See the general description above for a discussion of the locked and unlocked persistence modes.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the persistence saturation level for trace F1.
app.Math.F1.PersistenceSaturation = 40
```

ShowLastTrace *Bool*

Description

Sets/Queries the state of the Show Last Trace control. If True then when this trace is displayed in persistence mode the last acquired waveform will be superimposed on the accumulating persistence map. See the general description above for a discussion of the locked and unlocked persistence modes.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Do not show the last trace for the persistence trace of trace F1.
app.Math.F1.ShowLastTrace = False
```

PersistenceTime *Enum*

Description

Sets/Queries the state of the Persistence Time control. Controls the persistence decay time for this trace. See the general description above for a discussion of the locked and unlocked persistence modes.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the persistence time for the trace F1 to 10 seconds.
app.Math.F1.PersistenceTime = "10s"
```

Values

0.5s	
10s	
1s	
20s	
2s	
5s	
Infinite	

LabelsText *String*

Range Any number of characters

Description

Sets / Queries the text that appears in labels attached to acquisition trace Cx. Multiple labels may be specified by using comma as a delimiter. See the documentation on LabelsPosition for an example of use.

LabelsPosition *String*

Range Any number of characters

Description

Sets / Queries the horizontal position of the label attached to the trace Fx. The unit of measurement is the unit of the horizontal scale. The measurement is made from the trigger point.
Note that this control is a string, not a numeric value. This allows multiple labels to be positioned, as shown in the example below.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Add a couple of labels to trace F1, one at 0ns, and one at 55ns
app.SetToDefaultSetup
app.Math.F1.View = True
app.Math.F1.ViewLabels = True
app.Math.F1.LabelsPosition = "0.0, 55e-9"
app.Math.F1.LabelsText = "Hello,World"
```

ViewLabels *Bool*

Description

Sets/Queries whether trace labels, defined with LabelsText and LabelsPosition controls, are shown.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Show the user-defined trace label for trace F1
app.Math.F1.ViewLabels = True
```

MathMode *Enum*

Description

Sets/Queries the math mode.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the mode of the math function F1
app.Math.F1.MathMode = "TwoOperators"
```

Values

Graphing	Graphing mode, chain a measurement and a graphing operator
OneOperator	Single math operator
TwoOperators	Chain two math operators

Equation *String*

Range Any number of characters

Description

Queries the equation which defines the math function Fx.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Read the definition of math function F3.
EquationF3 = app.Math.F3.Equation
MsgBox EquationF3
```

Operator1 *Enum*

Description

Sets/Queries the first operator of math function Fx. When MathMode = "OneOperator, this is the only math operator, when MathMode = "TwoOperator", this is the first of two operators. Note that when MathMode = "Graph", this control has no effect.
Note also that the list of available math operators varies depending upon the instrument model number, and the list of installed software options.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Define the first operator of math function F1 as an Average
app.Math.F1.View = True
app.Math.F1.MathMode = "OneOperator"
app.Math.F1.Operator1 = "Average"
```

Values

AbsoluteValue	
Average	
Copy	
Derivative	
Deskew	
Difference	
EnhancedResolution	
Envelope	
Exp	
Exp10	
FFT	
Floor	
Histogram	
Integral	
Invert	
Ln	
Log10	
MATLABWaveform	
Null	
PersistenceHistogram	
Product	
Ratio	
Reciprocal	
Rescale	
Roof	
SinXOverX	
Square	
SquareRoot	
Sum	
Trend	
Zoom	

Source1 *Enum*

Description

Sets/Queries the first source of the first operator in Fx. Note that the two possible sources of Operator1 are Source1 and Source2, Source3 is the second source to Operator2, with the first source of Operator2 being the ouput of Operator1.
Note that the list of available sources is dependent upon the instrument model, and it's installed software options.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Define the first source of math function F1 as C3.
app.Math.F1.Source1 = "C3"
```

Values

C1	
C2	
C3	
C4	
F2	
F3	
F4	
F5	
F6	
F7	
F8	
M1	
M2	
M3	
M4	
P1	
P2	
P3	
P4	
P5	
P6	
P7	
P8	
ScanHisto	
ScanOverlay	
TDRN	
Z1	
Z2	
Z3	
Z4	

DoResetZoom *Action*

Description

Resets the zoom state of math trace Fx.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Reset zoom of math function F3.
app.Math.F3.DoResetZoom
```

View *Bool*

Description

Sets/Queries whether the trace of math function Fx is visible. Note that even when math traces are not visible, but are being used as inputs to other math functions and/or measurements, they are computed.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Show math trace F3.
app.Math.F3.View = True
```

DoStoreToMemoryTrace *Action*

Description

Store data from math function Fx to a memory trace.
Destination for F1 will be M1, F2 will be M2, etc.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Store math function F2 to a memory trace.
app.Math.F2.DoStoreToMemoryTrace
```

OPERATOR1SETUP

app.Math.Fx.Operator1Setup

This node is dynamically created, and will contain the controls for the operator currently selected into Operator1.
See the Math/Measure Control reference at the end of this manual for a lost of these controls.

RESULT

app.Math.Fx.Out.Result

Properties of the type xxxx.Out.Result.yyyy are those of the last completed acquisition. They are not affected if other cvars are changed after that acquisition was completed. This distinction between "Out.Result" properties and other cvars is most important when the trigger mode is Single or Stopped. You should treat "Out.Result" properties as read-only.

For a detailed description of all properties available for the output of a Math Function, please see Chapter 1.

ZOOM

app.Math.Fx.Zoom

This set of variables controls the zoom functions for math trace Fx.

HorPos	Double
HorZoom	Double
ResetZoom	Action
VariableHorZoom	Bool
VariableVerZoom	Bool
VerPos	Double
VerZoom	Double

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Turn on trace F1, will default to Zoom-Only
app.Math.F1.View = True
app.Math.F1.Source1 = "C1"

' Zoom trace F1 by a factor of 2 horizontally and vertically
app.Math.F1.Zoom.Rese
```

ResetZoom

Action

Description

Reset the trace Fx to x1 zoom and zero offset in both axes, so that it is identical to its input trace.

VariableVerZoom

Bool

Description

Sets/Queries the ability to zoom vertically by a continuously variable factor. Note that if a vertical zoom of 0.9 is set, while variable zoom is off, the vertical zoom will be set to 1.0. If the variable zoom is then enabled, the factor of 0.9 will have been remembered, and it will be used. Note that the previous value will not be remembered during a power-cycle.

VariableHorZoom ***Bool***

Description

Sets/Queries the ability to zoom horizontally by a continuously variable factor. Note that if a horizontal zoom of 0.9 is set, while variable zoom is off, the horizontal zoom will be set to 1.0. If the variable zoom is then enabled, the factor of 0.9 will have been remembered, and it will be used. Note that the previous value will not be remembered during a power-cycle.

VerZoom ***Double***

Range From 0.1 to 100 step (8 digits)

Description

Sets/Queries the vertical magnification of the trace Fx. The magnification will be in a 1 2 5 10 sequence unless VariableVerZoom has been set to True, in which case it will be continuously variable.

VerPos ***Double***

Range From -1.5 to 1.5 step (8 digits)

Description

Sets/Queries the vertical position of center of the grid on the zoomed trace Fx. The unit of measurement is the screen height, that is, 0.375 means a shift of three of the eight divisions. A positive value moves the trace downwards.

HorZoom ***Double***

Range From 0.1 to 1e+006 step (8 digits)

Description

Sets/Queries the horizontal magnification of the trace Fx. The magnification will be in a 1 2 5 10 sequence unless variable horizontal magnification has been set.

HorPos ***Double***

Range From -0.5 to 0.5 step (8 digits)

Description

Sets/Queries the horizontal position of center of the grid on the zoomed trace Fx. The unit of measurement is the screen width, that is, 0.3 means a shift of three of the ten divisions. A positive value moves the trace to the left.

This set of variables controls the display of data in X vs. Y mode. Only Valid when the instrument is in XY, XYSingle, or XYDual display modes.

ClearSweeps	Action
InputX	Enum
InputY	Enum
Persisted	Bool
PersistenceSaturation	Integer
PersistenceTime	Enum
PersistenceViewOn	Bool
ShowLastTrace	Bool

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Switch to XY+Dual Grid Mode
app.Display.GridMode = "XYDual"

' Configure XY to show C1 vs. C2 in 3D Persistence mode
app.Math.XY.InputX = "C1"
app.Math.XY.InputY = "C2"
app.Math
```

ClearSweeps *Action*

Description

Clears persistence X-Y plot.

Persisted *Bool*

Description

Sets/Queries the persisted state of the X-Y plot. If the Display.LockPersistence control is set to 'AllLocked' then the persisted state of all displayed waveforms will be the same. If the Display.LockPersistence control is set to 'PerTrace' then the persisted state of each waveform may be independently controlled.

PersistenceSaturation *Integer*

Range From 0 to 100 step 1

Description

Sets/Queries the saturation threshold for persisted X-Y plot. All information at this level or above will be recorded with the same color or intensity. See the general description above for a discussion of the locked and unlocked persistence modes.

ShowLastTrace ***Bool***

Description

Sets/Queries the state of the Show Last Trace control. If True then when this trace is displayed in persistence mode the last acquired waveform will be superimposed on the accumulating persistence map.
See the general description above for a discussion of the locked and unlocked persistence modes.

PersistenceTime ***Enum***

Description

Sets/Queries the state of the Persistence Time control. Controls the persistence decay time for the Xy persistence. See the general description above for a discussion of the locked and unlocked persistence modes.

Values

0.5s	
10s	
1s	
20s	
2s	
5s	
Infinite	

PersistenceViewOn ***Bool***

Description

Query the state of the persistence view.

InputX *Enum*

Description

Sets/Queries the name of the input channel for the X axis of the X-Y plot.

Values

C1	
C2	
C3	
C4	
F1	
F2	
F3	
F4	
F5	
F6	
F7	
F8	
M1	
M2	
M3	
M4	
ScanHisto	
ScanOverlay	
TDRN	
Z1	
Z2	
Z3	
Z4	

InputY *Enum*

Description

Sets/Queries the name of the input channel for the Y axis of the X-Y plot.

Values

C1	
C2	
C3	
C4	
F1	
F2	
F3	
F4	
F5	
F6	
F7	
F8	
M1	
M2	
M3	
M4	
ScanHisto	
ScanOverlay	
TDRN	
Z1	
Z2	
Z3	
Z4	

RESULT

app.Math.XY.Out.Result

Properties of the type xxxx.Out.Result.yyyy are those of the last completed acquisition. They are not affected if other cvars are changed after that acquisition was completed. This distinction between "Out.Result" properties and other cvars is most important when the trigger mode is Single or Stopped. You should treat "Out.Result" properties as read-only.

Note that this XY result object is very similar, but not identical to the result object exposed by the channel and math traces. The differences are due to the fact that the XY trace returns pairs of data values, one for X, one for Y.

For a detailed description of all properties available for the output of an XY trace, please see Chapter 1.

MEASURE

app.Measure

Variables of the form app.Measure control the parameters P1 through P8, and their associated statistical results and

histicons.

Names of the forms app.Measure.Measure("Premote").xxxx and app.Measure.Measure("Px").xxxx are aliases of simpler names which are described in this section of the manual. Examples of alias pairs are as follows -
app.Measure.Measure("Premote").OutResult is equivalent to app.Measure."Premote".OutResult
app.Measure.Measure("Px").Statistics is equivalent to app.Measure.Px.Statistics
Please see under Acquisition.Channels for a programming example.

ClearAll	Action
ClearAllHelpMarkers	Action
ClearSweeps	Action
DataCoding	Enum
EyeAperture	Double
EyeSource1	Enum
EyeSource2	Enum
HelpMarkers	Enum
HistoOn	Bool
MeasureSet	Enum
SetGateToDefault	Action
ShowAllHelpMarkers	Action
ShowEyeMarkers	Bool
ShowMeasure	Bool
StatsOn	Bool
StdGateStart	Double
StdGateStop	Double
StdSource	Enum
UseCustomThresh	Bool

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' get into the custom parameter mode
app.Measure.MeasureMode = "MyMeasure"
app.Measure.ClearAll
app.Measure.StatsOn = True
app.Measure.HistoOn = False

' Configure P1 to measure a
```

StatsOn *Bool*

Description

Sets/Queries the visibility of parameter statistics. Note: the statistics are accumulated whether the view of them is on or not, so you needn't have StatsOn = "On" to collect statistics.

HistoOn *Bool*

Description

Sets/Queries the visibility of the histicons of the parameters which are viewed.

StdSource *Enum*

Description

Sets/Queries the channel which is the source of ALL standard voltage or time parameters. Note that when in 'My Measure' mode each parameter has it's own Source selection, and this setting is ignored.

Values

C1	
C2	
C3	
C4	
F1	
F2	
F3	
F4	
F5	
F6	
F7	
F8	
M1	
M2	
M3	
M4	
ScanHisto	
ScanOverlay	
TDRN	
Z1	
Z2	
Z3	
Z4	

HelpMarkers *Enum*

Description

Sets/Queries the level of detail for help markers (if indeed any of the selected parameter definitions have help markers).

These markers are displayed on the source traces, and only if those traces are viewed simultaneously with the parameter measurements.

Note: this setting is global for all Px

Values

Detailed	Detailed help markers
Off	No help markers
Simple	Simple help markers

MeasureSet *Enum*

Description

Defines the mode in which the measurement system is working.

Values

EyeMeasure	
MyMeasure	Custom list, each chosen from the list of available measurements.
StdHorizontal	Standard Horizontal Measurements
StdVertical	Standard Vertical Measurements

ShowMeasure *Bool*

Description

Defines whether the measure results table is displayed or not.

ShowAllHelpMarkers *Action*

Description

Force all 'HelpMarkers' on.
(HelpMarkers are the on-trace annotation of measurement setup and results)

ClearAllHelpMarkers *Action*

Description

Force all 'HelpMarkers' off.
(HelpMarkers are the on-trace annotation of measurement setup and results)

StdGateStart *Double*

Range From 0 to 10 step 0.01

Description

Sets/Queries the position of the left hand limit of the measure gate (n divisions).
Valid only when in either Std. Vertical or Std. Horizontal measurement modes. For MyMeasure see the equivalent controls under Px.

StdGateStop *Double*

Range From 0 to 10 step 0.01

Description

Sets/Queries the position of the right hand limit of the measure gate (in divisions). Valid only when in either Std. Vertical or Std. Horizontal measurement modes. For MyMeasure see the equivalent controls under Px.

ClearSweeps *Action*

Description

Clears the accumulated statistics for parametersP1 to P8 as well as the acumulated statistics for their associated histicons.

ClearAll *Action*

Description

Resets all parameter setups, turning each of the parameters view to "off", the MeasurementType to "measure"and the selected paramEngine to "Null".

SetGateToDefault *Action*

Description

Sets the measure gate to its default state. Valid only when in either Std. Vertical or Std. Horizontal measurement modes. For MyMeasure see the equivalent controls under Px.

DataCoding *Enum*

Description

Data coding standard for measurements made in 'Eye Measure' mode.

Values

NRZ	
RZ	

UseCustomThresh *Bool*

Description

If True, the eye aperature may be specified when in 'Eye Measure' mode.

EyeAperture *Double*

Range From 0 to 100 step 0.1

Description

Eye Aperature, for measurements in 'Eye Measure' mode. Valid only when UseCustomThresh is set to true.

ShowEyeMarkers *Bool*

Description

Set to true to show marker annotation when in 'Eye Measure' mode.

EyeSource1 *Enum*

Description

First source trace for measurements made in 'Eye Measure' mode.

Values

C1	
C2	
C3	
C4	
M1	
M2	
M3	
M4	

EyeSource2 *Enum*

Description

Second source trace for measurements made in 'Eye Measure' mode.

Values

C1	
C2	
C3	
C4	
M1	
M2	
M3	
M4	

MEASURE

app.Measure.Measure

Names of the forms app.Measure.Measure("Premote").xxxx and app.Measure.Measure("Px").xxxx are aliases of simpler names which are described in the section of the manual which is devoted to app.Measure. Examples of alias pairs are as follows -

app.Measure.Measure("Premote").OutResult is equivalent to app.Measure."Premote".OutResult
app.Measure.Measure("Px").Statistics is equivalent to app.Measure.Px.Statistics

Please see under app.Acquisition.Channels("Cx") for a programming example.

PX

app.Measure.Px

This set of variables controls the parameters P1 through P8, (when the MeasureMode is "MyMeasure", otherwise these are predefined) and the statistical results and histicons which depend on them.

ClearSweeps	Action

GateByRange	Bool
GateByWform	Bool
GateStart	Double
GateStop	Double
HelpAlwaysOn	Bool
MeasurementType	Enum
ParamEngine	Enum
Source1	Enum

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

app.Measure.MeasureMode = "MyMeasure"

' Set parameter P1 to math on parameters.
App.Measure.P1.MeasurementType = "math"
```

ClearSweeps *Action*

Description

Clear any accumulated results for this paramater only.

HelpAlwaysOn *Bool*

Description

Defines whether Help Marters are always displayed for this measurement, even when the measurement dialog is closed.

MeasurementType *Enum*

Description

Sets/Queries the measurement type of the parameter Px.

Values

EyeMeasure	
measure	Standard measurement mode (parametric of a trace waveform)

ParamEngine *Enum*

Description

Sets/Queries the parameter (measurement on a trace) for Px. This setting applies only if the MeasurementType control is set to "measure".

Values

Amplitude	
Area	
Base	
Cycles	
Delay	
DeltaDelay	
DeltaTimeAtLevel	
DeltaTriggerTime	
Duration	
DutyCycle	
ExtinctionRatio	
EyeACRMS	
EyeAmplitude	
EyeBER	
EyeBitRate	
EyeBitTime	
EyeCrossing	
EyeCrossingNegative	
EyeCrossingPositive	
EyeCyclicArea	
EyeDelay	
EyeDeltaDelay	
EyeFallTime	
EyeHeight	
EyeMean	
EyeOneLevel	
EyeOpeningFactor	
EyeOvershootNegative	
EyeOvershootPositive	
EyePkpkJitter	
EyePkPkNoise	
EyePulseWid	
EyeQFactor	
EyeRiseTime	
EyeRMSJitter	
EyeSDEVNoise	
EyeSignalToNoise	
EyeSuppressionRatio	
EyeWidth	

EyeZeroLevel	
Fall	
Fall8020	
FallAtLevel	
FirstPoint	
Frequency	
FullWidthAtHalfMaximum	
FullWidthAtXX	
HistogramAmplitude	
HistogramBase	
HistogramMaximum	
HistogramMean	
HistogramMedian	
HistogramMid	
HistogramMinimum	
HistogramRms	
HistogramSdev	
HistogramTop	
LastPoint	
LevelAtX	
MATLABParameter	
Maximum	
MaximumPopulation	
Mean	
Median	
Minimum	
Mode	
npoints	
Null	
OvershootNegative	
OvershootPositive	
Peaks	
PeakToPeak	
Percentile	
Period	
PersistArea	
PersistDCD	
PersistDutyCycle	
PersistMax	
PersistMean	
PersistMid	
PersistMin	
PersistPkPk	
PersistPulseSymmetry	
PersistRMS	
Phase	

PopulationAtX	
Range	
Rise	
Rise2080	
RiseAtLevel	
RootMeanSquare	
StandardDeviation	
TDRCapInd	
TIE	
TimeAtLevel	
Top	
TotalPopulation	
Width	
WidthNegative	
XAtMaximum	
XAtMinimum	
XAtPeak	

Source1 *Enum*

Description

Sets/Queries the first trace source of the parameter Px. Used only when MeasurementType = "measure", for MeasurementType = "math", refer to PSource1.

Values

C1	
C2	
C3	
C4	
F1	
F2	
F3	
F4	
F5	
F6	
F7	
F8	
M1	
M2	
M3	
M4	
ScanHisto	
ScanOverlay	
TDRN	
Z1	
Z2	
Z3	
Z4	

GateStart *Double*

Range From 0 to 10 step 0.01

Description

Sets/Reads the position of the left hand edge of the measure gate for parameter Px.

GateStop *Double*

Range From 0 to 10 step 0.01

Description

Sets/Reads the position of the right hand edge of the measure gate for parameter Px.

GateByWform *Bool*

Description

If True, measurements are gated by the state of the waveform defined by the WformSource control.

GateByRange *Bool*

Description

If True, only measurements who's value(s) fall between the limits defined by the LowerLimit and UpperLimit controls, are accepted.

RESULT *app.Measure.Px.histo.Result*

RESULT *app.Measure.Px.last.Result*

RESULT *app.Measure.Px.max.Result*

RESULT *app.Measure.Px.mean.Result*

RESULT *app.Measure.Px.min.Result*

RESULT

app.Measure.Px.num.Result

OPERATOR

app.Measure.Px.Operator

This path specifies that the selected ParamEngine or ArithEngine control variables are "here"

RESULT

app.Measure.Px.Out.Result

Properties of the type xxxx.Out.Result.yyyy are those of the last completed acquisition. They are not affected if other cvars are changed after that acquisition was completed. This distinction between "Out.Result" properties and other cvars is most important when the trigger mode is Single or Stopped. You should treat "Out.Result" properties as read-only.

RESULT

app.Measure.Px.sdev.Result

STATISTICS

app.Measure.Px.Statistics

This set of variables controls the statistical summaries that are provided for all the parameters.

MEMORY

app.Memory

Variables of the form app.Memory.xxxx control the memories M1 through M4.

Names of the form app.Memory.Memories("Mx").xxxx are aliases of simpler names which are described in this section of the manual. Examples of alias pairs are as follows -

app.Memory.Memories("Mx").Out.Result is equivalent to app.Memory.Mx.Out.Result

app.Memory.Memories("Mx").Zoom is equivalent to app.Memory.Mx.Zoom
Please see under app.Acquisition.Channels("Cx") for a programming example.

ClearAllMem	Action

ClearAllMem *Action*

Description

Clears the contents of all trace memories.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Clear the contents of all trace memories.
app.Memory.ClearAllMem
```

MEMORIES *app.Memory.Memories*

Names of the form app.Memory.Memories("Mx").xxxx are aliases of simpler names which are described in the section of the manual which is devoted to app.Memory. Examples of alias pairs are as follows -
app.Memory.Memories("Mx").Out.Result is equivalent to app.Memory.Mx.Out.Result
app.Memory.Memories("Mx").Zoom is equivalent to app.Memory.Mx.Zoom
Please see under Acquisition.Channels for a programming example.

MX *app.Memory.Mx*

This set of variables controls the memories M1 through M4.

ClearMem	Action
Copy	Action
LabelsPosition	String
LabelsText	String
PersistenceViewOn	Bool
Source1	Enum
UseGrid	String
UserText	String
View	Bool
ViewLabels	Bool

View *Bool*

Description

Sets/Queries whether memory trace Mx is visible.

UseGrid *String*

Range Any number of characters

Description

Sets/Queries the grid used for memory trace Mx.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set memory trace M2 to use grid YT3.
app.Memory.M2.UseGrid = "YT3"
```

PersistenceViewOn *Bool*

Description

Query the state of the persistence view.

LabelsText *String*

Range Any number of characters

LabelsPosition *String*

Range Any number of characters

Description

Sets / Queries the horizontal position of the label attached to the acquisition trace Cx. The unit of measurement is the unit of the horizontal scale. The measurement is made from the trigger point. Note that this control is a string, not a numeric value. This allows multiple labels to be positioned, as shown in the example below.

ViewLabels *Bool*

Description

Sets/Queries whether labels are visible for trace Mx.

ClearMem *Action*

Description

Initiates a clear memory operation for memory Mx.

Copy *Action*

Description

Copy the trace specified by the Source1 control into this memory.

UserText *String*

Range Any number of characters

Description

Text field, used to attach arbitrary comments to a memory waveform.

Source1 *Enum*

Description

Source trace for Copy operations (see 'Copy' control)

Values

C1	
C2	
C3	
C4	
F1	
F2	
F3	
F4	
F5	
F6	
F7	
F8	
M2	
M3	
M4	
ScanHisto	
ScanOverlay	
TDRN	
Z1	
Z2	
Z3	
Z4	

RESULT

app.Memory.Mx.Out.Result

See app.Acquisition.Cx.Out.Result for a definition of methods and properties used to access the Mx waveform result.

ZOOM

app.Memory.Mx.Zoom

This set of variables controls zooming of the memory traces M1 through M4.

HorPos	Double
HorZoom	Double
ResetZoom	Action
VariableHorZoom	Bool
VariableVerZoom	Bool
VerPos	Double

VerZoom	Double

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Save C1 into M1
app.SaveRecall.Waveform.SaveTo = "Memory"
app.SaveRecall.Waveform.SaveSource = "C1"
app.SaveRecall.Waveform.SaveDestination = "M1"
app.SaveRecall.Waveform.DoSave
```

ResetZoom ***Action***

Description

Resets the zoom for trace Mx.

VariableVerZoom ***Bool***

Description

Sets/Queries the ability to zoom vertically by a continuously variable factor as opposed to a factor that follows a 1, 2, 5 sequence.

VariableHorZoom ***Bool***

Description

Sets/Queries the ability to zoom horizontally by a continuously variable factor as opposed to a factor that follows a 1, 2, 5 sequence.

VerZoom ***Double***

Range From 0.1 to 100 step (8 digits)

Description

Sets/Queries the vertical magnification of the trace Mx. The magnification will be in a 1 2 5 10 sequence unless variable vertical magnification has been set.

VerPos *Double*

Range From -1.5 to 1.5 step (8 digits)

Description

Sets/Queries the vertical position of center of the grid on the zoomed trace Mx. The unit of measurement is the screen height, that is, 0.375 means a shift of three of the eight divisions. A positive value moves the trace downwards.

HorZoom *Double*

Range From 0.1 to 1e+006 step (8 digits)

Description

Sets/Queries the horizontal magnification of the trace Mx. The magnification will be in a 1 2 5 10 sequence unless variable horizontal magnification has been set.

HorPos *Double*

Range From -0.5 to 0.5 step (8 digits)

Description

Sets/Queries the horizontal position of center of the grid on the zoomed trace Mx. The unit of measurement is the screen width, that is, 0.3 means a shift of three of the ten divisions. A positive value moves the trace to the left.

PASSFAIL *app.PassFail*

Names of the forms app.PassFail("Qremote").xxxx and app.PassFail("Qx").xxxx are aliases of simpler names which are described in this section of the manual. Examples of alias pairs are as follows -

app.PassFail.PassFail("Qremote").Operator is equivalent to app.PassFail.Qremote.Operator

app.PassFail.PassFail("Qx").Out.Result is equivalent to app.PassFail.Qx.Out.Result

Please see under app.Acquisition.Channels("Cx") for a programming example.

ActionOn	Enum
Alarm	Bool
EnableActions	Bool
PredefinedConditions	Enum
PrintScreen	Bool
Save	Bool
Stop	Bool
StopAfter	Integer
StopTesting	Bool
SummaryView	Bool
Testing	Bool

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Setup Parameter P1 to be the amplitude of C1
app.Measure.MeasureMode = "MyMeasure"
app.Measure.P1.ParamEngine = "Ampl"
app.Measure.P1.Source1 = "C1"
app.Measure.P1.View = True
```

Testing *Bool*

Description

Sets/Queries whether PassFail testing is on.

EnableActions *Bool*

Description

Sets/Queries whether the selected actions will be executed if the selected PassFail condition is met.

PredefinedConditions *Enum*

Description

Sets/Queries the logical criteria that must be met in a pass-fail test. For example, the condition AnyTrue means that the pass-fail criterion is met if at least one of the test conditions results in a True result.

Values

AllFalse	
AllQ1ToQ4OrAllQ5ToQ8	
AllTrue	
AnyFalse	
AnyQ1ToQ4AndAnyQ5T	
AnyTrue	

ActionOn *Enum*

Description

Sets/Queries whether a Pass condition or a Fail condition will initiate the pre-selected actions.

Values

Fail	
Pass	

Stop *Bool*

Description

Sets/Queries whether Stop is included in the PassFail actions.

Alarm *Bool*

Description

Sets/Queries whether Alarm is included in the PassFail actions.

PrintScreen *Bool*

Description

Sets/Queries whether Print Screen is included in the PassFail actions.

Save *Bool*

Description

Sets/Queries whether Save is included in the PassFail actions.

StopTesting *Bool*

Description

If Enabled, testing will stop after a number of sweeps defined by the StopAfter control.

StopAfter *Integer*

Range From 1 to 1000000000 step 1

Description

Sets/Queries the maximum number of sweeps that will be acquired before testing is halted.

SummaryView *Bool*

Description

Summary view

RESULT *app.PassFail.LastPass.Result*

RESULT *app.PassFail.NumPassed.Result*

QX *app.PassFail.Qx*

This set of variables controls the tests Q1 through Q8 in the pass fail system.

ClearSweeps	Action
ConditionEngine	Enum
Equation	String
PSource1	Enum
ShortDescription	String
View	Bool
WSource2	Enum

View *Bool*

Description

Sets/Queries whether pass-fail test Qx is visible.

ClearSweeps *Action*

Description

ClearSweeps

ConditionEngine *Enum*

Description

Sets/Queries whether pass-fail test Qx uses mask testing or parameter comparison.

Values

DualParamCompare	
MaskTestCondition	
ParameterCompare	

Equation *String*

Range Any number of characters

Description

Inspects the equation for pass-fail test Qx. A typical equation would be "All P3 < 0.7071".

ShortDescription *String*

Range Any number of characters

Description

ShortDescription

PSource1 *Enum*

Description

PSource1

Values

P1	
P2	
P3	
P4	
P5	
P6	
P7	
P8	

WSource2 *Enum*

Description

WSource2

Values

C1	
C2	
C3	
C4	
F1	
F2	
F3	
F4	
F5	
F6	
F7	
F8	
M1	
M2	
M3	
M4	
ScanHisto	
ScanOverlay	
TDRN	
Z1	
Z2	
Z3	
Z4	

RESULT

app.PassFail.Qx.Out.Result

Properties of the type xxxx.Out.Result.yyyy are those of the last completed acquisition. They are not affected if other cvars are changed after that acquisition was completed. This distinction between "Out.Result" properties and other cvars is most important when the trigger mode is Single or Stopped. You should treat "Out.Result" properties as read-only.

RESULT

app.PassFail.Rate.Result

RESULT

app.PassFail.Tests.Result

PREFERENCES

app.Preferences

This set of variables controls user preferences for the instrument setup and operation.

AudibleFeedback	Bool
EnhancedPrecisionMode	Bool
HorOffsetControl	Enum
OffsetControl	Enum
Performance	Enum

AudibleFeedback *Bool*

Description

Sets/Queries whether audible feedback is enabled, to sound when a control is touched.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Turn on the audible feedback function.
app.Preferences.AudibleFeedback = True
```

Performance *Enum*

Description

Sets/Queries the variable value that control the Optimization of the instrument in terms of analysis or display.
When set to Analysis the display is given low priority and will update less frequently. Use this mode where analysis performance is much more important than display rate.

Values

Analysis	
AnalysisMid	
Default	
Display	
DisplayMid	

OffsetControl *Enum*

Description

Sets/Queries whether Vertical Offset constant in Volts or Divisions when the vertical scale control is changed.

Values

Div	
Volts	

HorOffsetControl *Enum*

Description

HorOffsetControl.

Values

Div	
Time	

EnhancedPrecisionMode *Bool*

Description

EnhancedPrecisionMode

EMAIL

app.Preferences.EMail

This set of variables controls user preferences for the instrument e-mail system.
E-Mail may be sent when the hardcopy button is pressed when the hardcopy system is appropriately configured. Two standards are supported, SMTP (Simple Mail Transport Protocol), and MAPI (Messaging Application Programming Interface).

DefaultRecipient	String

Mode	Enum
OriginatorAddress	String
SendTestMail	Action
SMTPServer	String

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Configure the originator and recipient addresses, replace these with
' appropriate values for your corporate network.
app.Preferences.Email.DefaultRecipient = "recipientAddress@do
```

Mode *Enum*

Description

Sets/Queries the transmission mode for e-mail.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

'  Set e-Mail mode to MAPI.
app.Preferences.EMail.Mode = "MAPI"
```

Values

MAPI	Messaging Application Programming Interface (Uses Outlook Express by default)
SMTP	Simple Mail Transfer Protocol, requires an SMTP server

DefaultRecipient *String*

Range Any number of characters

Description

Sets/Queries the default recipient of e-mail transmissions.

SMTPServer *String*

Range Any number of characters

Description

Sets/Queries the name of the SMTP Server for e-mail. Ask your system administrator if you are unsure of what value to set this to.

OriginatorAddress *String*

Range Any number of characters

Description

Sets/Queries the originator address for e-mail. This may be any address, and will be used when the recipient replies to a mail, note that the instrument doesn't necessarily have to have it's own E-Mail account in order to use this.

SendTestMail *Action*

Description

Sends a message by e-mail to test the system.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

'  Send an e-Mail message to test the system.
app.Preferences.EMail.SendTestMail
```

SAVERECALL

app.SaveRecall

Controls for the Save/Recall subsystem. Includes nodes for saving and recalling both Waveforms and Panels (Setups).

SETUP

app.SaveRecall.Setup

Controls for Saving and Recalling instrument setups.

DoRecallDefaultNvlPanel	Action
DoRecallDefaultPanel	Action
DoRecallPanel	Action
DoSavePanel	Action
InternalName1	String
InternalName2	String
InternalName3	String
InternalName4	String
InternalName5	String
InternalName6	String
PanelDir	FileName
PanelFilename	FileName
RecallInternal1	Action
RecallInternal2	Action
RecallInternal3	Action

RecallInternal4	Action
RecallInternal5	Action
RecallInternal6	Action
SaveInternal1	Action
SaveInternal2	Action
SaveInternal3	Action
SaveInternal4	Action
SaveInternal5	Action
SaveInternal6	Action

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Reset to default setup
app.SaveRecall.Setup.DoRecallDefaultPanel

' Store the current setup into the first of the 6 setup stores.
app.SaveRecall.Setup.InternalName1 = "My Setup1"
```

DoSavePanel *Action*

Description

Saves the current panel settings to the previously specified file. If the filename already exists, the file will be over-written without a prompt.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Create the filename for the next panel setup save.
app.SaveRecall.Setup.PanelFilename = "TestSave"

' Save the panel setup to the named file.
app.SaveRecall.Setup.DoSavePanel
```

SaveInternal1 *Action*

Description

Saves the current instrument settings into internal panel memory 1.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Save the current settings into internal panel memory 1.
app.SaveRecall.Setup.SaveInternal1
```

SaveInternal2 *Action*

Description

Please see SaveInternal1.

SaveInternal3 *Action*

Description

Please see SaveInternal1.

SaveInternal4 *Action*

Description

Please see SaveInternal1.

SaveInternal5 *Action*

Description

Please see SaveInternal1.

SaveInternal6 *Action*

Description

Please see SaveInternal1.

DoRecallDefaultPanel *Action*

Description

Recalls the factory set panel settings.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Recall the factory default panel settings.
app.SaveRecall.Setup.DoRecallDefaultPanel
```

DoRecallDefaultNvlPanel *Action*

Description

Recalls the factory set NVL (preference) panel settings. These are controls which are not affected when the default panel is recalled, and includes items such as the color preferences, remote control preferences, etc. Use with care!, especially when invoking via the VBS? Remote command via GPIB or TCP/IP, which could result in the controller being disconnected when the default port is selected.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Recall the factory default nvl panel settings.
app.SaveRecall.Setup.DoRecallDefaultNvlPanel
```

DoRecallPanel *Action*

Description

Recall the panel file named in the PanelFilename control.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Create the filename for the next panel setup to be recalled.
app.SaveRecall.Setup.PanelFilename = "Setup89"

' Recall the panel setup from the named file.
app.SaveRecall.Setup.DoRecallPanel
```

RecallInternal1 *Action*

Description

Recall the settings which are stored in internal panel memory 1.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Recall the settings from internal panel memory 1.
app.SaveRecall.Setup.RecallInternal1
```

RecallInternal2 *Action*

Description

Please see RecallInternal1.

RecallInternal3 *Action*

Description

Please see RecallInternal1.

RecallInternal4 *Action*

Description

Please see RecallInternal1.

RecallInternal5 *Action*

Description

Please see RecallInternal1.

RecallInternal6 *Action*

Description

Please see RecallInternal1.

PanelDir *FileName*

Range Any number of characters

Description

Directory in which setups are stored/recalled.

PanelFilename *FileName*

Range Any number of characters

Description

Sets/Queries the current filename for saving a panel setup. Note that a '.lss' extension is automatically appended if not supplied.

InternalName1 *String*

Range Any number of characters

Description

Sets/Queries the name of internal panel setup memory 1.

InternalName2 *String*

Range Any number of characters

Description

Please see InternalName1.

InternalName3 *String*

Range Any number of characters

Description

Please see InternalName1.

InternalName4 *String*

Range Any number of characters

Description

Please see InternalName1.

InternalName5 *String*

Range Any number of characters

Description

Please see InternalName1.

InternalName6 *String*

Range Any number of characters

Description

Please see InternalName1.

UTILITIES *app.SaveRecall.Utilities*

Controls used to manage files and folders, including the ability to create and delete folders, and the ability to delete files.

CreateDir	Action
DeleteAll	Action
DeleteFile	Action
Directory	FileName

Directory *FileName*

Range Any number of characters

Description

Defines the directory which will be used for the operations in this automation node.

DeleteAll *Action*

Description

Deletes all files in the directory specified by the Directory control without a cautionary prompt. Use with care! Files cannot be recovered if deleted accidentally.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Delete all files without showing a yes/no prompt.
app.SaveRecall.Utilities.Directory = "C:\MyDir"
app.SaveRecall.Utilities.DeleteAll
```

DeleteFile *Action*

Description

Deletes the file named by the Filename control

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Delete the named file
app.SaveRecall.Utilities.Filename = "C:\MyDir\MyFile.txt"
app.SaveRecall.Utilities.DeleteFile
```

CreateDir *Action*

Description

Creates the directory specified in the Directory control.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Create a named directory
app.SaveRecall.Utilities.Directory = "C:\MyDir"
app.SaveRecall.Utilities.CreateDir
```

WAVEFORM *app.SaveRecall.Waveform*

Contains controls used for saving and recalling waveforms.

Delimiter	Enum
DoRecall	Action
DoSave	Action
RecallDestination	Enum
RecallFrom	Enum
RecallSource	Enum
SaveDestination	Enum
SaveSource	Enum
SaveTo	Enum
TraceTitle	String
WaveFormat	Enum
WaveformDir	FileName

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Save C1 into M1
app.SaveRecall.Waveform.SaveTo = "Memory"
app.SaveRecall.Waveform.SaveSource = "C1"
app.SaveRecall.Waveform.SaveDestination = "M1"
app.SaveRecall.Waveform.DoSave
```

SaveTo *Enum*

Description

Sets/Queries type of destination for waveform save.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

'  Set the destination to Memory for waveform save.
app.SaveRecall.Waveform.SaveTo = "Memory"
```

Values

File	Save into file on a mass-storage device
Memory	Save into an internal memory (M1..M4)

WaveFormat *Enum*

Description

Sets/Queries the format to use when saving waveform data into a file. 'Binary' is the most efficient, storing one or two bytes per data sample, depending upon the number of significant bits. When in ASCII mode, the Subformat and Delimiter controls define the data format.

Values

ASCII	Plain ASCII files with choice of various delimiters
Binary	LeCroy's standard binary waveform format
Excel	
MathCad	
MATLAB	

SaveSource *Enum*

Description

Sets/Queries the source from which waveform data will be saved.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

'  Set the destination to memory for waveform save.
app.SaveRecall.Waveform.SaveTo = "Memory"
'  Set the source to C2, for saving a waveform.
app.SaveRecall.Waveform.SaveSource = "C2"
'  Set the destination to memory M4, for saving a waveform.
app.SaveRecall.Waveform.SaveDestination = "M4"
'  Save waveform data as previously specified.
app.SaveRecall.Waveform.DoSave
```

Values

AllDisplayed	
C1	
C2	
C3	
C4	
F1	
F2	
F3	
F4	
F5	
F6	
F7	
F8	
M1	
M2	
M3	
M4	
ScanHisto	
ScanOverlay	
TDRN	
Z1	
Z2	
Z3	
Z4	

SaveDestination *Enum*

Description

Sets/Queries the destination to which waveform data will be saved. Used only when the SaveTo = "Memory".

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

'  Setup to store trace C2 into M4 and perform the save operation
app.SaveRecall.Waveform.SaveTo = "Memory"
app.SaveRecall.Waveform.SaveSource = "C2"
app.SaveRecall.Waveform.SaveDestination = "M4"
app.SaveRecall.Waveform.DoSave
```

Values

M1	
M2	
M3	
M4	

Delimiter *Enum*

Description

Sets/Queries the delimiter to use when saving data in ASCII text mode.

Values

Comma	
Semicolon	
Space	
Tab	

WaveformDir *FileName*

Range Any number of characters

Description

Sets/Queries the directory for storing waveform files.

DoSave *Action*

Description

Save waveform data into an internal memory, or file on a mass-storage device, using the pre-specified source and destination.

TraceTitle *String*

Range Any number of characters

Description

Sets/Queries the title (prefix) to use when naming saved traces. This prefix will have the family (sequence) number appended to it when forming the filename.

RecallFrom *Enum*

Description

Sets/Queries the type of source for waveform recall.

Values

File	Recall from file on a mass-storage device
Memory	Recall from one of the internal memories (M1..M4)

RecallSource *Enum*

Description

Sets/Queries the source for recalling waveform data. Used only when recalling from an internal memory with RecallSource = "Memory".

Values

M1	
M2	
M3	
M4	

RecallDestination *Enum*

Description

Sets/Queries the destination for waveform recall. When the DoRecall action is executed the waveform will be transferred into this destination trace.

Values

M1	
M2	
M3	
M4	

DoRecall *Action*

Description

Recall waveform data into a trace memory. Source may be either an internal memory (M1..M4), or a file on a mass-storage device, depending on the state of the 'RecallFrom' control.

SYSTEMCONTROL *app.SystemControl*

ModalDialogTimeout	Integer

ModalDialogTimeout *Integer*

Range From 0 to 120 step 1

Description

Set a timeout, in units of seconds, used to auto-dismiss modal dialogs, with their default responses.

TDR *app.TDR*

Averages	Integer
Polarity1	Enum
Polarity2	Enum
Skew1	Double
Skew2	Double
Skew3	Double
Skew4	Double

Polarity1 *Enum*

Description

TDR Edge polarity for TDR1 trace

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Change TDR trace 1 polarity to negative
app.TDR.Polarity1 = "Neg"
```

Values

Neg	
Pos	

Skew1 *Double*

Range From -5e-011 to 5e-011 step 1e-013

Description

TDR module 1 hardware Skew control

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Change TDR Pod 1 skew to -15pS
app.TDR.Skew1 = -15e-12
```

Polarity2 *Enum*

Description

TDR Edge polarity for TDR2 trace

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Change TDR trace 2 polarity to negative
app.TDR.Polarity2 = "Neg"
```

Values

Neg	
Pos	

Skew2 *Double*

Range From -5e-011 to 5e-011 step 1e-013

Description

TDR module 2 hardware Skew control

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Change TDR Pod 2 skew to -15pS
app.TDR.Skew2 = -15e-12
```

Skew3 *Double*

Range From -5e-011 to 5e-011 step 1e-013

Description

TDR module 3 hardware Skew control

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Change TDR Pod 3 skew to -15pS
app.TDR.Skew3 = -15e-12
```

Skew4 *Double*

Range From -5e-011 to 5e-011 step 1e-013

Description

TDR module 4 hardware Skew control

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Change TDR Pod 4 skew to -15pS
app.TDR.Skew4 = -15e-12
```

Averages *Integer*

Range From 1 to 512 step 1

Description

Number of TDR signal averages in two port calibration mode, ie number of acquisition that are taken on one port 1 before switching to port 2 and back.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' change number of acquisitions in two port TDR mode to 10
app.TDR.Averages = 10
```

TDRN

app.TDR.TDRN

View	Bool

View *Bool*

Description

Turns the Normalized TDR trace view On or Off. There has to be a valid TDR calibration performed or loaded before this control becomes active

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Turn TDR Normalized trace off
app.TDR.TDRN.View = "off"
```

RESULT

app.TDR.TDRN.Out.Result

ZOOM

app.TDR.TDRN.Zoom

HorPos	Double
HorZoom	Double
ResetZoom	Action
VariableHorZoom	Bool
VariableVerZoom	Bool
VerPos	Double
VerZoom	Double

ResetZoom

Action

Description

Resets TDR Normalized trace view to show the whole trace

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Reset TDR normalized trace zoom settings to default
app.TDR.TDRN.Zoom.Reset
```

VariableVerZoom *Bool*

Description

Turns on/off TDR Normalized trace variable vertical zoom

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Turn off TDR normalized trace variable vertical zoom scale
app.TDR.TDRN.Zoom.VariableVerrZoom = "Off"
```

VariableHorZoom *Bool*

Description

Turns on/off TDR Normalized trace variable horizontal zoom

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Turn on TDR normalized trace variable horizontal zoom scale
app.TDR.TDRN.Zoom.VariableHorZoom = "On"
```

VerZoom *Double*

Range From 0.1 to 100 step (8 digits)

Description

TDR Normalized trace vertical zoom

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Assuming the normalized trace shows impedance parameter
' Change TDR normalized trace vertical scale per division to 10 ohms
app.TDR.TDRN.Zoom.VerZoom = 10
```

VerPos *Double*

Range From -1.5 to 1.5 step (8 digits)

Description

TDR Normalized trace vertical position

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Assuming the normalized trace shows S parameters
' Change TDR normalized trace center vertical position to start at -10dB
app.TDR.TDRN.Zoom.VerrPos = -10
```

HorZoom *Double*

Range From 0.1 to 1e+006 step (8 digits)

Description

TDR Normalized trace horizontal zoom (span per division)

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Assuming the normalized trace shows impedance parameter
' Change TDR normalized trace scale per division to 10nS
app.TDR.TDRN.Zoom.HorZoom = 10e-9
```

HorPos *Double*

Range From -0.5 to 0.5 step (8 digits)

Description

TDR Normalized trace horizontal position

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Assuming the normalized trace shows S parameters
' Change TDR normalized trace center position to start at 1GHz
app.TDR.TDRN.Zoom.HorPos = 1e9
```

DATETIMESETUP

app.Utility.DateTimeSetup

This set of variables controls user the date and time setup. In addition to manual controls for hh/mm/ss, dd/mm/yy, there is the ability to set the time and date from an Internet clock using the SNTP protocol.

CurrentDateAndTime	String
Day	Integer
Hour	Integer
Minute	Integer
Month	Integer
Second	Integer
SetFromSNTP	Action
Validate	Action
Year	Integer

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set time/date from the NIST Internet clock
app.Utility.DateTimeSetup.SetFromSNTP
```

Validate *Action*

Description

Validates any new settings. This action is equivalent to clicking 'Validate Changes' on the Date/Time page.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the day, hour, and minute, and validate.
app.Utility.DateTimeSetup.Day = 3
app.Utility.DateTimeSetup.Hour = 5
app.Utility.DateTimeSetup.Minute = 8

app.Utility.DateTimeSetup.Validate
```

SetFromSNTP *Action*

Description

Sets the real time clock from the simple network time protocol.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the real time clock from the simple network time protocol.
app.Utility.DateTimeSetup.SetFromSNTP
```

CurrentDateAndTime *String*

Range Any number of characters

Description

Reads the current date and time from the real-time calendar and clock.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Read the current date and time from the real-time calendar and clock.
app.Utility.DateTimeSetup.CurrentDateAndTime
```

Day *Integer*

Range From 1 to 31 step 1

Description

Sets/Queries the day of the month setting of the real-time clock as a number. The value will not be accepted by the clock until app.Utility.DateTimeSetup.Validate is sent. All time/date controls are validated at the same time.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the day of the month as 21.
app.Utility.DateTimeSetup.Day = 21
app.Utility.DateTimeSetup.Validate
```

Month *Integer*

Range From 1 to 12 step 1

Description

Sets/Queries the month setting of the real-time clock as a number. The value will not be accepted by the clock until app.Utility.DateTimeSetup.Validate is sent. All time/date controls are validated at the same time.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the month as August.
app.Utility.DateTimeSetup.Month = 8
app.Utility.DateTimeSetup.Validate
```

Year *Integer*

Range From 2000 to 2037 step 1

Description

Sets/Queries the year setting of the real-time clock as a number. The value will not be accepted by the clock until app.Utility.DateTimeSetup.Validate is sent. All time/date controls are validated at the same time.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the year as 2003.
app.Utility.DateTimeSetup.Year = 2003
app.Utility.DateTimeSetup.Validate
```

Hour *Integer*

Range From 0 to 23 step 1

Description

Sets/Queries the hours setting of the real-time clock as a number.
The value will not be accepted by the clock until app.Utility.DateTimeSetup.Validate is sent. All time/date controls are validated at the same time.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the hour as 13.
app.Utility.DateTimeSetup.Hour = 13
app.Utility.DateTimeSetup.Validate
```

Minute *Integer*

Range From 0 to 59 step 1

Description

Sets/Queries the minutes setting of the real-time clock as a number.
The value will not be accepted by the clock until app.Utility.DateTimeSetup.Validate is sent. All time/date controls are validated at the same time.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the minute as 34.
app.Utility.DateTimeSetup.Minute = 34
app.Utility.DateTimeSetup.Validate
```

Second *Integer*

Range From 0 to 59 step 1

Description

Sets/Queries the seconds setting of the real-time clock as a number.
The value will not be accepted by the clock until app.Utility.DateTimeSetup.Validate is sent. All time/date controls are validated at the same time.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the seconds as 55.
app.Utility.DateTimeSetup.Second = 55
app.Utility.DateTimeSetup.Validate
```

OPTIONS

app.Utility.Options

Options subsystem, contains controls to query the list of installed software and hardware options.

InstalledHWOptions	String

InstalledSWOptions	String
ScopeID	String

InstalledHWOptions *String*

Range Any number of characters

Description

Shows a list of the installed hardware options.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Read the list of installed hardware options and present
' in a popup dialog
MsgBox app.Utility.Options.InstalledHWOPtions
```

InstalledSWOptions *String*

Range Any number of characters

Description

Shows list of installed software options.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Read the list of installed software options and display
' in a popup dialog
MsgBox app.Utility.Options.InstalledSWOPtions
```

ScopeID *String*

Range Any number of characters

Description

Queries the ID of the instrument. This ID should be specified when purchasing software options for your instrument.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Read the ID of the instrument.
MsgBox app.Utility.Options.ScopeID
```

REMOTE *app.Utility.Remote*

Controls related to the remote control section of the instrument. Note that in this context Automation is not considered part of 'Remote'. Remote control currently includes control using ASCII remote commands from GPIB or TCP/IP.

Assistant	Enum

Interface	Enum
RestrictControl	Enum
SetToErrorsOnlyAndClearAtStartup	Bool

Interface *Enum*

Description

Sets/Queries the currently selected type of currently selected remote control interface.

Values

Off	
TCPIP	

Assistant *Enum*

Description

Sets/Queries the setting of the remote assistant.

Values

EO	Log errors only
FD	Log all remote commands/queries
OFF	Turn the assistant off

RestrictControl *Enum*

Description

Sets/Queries whether remote control is restricted to certain hosts, where the host name is defined either by IP address, or dns name.

Values

No	
Yes	

SetToErrorsOnlyAndClearAtStartup *Bool*

Description

Enable the resetting of the remote assistant to 'Errors Only' mode when the instrument is reset. Also ensure s that the remote assistant log is cleared upon startup.
This control is set by default to lower the risk that the remote assistant will be set to 'Full Dialog' mode and be forgotten, causing a decrease in remote control performance.

WAVESCAN *app.WaveScan*

This is the root of the WaveScan automation hierarchy.

WaveScan enables you to search for unusual events in a single capture, or to scan for an event in many acquisitions over a long period of time.

It may be considered a kind of software trigger.

Enable	Bool
FindRare1Sigma	Action
FindRare3Sigma	Action
FindRare5Sigma	Action
FindSelectRarest	Action
FindUseMean	Action

Enable *Bool*

Description

Sets/Queries the WaveScan enabled state.

FindUseMean *Action*

Description

Setup the filter to find measurements with values > the current statistical mean.

FindRare1Sigma *Action*

Description

Preset the filter limit and delta to find rare events. Uses the history of measurements since the last Clear Sweeps, or control change, to set the limit and delta to capture +/- 1 sigma events.

FindRare3Sigma *Action*

Description

Preset the filter limit and delta to find rare events. Uses the history of measurements since the last Clear Sweeps, or control change, to set the limit and delta to capture +/- 3 sigma events.

FindRare5Sigma *Action*

Description

Preset the filter limit and delta to find rare events. Uses the history of measurements since the last Clear Sweeps, or control change, to set the limit and delta to capture +/- 5 sigma events.

FindSelectRarest *Action*

Description

Setup the filter to find the rarest events (measurements) in each acquisition.
Using the RarestMode control, the smallest, largest, or both may be selected. Using the NumRareEvents control, the number of events to be found may be specified. Note that in 'Both' mode 1/2 of the selected number of events are taken from the Smallest, and Largest values.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' configure WaveScan to measure the 5 smallest, and 5 largest, rise time
measurements
app.WaveScan.Enable = True
app.WaveScan.Mode = "Measurement"
app.WaveScan.Measurement = "rise"
app.WaveScan.FindSelectRarest
app.WaveScan.RarestMode = "Both"
app.WaveScan.NumRareEvents = 10
```

SCANDECODE

app.WaveScan.ScanDecode

ClearSweeps	Action
Grid	String
View	Bool

View *Bool*

Description

Sets/Queries the trace's 'Viewed' state. When true, the trace is displayed on one of the display graticules. Note that even when a trace is not visible, it may be used as a source for Math, Measure, etc.

ClearSweeps *Action*

Description

Clear any accumulated result data. Useful for example to restart an average, or parameter statistics.

Grid *String*

Range Any number of characters

Description

Sets/Queries the graticule on which the trace is displayed. Typical values include:
YT1..YT8: one of the YT graticules used in Single, Dual, Quad, and Octal display modes.
NotOnGrid: not displayed on any graticule.

RESULT

app.WaveScan.ScanDecode.Out.Result

SCANHISTO

app.WaveScan.ScanHisto

ClearSweeps	Action
LabelsPosition	String
LabelsText	String
Persisted	Bool
PersistenceSaturation	Integer
PersistenceTime	Enum
PersistenceViewOn	Bool
ShowLastTrace	Bool
View	Bool
ViewLabels	Bool

View *Bool*

Description

Sets/Queries the trace's 'Viewed' state. When true, the trace is displayed on one of the display graticules. Note that even when a trace is not visible, it may be used as a source for Math, Measure, etc.

ClearSweeps *Action*

Description

Clear any accumulated result data. Useful for example to restart an average, or parameter statistics.

Persisted *Bool*

Description

Sets/Queries the persisted state of the waveform. If the Display.LockPersistence control is set to 'AllLocked' then the persisted state of all displayed waveforms will be the same. If the Display.LockPersistence control is set to 'PerTrace' then the persisted state of each waveform may be independently controlled.

PersistenceSaturation *Integer*

Range From 0 to 100 step 1

Description

Sets/Queries the saturation threshold for persisted waveforms.
All information at this level or above will be recorded with the same color or intensity.
See the general description above for a discussion of the locked and unlocked persistence modes.

ShowLastTrace *Bool*

Description

Sets/Queries the state of the Show Last Trace control. If True then when this trace is displayed in persistence mode the last acquired waveform will be superimposed on the accumulating persistence map.
See the general description above for a discussion of the locked and unlocked persistence modes.

PersistenceTime *Enum*

Description

Sets/Queries the state of the Persistence Time control. Controls the persistence decay time for this trace. See the general description above for a discussion of the locked and unlocked persistence modes.

Values

0.5s	
10s	
1s	
20s	
2s	
5s	
Infinite	

PersistenceViewOn *Bool*

Description

Query the state of the persistence view.

LabelsText *String*

Range Any number of characters

LabelsPosition *String*

Range Any number of characters

Description

Sets / Queries the horizontal position of the label attached to the acquisition trace Cx. The unit of measurement is the unit of the horizontal scale. The measurement is made from the trigger point. Note that this control is a string, not a numeric value. This allows multiple labels to be positioned, as shown in the example below.

ViewLabels *Bool*

Description

Sets/Queries whether the user-defined labels for the trace are visible.
See Also: LabelsPosition and LabelsText controls.

HISTOGRAM *app.WaveScan.ScanHisto.Histogram*

AutoFindScale	Bool
Bins	DoubleLockstep
BufferSize	Integer
Center	Double
ClearSweeps	Action
FindScale	Action
HorScale	DoubleLockstep
Values	Integer
VerScaleType	Enum

ClearSweeps *Action*

Description

Clear any accumulated result data. Useful for example to restart an average, or parameter statistics.

Values *Integer*

Range From 20 to 1000 step 1

Description

Size of the buffer in which all values currently histogrammed are queued.

BufferSize *Integer*

Range From 200 to 1000 step 1

Description

Size of the buffer which stores incoming parameter values, ready to be histogrammed.
Not to be confused with the 'Values' control, which defines the number of values from the buffer which are currently rendered in the histogram.

Bins *DoubleLockstep*

Range From 20 to 2000 step 1, locked to 1 2 5, fine grain allowed=false, on=false

Description

Number of bins in the histogram.

HorScale *DoubleLockstep*

Range From 1e-012 to 1e+012 step 0.01, locked to 1 2 5, fine grain allowed=false, on=false

Description

Horizontal scale of the histogram, per division of the graticule.

Center *Double*

Range From -1e+010 to 1e+010 step 1e-012

Description

Defines the value of the bin which is centered horizontally within the graticule.

FindScale *Action*

Description

Automatically determine an appropriate horizontal scale for the histogram, using the values currently in the histogram buffer.

VerScaleType *Enum*

Description

Vertical Scale mode of the histogram, Linear, or 'Linear with Constant Maximum'.

Values

LinConstMax	
Linear	

AutoFindScale *Bool*

Description

Defines whether the histogram horizontal axis is automatically scaled when sufficient data has been accumulated.
The FindScale control may be used to manually find the scale, if this control is set to False.

RESULT

app.WaveScan.ScanHisto.Out.Result

ZOOM

app.WaveScan.ScanHisto.Zoom

HorPos	Double
HorZoom	Double
ResetZoom	Action
VariableHorZoom	Bool
VariableVerZoom	Bool
VerPos	Double
VerZoom	Double

ResetZoom *Action*

Description

Resets the zoom settings to their default values.

VariableVerZoom *Bool*

Description

Enable/Disable the variable Vertical Zoom control. If enabled, the VerZoom control may be set to a value other than the standard 1, 2, 5 sequence.

VariableHorZoom *Bool*

Description

Enable/Disable the variable Horizontal Zoom control. If enabled, the HorZoom control may be set to a value other than the standard 1, 2, 5 sequence.

VerZoom *Double*

Range From 0.1 to 100 step (8 digits)

Description

Vertical Zoom setting. Locked to a 1, 2, 5 sequence unless VariableVerZoom is set to True .

VerPos *Double*

Range From -1.5 to 1.5 step (8 digits)

Description

Vertical Position of the trace, normalized to a value between -1.5 and 1.5. A value of zero is the default, and indicates no position change relative to the source trace.

HorZoom *Double*

Range From 0.1 to 1e+006 step (8 digits)

Description

Horizontal Zoom setting. Locked to a 1, 2, 5 sequence unless VariableHorZoom is set to True .

HorPos *Double*

Range From -0.5 to 0.5 step (8 digits)

Description

Horizontal Position of the trace, normalized to a value between -0.5 and 0.5. A value of zero is the default, and indicates no position change relative to the source trace.

SCANOVERLAY

app.WaveScan.ScanOverlay

ClearSweeps	Action
EnablePersistence	Bool
Persisted	Bool

PersistenceSaturation	Integer
PersistenceTime	Enum
ShowLastTrace	Bool
View	Bool

View *Bool*

Description

Sets/Queries the trace's 'Viewed' state. When true, the trace is displayed on one of the display graticules. Note that even when a trace is not visible, it may be used as a source for Math, Measure, etc.

ClearSweeps *Action*

Description

Clear any accumulated result data. Useful for example to restart an average, or parameter statistics.

Persisted *Bool*

Description

Sets/Queries the persisted state of the waveform. If the Display.LockPersistence control is set to 'AllLocked' then the persisted state of all displayed waveforms will be the same. If the Display.LockPersistence control is set to 'PerTrace' then the persisted state of each waveform may be independently controlled.

PersistenceSaturation *Integer*

Range From 0 to 100 step 1

Description

Sets/Queries the saturation threshold for persisted waveforms.
All information at this level or above will be recorded with the same color or intensity.
See the general description above for a discussion of the locked and unlocked persistence modes.

ShowLastTrace *Bool*

Description

Sets/Queries the state of the Show Last Trace control. If True then when this trace is displayed in persistence mode the last acquired waveform will be superimposed on the accumulating persistence map.
See the general description above for a discussion of the locked and unlocked persistence modes.

PersistenceTime *Enum*

Description

Sets/Queries the state of the Persistence Time control. Controls the persistence decay time for this trace. See the general description above for a discussion of the locked and unlocked persistence modes.

Values

0.5s	
10s	
1s	
20s	
2s	
5s	
Infinite	

EnablePersistence *Bool*

Description

Set to place the WaveScan 'ScanOverlay' in persistence mode, as opposed to 'overlay' mode (where all contributing sub-waveforms are overlaid)

RESULT

app.WaveScan.ScanOverlay.Out.Result

ZOOM

app.Zoom

GoToEnd	Action
GoToStart	Action
HorZoomIn	Action
HorZoomOut	Action

MultiZoomOn	Bool
QuickZoom	Action
ResetAll	Action
ResetZoom	Action
VariableHorZoom	Bool

ResetAll *Action*

Description

Reset all Zx to their default settings.

QuickZoom *Action*

Description

Zoom all Cx that are on at an horizontal factor of 10.

MultiZoomOn *Bool*

Description

Turn MultiZoom On and includes all the Zx automatically if any viewed.

ResetZoom *Action*

Description

Resets the zoom settings to their default values.

VariableHorZoom *Bool*

Description

Enable/Disable the variable Horizontal Zoom control. If enabled, the HorZoom control may be set to a value other than the standard 1, 2, 5 sequence.

GoToStart *Action*

Description

When in multi-zoom mode, scroll to the start of the source waveform, who's first point will be centered on the graticule.

GoToEnd *Action*

Description

When in multi-zoom mode, scroll to the end of the source waveform, who's last point will be centered on the graticule.

HorZoomIn *Action*

Description

Horizontally zoom in all the traces included in MultiZoom.

HorZoomOut *Action*

Description

Horizontally zoom out all the traces included in MultiZoom.

ZX *app.Zoom.Zx*

ClearSweeps	Action
DoStoreToMemoryTrace	Action
Equation	String
LabelsPosition	String
LabelsText	String
Persisted	Bool
PersistenceSaturation	Integer
PersistenceTime	Enum
PersistenceViewOn	Bool
ShowLastTrace	Bool
Source	Enum
UseGrid	String
View	Bool
ViewLabels	Bool

View *Bool*

Description

Sets/Queries the trace's 'Viewed' state. When true, the trace is displayed on one of the display graticules. Note that even when a trace is not visible, it may be used as a source for Math, Measure, etc.

ClearSweeps *Action*

Description

Clear any accumulated result data. Useful for example to restart an average, or parameter statistics.

UseGrid *String*

Range Any number of characters

Description

Sets/Queries the grid in use for the zoom trace Zx.
See also app.Acquisition.Cx.UseGrid.

Persisted *Bool*

Description

Sets/Queries the persisted state of the waveform. If the Display.LockPersistence control is set to 'AllLocked' then the persisted state of all displayed waveforms will be the same. If the Display.LockPersistence control is set to 'PerTrace' then the persisted state of each waveform may be independently controlled.

PersistenceSaturation *Integer*

Range From 0 to 100 step 1

Description

Sets/Queries the saturation threshold for persisted waveforms.
All information at this level or above will be recorded with the same color or intensity.
See the general description above for a discussion of the locked and unlocked persistence modes.

ShowLastTrace *Bool*

Description

Sets/Queries the state of the Show Last Trace control. If True then when this trace is displayed in persistence mode the last acquired waveform will be superimposed on the accumulating persistence map.
See the general description above for a discussion of the locked and unlocked persistence modes.

PersistenceTime *Enum*

Description

Sets/Queries the state of the Persistence Time control. Controls the persistence decay time for this trace. See the general description above for a discussion of the locked and unlocked persistence modes.

Values

0.5s	
10s	
1s	
20s	
2s	
5s	
Infinite	

PersistenceViewOn *Bool*

Description

Query the state of the persistence view.

LabelsText *String*

Range Any number of characters

LabelsPosition *String*

Range Any number of characters

Description

Sets / Queries the horizontal position of the label attached to the acquisition trace Cx. The unit of measurement is the unit of the horizontal scale. The measurement is made from the trigger point. Note that this control is a string, not a numeric value. This allows multiple labels to be positioned, as shown in the example below.

ViewLabels *Bool*

Description

Sets/Queries whether the user-defined labels for the trace are visible.
See Also: LabelsPosition and LabelsText controls.

Source *Enum*

Description

Zoom source trace.

Values

C1	
C2	
C3	
C4	
F1	
F2	
F3	
F4	
F5	
F6	
F7	
F8	
M1	
M2	
M3	
M4	
ScanHisto	
ScanOverlay	
TDRN	
Z2	
Z3	
Z4	

DoStoreToMemoryTrace *Action*

Description

Store the content of Zx into the corresponding Memory Slot (Mx).

Equation *String*

Range Any number of characters

Description

Same as app.Math.Fx.Equation.

RESULT

app.Zoom.Zx.Out.Result

ZOOM

app.Zoom.Zx.Zoom

HorPos	Double
HorZoom	Double
ResetZoom	Action
VariableHorZoom	Bool
VariableVerZoom	Bool
VerPos	Double
VerZoom	Double

ResetZoom *Action*

Description

Resets the zoom settings to their default values.

VariableVerZoom ***Bool***

Description

Enable/Disable the variable Vertical Zoom control. If enabled, the VerZoom control may be set to a value other than the standard 1, 2, 5 sequence.

VariableHorZoom ***Bool***

Description

Enable/Disable the variable Horizontal Zoom control. If enabled, the HorZoom control may be set to a value other than the standard 1, 2, 5 sequence.

VerZoom ***Double***

Range From 0.1 to 100 step (8 digits)

Description

Vertical Zoom setting. Locked to a 1, 2, 5 sequence unless VariableVerZoom is set to True .

VerPos ***Double***

Range From -1.5 to 1.5 step (8 digits)

Description

Vertical Position of the trace, normalized to a value between -1.5 and 1.5. A value of zero is the default, and indicates no position change relative to the source trace.

HorZoom ***Double***

Range From 0.1 to 1e+006 step (8 digits)

Description

Horizontal Zoom setting. Locked to a 1, 2, 5 sequence unless VariableHorZoom is set to True .

HorPos ***Double***

Range From -0.5 to 0.5 step (8 digits)

Description

Horizontal Position of the trace, normalized to a value between -0.5 and 0.5. A value of zero is the default, and indicates no position change relative to the source trace.

AVERAGE

app.Math.Fx.OperatorYSetup (Operator = "Average")

Waveform Averaging.

AverageType	Enum
ClearSweeps	Action
Sweeps	Integer

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Turn trace F1 on and setup to average the data from C1
' Average mode is set to Continuous
app.Math.F1.View = True
app.Math.F1.Operator1 = "Average"
app.Math.F1.MathMode = "OneOpe
```

AverageType *Enum*

Description

Sets / Queries the averaging mode. Continuous and Summation modes are supported.

Values

Continuous	
Summed	

Sweeps *Integer*

Range From 1 to 1000000 step 1

Description

Sets / Queries the number of sweeps to be averaged when trace Fx is set to averaging - continuous or summed.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set number of sweeps to be averaged in trace F1 as 20.
app.Math.F1.Operator1Setup.Sweeps = 20
```

ClearSweeps *Action*

Description

Clears all averaged sweeps.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Clear sweeps for average in trace F1.
app.Math.F1.Operator1Setup.ClearSweeps
```

DERIVATIVE

app.Math.Fx.OperatorYSetup (Operator = "Derivative")

Computes the derivative of the waveform (next_sample_value - this_sample_value) / horizontal_sample_interval.

EnableAutoScale	Bool
FindScale	Action
VerOffset	Double
VerScale	DoubleLockstep

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Start a find scale operation for derivative function trace F1
app.Math.F1.View = True
app.Math.F1.MathMode = "OneOperator"
app.Math.F1.Operator1 = "Derivative"
app.Math.F1.Operato
```

VerScale *DoubleLockstep*

Range From 1e-012 to 1e+013 step 10000, locked to 1 2 5, fine grain allowed=false, on=false

Description

Sets/Queries the vertical scale of the derivative function Fx.

VerOffset *Double*

Range From -1e+006 to 1e+006 step 1e-009

Description

Sets/Queries the vertical offset of the derivative function trace Fx.

FindScale *Action*

Description

Initiates a Find Scale action, to set a suitable vertical scale for the derivative function trace Fx.

EnableAutoScale *Bool*

Description

Sets/Queries whether the autoscale function is enabled for the derivative function trace Fx. If enabled, an auto-scale operation is performed whenever the setup changes.

DESKEW

app.Math.Fx.OperatorYSetup (Operator = "Deskew")

Deskew waveform by shifting it in time.

WaveDeskew	Double

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set the displacement of the trace F3 to 3.7e-9
app.Math.F3.View = True
app.Math.F3.MathMode = "OneOperator"
app.Math.F3.Operator1 = "Deskew"
app.Math.F3.Operator1Setup.WaveDeskew
```

WaveDeskew *Double*

Range From -2.5e-008 to 2.5e-008 step 1e-012

Description

Sets/Queries the displacement in time of the trace Fx. A positive value delays the signal: a negative one makes it appear earlier.

ENHANCEDRESOLUTION

app.Math.Fx.OperatorYSetup (Operator = "EnhancedResolution")

Bits	Enum

Bits *Enum*

Description

Number of bits of enhanced resolution. ERES is a FIR filter with a gaussian frequency response.

Values

0.5	Enhance by 0.5 bits
1	Enhance by 1 bits
1.5	Enhance by 1.5 bits
2	Enhance by 2 bits
2.5	Enhance by 2.5 bits
3	Enhance by 3 bits

ENVELOPE

app.Math.Fx.OperatorYSetup (Operator = "Envelope")

Envelope of minimum and maximum values for an ensemble of sweeps, or 'Extrema'

ClearSweeps	Action
Sweeps	Integer

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Configure F3 to be an envelope of C1
app.Math.F3.View = True
app.Math.F3.Source1 = "C1"
app.Math.F3.MathMode = "OneOperator"
app.Math.F3.Operator1 = "Envelope"
app.Math.F3.Operat
```

Sweeps ***Integer***

Range From 1 to 1000000 step 1

Description

Sets/Queries the maximum number of sweeps to be used by the envelope function trace Fx.

ClearSweeps ***Action***

Description

Initiates a Clear Sweeps operation for envelope function trace Fx.

FFT

app.Math.Fx.OperatorYSetup (Operator = "FFT")

Fast Fourier Transform of waveform data.

Algorithm	Enum
FillType	Enum
SuppressDC	Bool
Type	Enum
Window	Enum

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Configure F3 to perform an FFT of C1
app.Math.F3.View = True
app.Math.F3.Source1 = "C1"
app.Math.F3.MathMode = "OneOperator"
app.Math.F3.Operator1 = "FFT"
app.Math.F3.Operator1Se
```

Type *Enum*

Description

Sets/Queries the type of FFT spectrum for function trace Fx.

Values

Magnitude	Magnitude with linear vertical scale
Phase	Phase
PowerSpectrum	Power Spectrum

Window *Enum*

Description

Sets/Queries the type of window for the FFT function trace Fx.

Values

BlackmanHarris	
FlatTop	
Hamming	
Rectangular	
VonHann	

Algorithm *Enum*

Description

Sets/Queries the algorithm for the FFT in function trace Fx.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set function trace F4 to FFT.
app.Math.F4.Operator1 = "FFT"
' Set the FFT algorithm to power of two.
app.Math.F4.Operator1Setup.Algorithm = "Power2"
```

Values

LeastPrime	
Power2	

FillType *Enum*

Description

Sets/Queries the type of trace fill to use in the FFT function trace Fx.

Values

Truncate	
ZeroFill	

SuppressDC *Bool*

Description

Enables/Disables suppression of the value at zero frequency in the FFT spectrum.

FLOOR

app.Math.Fx.OperatorYSetup (Operator = "Floor")

Most negative or minimum values for an ensemble of sweeps, or "Floor"

ClearSweeps	Action
Sweeps	Integer

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Configure F1 to measure the Floor of the first 1000
' sweeps of C1
app.Math.F1.View = True
app.Math.F1.Source1 = "C1"
app.Math.F1.MathMode = "OneOperator"
app.Math.F1.Operator1
```

ClearSweeps *Action*

Description

Initiates a clear sweeps action for the Floor function trace Fx.

Sweeps *Integer*

Range From 1 to 1000000 step 1

Description

Sets/Queries the maximum number of sweeps for the Floor function trace Fx.

HISTOGRAM

app.Math.Fx.OperatorYSetup (Operator = "Histogram")

Histogram of the values of a parameter, or if a waveform is used as the input, histogram the waveform sample amplitudes.

AutoFindScale	Bool
Bins	DoubleLockstep
Center	Double
ClearSweeps	Action
FindScale	Action
HorScale	DoubleLockstep
Values	Integer
VerScaleType	Enum

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Configure F1 to histogram the first 200000 sample
' values from source waveform C1 into 50 bins.
' Auto find-scale is enabled.
app.Math.F1.View = True
app.Math.F1.Source1 = "C1"
```

ClearSweeps *Action*

Description

Clears the contents of all the bins of the histogram function Fx.

Values *Integer*

Range From 20 to 1000 step 1

Description

Sets/Queries the maximum number of values from the source result to include in the histogram function Fx.

Bins *DoubleLockstep*

Range From 20 to 2000 step 1, locked to 1 2 5, fine grain allowed=false, on=false

Description

Sets/Queries the number of bins in the histogram function Fx.

HorScale *DoubleLockstep*

Range From 1e-012 to 1e+012 step 0.01, locked to 1 2 5, fine grain allowed=false, on=false

Description

Sets/Queries the horizontal scale in units per division for the histogram function Fx. Use the FindScale control to automatically determine the scale by looking at the non-zero populated bins.

Center *Double*

Range From -1e+010 to 1e+010 step 1e-012

Description

Sets/Queries the horizontal value at the center of the graticule of the histogram function Fx.

FindScale *Action*

Description

Creates a suitable horizontal position and scale to include all the non-empty bins of the histogram Fx.

VerScaleType *Enum*

Description

Sets/Queries the way that the vertical scale is calculated as the histogram Fx grows.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set function F1 as histogram.
app.Math.F1.Operator1 = "Histogram"
' Set the vertical scale type to linear with constant maximum.
app.Math.F1.Operator1Setup.VerScaleType = "LinConstMax"
```

Values

LinConstMax	Linear scale with constant maximum value
Linear	Linear scale

AutoFindScale *Bool*

Description

Enables/Disables automatic scale setting for the histogram function Fx.

INTEGRAL

app.Math.Fx.OperatorYSetup (Operator = "Integral")

Integral of the linearly rescaled (multiplier and adder) input.

Adder	Double
AutoFindScale	Bool
FindScale	Action
Multiplier	Double
VerOffset	Double
VerScale	DoubleLockstep

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Configure F1 to integrate C1
app.Math.F1.View = True
app.Math.F1.Source1 = "C1"
app.Math.F1.MathMode = "OneOperator"
app.Math.F1.Operator1 = "Integral"
app.Math.F1.Operator1Setup
```

Multiplier *Double*

Range From -1e+006 to 1e+006 step 1e-006

Description

Sets/Queries the multiplying constant M for the integral function Fx, where Fx = M . Input + A

Adder *Double*

Range From -1e-009 to 1e-009 step 1e-012

Description

Sets/Queries the additive A for the integral function Fx, where Fx = M . Input + A.

VerScale *DoubleLockstep*

Range From 1e-012 to 1e+007 step 0.01, locked to 1 2 5, fine grain allowed=false, on=false

Description

Sets/Queries the vertical scale for the integral function trace Fx.

VerOffset *Double*

Range From -1e+006 to 1e+006 step 1e-015

Description

Sets/Queries the vertical offset for the integral function trace Fx.

FindScale *Action*

Description

Initiates an action to find suitable vertical offset and scale for the integral function trace Fx.

AutoFindScale *Bool*

Description

Set/Query the state of the 'AutoFindScale' cvar, which enables the automatic scaling of the Integral when the acquisition setup changes.

MATLABWAVEFORM

app.Math.Fx.OperatorYSetup (Operator = "MATLABWaveform")

Process a waveform using an external MATLAB application.

MATLABCode	String

MATLABPlot	Bool
MATLABScalePerDiv	Double
MATLABZeroOffset	Double

MATLABCode ***String***

Range Any number of characters

Description

String containing the MATLAB code to execute when new data is presented.

MATLABPlot ***Bool***

Description

If true, the result of the MATLAB processing operation is plotted by MATLAB, in a floating window.

MATLABZeroOffset ***Double***

Range From -1e+009 to 1e+009 step 1e-009

Description

Zero Offset (vertically), used to scale the waveform returned from MATLAB to the DSO's graticule.

MATLABScalePerDiv ***Double***

Range From 1e-009 to 1e+009 step 1e-009

Description

Vertical Scaling, used to scale the waveform returned from MATLAB to the DSO's graticule.

PERSISTENCEHISTOGRAM

app.Math.Fx.OperatorYSetup (Operator = "PersistenceHistogram")

CenterCursor	Action
ClearSweeps	Action
CutDirection	Enum
VerCutCenter	Double
VerCutWidth	Double

VerCutCenter ***Double***

Range From -1.79769e+308 to 1.79769e+308 step 0

Description

Horizontal coordinate of center of cut or slice from the persistence map (in horizontal units)

VerCutWidth ***Double***

Range From -1.79769e+308 to 1.79769e+308 step 0

Description

Horizontal coordinate of center of cut or slice from the persistence map (in horizontal units)

CutDirection ***Enum***

Description

Specifies either a " vertical" cut direction or "horizontal" cut direction producing a histogram with the same horizontal coordinates as the chosen direction.

Values

Horizontal	
Vertical	

ClearSweeps ***Action***

Description

Clear any accumulated result data. Useful for example to restart an average, or parameter statistics.

CenterCursor ***Action***

Description

Center the slice about the center of the axis, Vertical or Horizontal, depending upon the CutDirection Setting.

RESCALE

app.Math.Fx.OperatorYSetup (Operator = "Rescale")

Linearly transform the vertical values of a waveform.

Adder	Double

CustomUnit	Bool
Multiplier	Double

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Configure F1
app.Math.F1.View = True
app.Math.F1.Source1 = "C1"
app.Math.F1.MathMode = "OneOperator"
app.Math.F1.Operator1 = "Rescale"
app.Math.F1.Operator1Setup.Adder = 2.0
app
```

Multiplier ***Double***

Range From -1e+018 to 1e+018 step (9 digits)

Description

Sets/Queries the multiplicative constant M in the rescale function Fx = M.Input + A

Adder ***Double***

Range From -1e+018 to 1e+018 step (9 digits)

Description

Sets/Queries the additive constant A in the rescale function Fx = M.Input + A

CustomUnit ***Bool***

Description

Enables/Disables the application of a custom unit of measurement to the rescale function trace Fx.

ROOF

app.Math.Fx.OperatorYSetup (Operator = "Roof")

Most positive or maximum values for an ensemble of sweeps, or "Roof"

ClearSweeps	Action
Sweeps	Integer

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Configure F1 to measure the Roof of the first 1000
' sweeps of C1
```

```
app.Math.F1.View = True
app.Math.F1.Source1 = "C1"
app.Math.F1.MathMode = "OneOperator"
app.Math.F1.Operator1 =
```

ClearSweeps *Action*

Description

Initiates a clear sweeps action for the roof function trace Fx.

Sweeps *Integer*

Range From 1 to 1000000 step 1

Description

Sets/Queries the maximum number of sweeps for the Roof function trace Fx.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set function trace F2 to roof.
app.Math.F2.Operator1 = "Roof"
' Set the maximum number of sweeps to 150.
app.Math.F2.Operator1Setup.Sweeps = 150
```

SINXOVERX

app.Math.Fx.OperatorYSetup (Operator = "SinXOverX")

TREND

app.Math.Fx.OperatorYSetup (Operator = "Trend")

Trend of the values of a parameter, if connected to a parameter result source, or a trend of the sample values of a waveform, if connected to a waveform result source.

AutoFindScale	Bool
Center	Double
ClearSweeps	Action
FindScale	Action
Mode	Enum
Values	Integer
VerScale	DoubleLockstep

ClearSweeps *Action*

Description

Clears the contents of the trend trace Fx.

Values *Integer*

Range From 2 to 1000 step 1

Description

Sets/Queries the number of visible values in the trend trace Fx.

VerScale *DoubleLockstep*

Range From 1e-015 to 1e+012 step 0.01, locked to 1 2 5, fine grain allowed=false, on=false

Description

Sets/Queries the vertical scale of the trend trace Fx.

Center *Double*

Range From -1.79769e+308 to 1.79769e+308 step 0

Description

Sets/Queries the vertical position of the centre of the trend trace Fx.

AutoFindScale *Bool*

Description

Enables/Disables the automatic setting of the vertical scale and vertical offset for the trend trace Fx.

FindScale *Action*

Description

Sets the vertical scale and offset to optimum values to display the trend trace Fx.

Mode *Enum*

Description

Trend mode, defines which parameter measurements are used to build the trend.

Values

All	Trend all values
AllperTrace	Trend an average of all values per acquisition
Average	Trend all values per trace, clear before new acquisition.

PARAMDIFFERENCE

app.Measure.Px.Operator (ArithEngine = "ParamDifference")

PARAMINVERT

app.Measure.Px.Operator (ArithEngine = "ParamInvert")

PARAMPRODUCT

app.Measure.Px.Operator (ArithEngine = "ParamProduct")

PARAMRATIO

app.Measure.Px.Operator (ArithEngine = "ParamRatio")

PARAMRESCALE

app.Measure.Px.Operator (ArithEngine = "ParamRescale")

PARAMSUM

app.Measure.Px.Operator (ArithEngine = "ParamSum")

AMPLITUDE

app.Measure.Px.Operator (ParamEngine = "Amplitude")

AREA

app.Measure.Px.Operator (ParamEngine = "Area")

Calculates the area of the input waveform relative to zero.

Cyclic	Bool

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set parameter P1 to area.
app.Measure.P1.View = True
app.Measure.P1.MeasurementType = "measure"
app.Measure.P1.ParamEngine = "Area"
app.Measure.P1.Source1 = "C1"

' Enable cycli
```

Cyclic ***Bool***

Description

Enables/Disables cyclic calculation of area parameter Px, that is calculated using a whole number of cycles of the signal.

Note: the HelpMarkers aid in observing over which region of the waveform the measurement is made.

BASE

app.Measure.Px.Operator (ParamEngine = "Base")

DELTATIMEATLEVEL

app.Measure.Px.Operator (ParamEngine = "DeltaTimeAtLevel")

FindLevel1	Action
FindLevel2	Action
Hysteresis1	Double
Hysteresis2	Double

LevelType1	Enum
LevelType2	Enum
PercentLevel1	Double
PercentLevel2	Double
Slope1	Enum
Slope2	Enum

LevelType1 ***Enum***

Description

Type of level on first trace: absolute/percent and %Pkpk, %0-Min, %0-Max with EMC option

Values

Absolute	
Percent	

LevelType2 ***Enum***

Description

Type of level on second trace: absolute/percent and %Pkpk, %0-Min, %0-Max with EMC option.

Values

Absolute	
Percent	

Hysteresis1 ***Double***

Range From 0 to 10 step 0.1

Description

Hysteresis in divisions around the level on first trace. The signal must enter the hysteresis zone (shown as a cursor) on one side and exit from the other side to qualify a transition.

Hysteresis2 ***Double***

Range From 0 to 10 step 0.1

Description

Hysteresis in divisions around the level on second trace. The signal must enter the hysteresis zone (shown as a cursor) on one side and exit from the other side to qualify a transition.

FindLevel1 *Action*

Description

When in absolute level mode, finds the absolute level at 50% on the first trace

FindLevel2 *Action*

Description

When in absolute level mode, finds the absolute level at 50% on the second trace

PercentLevel1 *Double*

Range From 0 to 100 step 1

Description

Level on first trace in percent.

PercentLevel2 *Double*

Range From 0 to 100 step 1

Description

Level on second trace in percent.

Slope1 *Enum*

Description

Sign of detected transition on first trace: positive, negative, both.

Values

Both	
Neg	
Pos	

Slope2 *Enum*

Description

Sign of detected transition on second trace: positive, negative, both

Values

Both	
Neg	
Pos	

EXTINCTIONRATIO

app.Measure.Px.Operator (ParamEngine = "ExtinctionRatio")

Aperture	Double
CalcType	Enum
CursorDisplay	Enum

Aperture *Double*

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

CursorDisplay *Enum*

Description

Set/Query the CursorDisplay cvar. This defines whether the source trace is annotated with 'Help Markers' generated by the measurement.

Values

Detailed	
Off	
Simple	

CalcType *Enum*

Description

Extinction ratio units.

Values

db	
linear	
pct	

EYEAMPLITUDE

app.Measure.Px.Operator (ParamEngine = "EyeAmplitude")

Aperture	Double
CursorDisplay	Enum

Aperture *Double*

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

CursorDisplay *Enum*

Description

Set/Query the CursorDisplay cvar. This defines whether the source trace is annotated with 'Help Markers' generated by the measurement.

Values

Detailed	
Off	
Simple	

EYEBER

app.Measure.Px.Operator (ParamEngine = "EyeBER")

Aperture	Double
CursorDisplay	Enum

Aperture *Double*

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

CursorDisplay *Enum*

Description

Set/Query the CursorDisplay cvar. This defines whether the source trace is annotated with 'Help Markers' generated by the measurement.

Values

Detailed	
Off	
Simple	

EYECROSSING

app.Measure.Px.Operator (ParamEngine = "EyeCrossing")

Output	Enum

Output *Enum*

Description

Type of output returned, percentage of eye height, or absolute voltage.

Values

Absolute	
Percent	

EYEFALLTIME

app.Measure.Px.Operator (ParamEngine = "EyeFallTime")

Aperture	Double
HighPct	Double
LowPct	Double

LowPct ***Double***

Range From 10 to 40 step 1

Description

High level in percent.

HighPct ***Double***

Range From 60 to 90 step 1

Description

High level in percent.

Aperture ***Double***

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

EYEHEIGHT

app.Measure.Px.Operator (ParamEngine = "EyeHeight")

Aperture	Double
CalcUnits	Enum
CursorDisplay	Enum

Aperture ***Double***

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

CalcUnits *Enum*

Description

Specifies the units of the parameter readout, linear (volts), or decibels.

Values

dB	
linear	

CursorDisplay *Enum*

Description

Set/Query the CursorDisplay cvar. This defines whether the source trace is annotated with 'Help Markers' generated by the measurement.

Values

Detailed	
Off	
Simple	

EYEMEAN

app.Measure.Px.Operator (ParamEngine = "EyeMean")

Aperture	Double

Aperture *Double*

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.

It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

EYEONELEVEL

app.Measure.Px.Operator (ParamEngine = "EyeOneLevel")

Aperture	Double

Aperture *Double*

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

EYEOPENINGFACTOR

app.Measure.Px.Operator (ParamEngine = "EyeOpeningFactor")

Aperture	Double
CursorDisplay	Enum

Aperture ***Double***

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

CursorDisplay ***Enum***

Description

Set/Query the CursorDisplay cvar. This defines whether the source trace is annotated with 'Help Markers' generated by the measurement.

Values

Detailed	
Off	
Simple	

EYEOVERSHOOTNEGATIVE

app.Measure.Px.Operator (ParamEngine = "EyeOvershootNegative")

Aperture	Double

Aperture ***Double***

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

EYEOVERSHOOTPOSITIVE

app.Measure.Px.Operator (ParamEngine = "EyeOvershootPositive")

Aperture	Double

Aperture ***Double***

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

EYEPKPKNOISE

app.Measure.Px.Operator (ParamEngine = "EyePkPkNoise")

Aperture	Double
Level	Enum

Level ***Enum***

Description

Defines the level for which the peak noise is computed, One or Zero.

Values

One	
Zero	

Aperture ***Double***

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

EYEQFACTOR

app.Measure.Px.Operator (ParamEngine = "EyeQFactor")

Aperture	Double
CursorDisplay	Enum

Aperture ***Double***

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

CursorDisplay ***Enum***

Description

Set/Query the CursorDisplay cvar. This defines whether the source trace is annotated with 'Help Markers' generated by the measurement.

Values

Detailed	
Off	
Simple	

EYERISETIME

app.Measure.Px.Operator (ParamEngine = "EyeRiseTime")

Aperture	Double
HighPct	Double
LowPct	Double

LowPct *Double*

Range From 10 to 40 step 1

Description

High level in percent.

HighPct *Double*

Range From 60 to 90 step 1

Description

High level in percent.

Aperture *Double*

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

EYESDEVNOISE

app.Measure.Px.Operator (ParamEngine = "EyeSDEVNoise")

Aperture	Double
Level	Enum

Level *Enum*

Description

Defines the level for which the standard deviation is computed, One or Zero.

Values

One	
Zero	

Aperture ***Double***

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

EYESIGNALTONOISE

app.Measure.Px.Operator (ParamEngine = "EyeSignalToNoise")

Aperture	Double
Level	Enum

Level ***Enum***

Description

Specifies if "noise" is to be determined from the one level, the zero level or both levels

Values

Both	
One	
Zero	

Aperture ***Double***

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

EYESUPPRESSIONRATIO

app.Measure.Px.Operator (ParamEngine = "EyeSuppressionRatio")

Aperture	Double
Output	Enum

Aperture ***Double***

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

Output ***Enum***

Values

db	
linear	
pct	

EYEZEROLEVEL

app.Measure.Px.Operator (ParamEngine = "EyeZeroLevel")

Aperture	Double

Aperture ***Double***

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

FALLATLEVEL

app.Measure.Px.Operator (ParamEngine = "FallAtLevel")

HighPct	Double
LevelsAre	Enum
LowPct	Double
SetLevel1090	Action
SetLevel2080	Action

LevelsAre *Enum*

Description

Type of level: absolute, percent, %PkPk or %0-Min with EMC option.

Values

Absolute	
Percent	

LowPct *Double*

Range From 5 to 90 step 1

Description

High level in percent.

HighPct *Double*

Range From 10 to 95 step 1

Description

High level in percent.

SetLevel1090 *Action*

Description

Set the levels to 10% and 90% of full amplitude.

SetLevel2080 *Action*

Description

Set the levels to 20% and 80% of full amplitude.

FULLWIDTHATXX

app.Measure.Px.Operator (ParamEngine = "FullWidthAtXX")

HFractionHt	Double

HFractionHt ***Double***

Range From 0 to 100 step 1

Description

Percentage at which the width of the histogram is estimated (e.g. 50% gives the same answer as Full Width at Half Max).

LEVELATX

app.Measure.Px.Operator (ParamEngine = "LevelAtX")

HorValue	Double
PinToData	Bool
TimeFromCvar	Bool

HorValue ***Double***

Range From -1.79769e+308 to 1.79769e+308 step 0

Description

if the "TimeFromCvar" is set to true, this specifies the horizontal (x) coordinate at which the waveform data "level" or value is to be evaluated

PinToData ***Bool***

Description

If set to true, the vertical values are restricted to actual data points, else if false the values can be interpolated (linearly) between points.

TimeFromCvar ***Bool***

Description

If true, the horizontal coordinate (typically time) is specified by the "HorValue" cvar, otherwise the time is specified by the input pin.

MATLABPARAMETER

app.Measure.Px.Operator (ParamEngine = "MATLABParameter")

MATLABCode	String
MATLABPlot	Bool

MATLABCode ***String***

Range Any number of characters

Description

String containing the MATLAB code to execute when new data is presented.

MATLABPlot ***Bool***

Description

If true, the result of the MATLAB processing operation is plotted by MATLAB, in a floating window.

MAXIMUM

app.Measure.Px.Operator (ParamEngine = "Maximum")

Calculates the maximum vertical value of the waveform

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")
app.Measure.P1.ParamEngine = "Maximum"
```

MEAN

app.Measure.Px.Operator (ParamEngine = "Mean")

Calculates the mean value of the input waveform's vertical values. When Cyclic = true, the range of values used is limited to a whole number of cycles.

Cyclic	Bool

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set parameter P1 to mean.
app.Measure.P1.ParamEngine = "Mean"
' Set the mean parameter for cyclic measurements.
app.Measure.P1.Operator.Cyclic = true
```

Cyclic ***Bool***

Description

Sets/Queries whether the mean parameter Px is to be measured over a number of complete cycles.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set parameter P2 to mean.
app.Measure.P2.ParamEngine = "Mean"
' Set the mean parameter for cyclic measurements.
app.Measure.P2.ParamEngine.Cyclic = True
```

MEDIAN

app.Measure.Px.Operator (ParamEngine = "Median")

Calculates the median (division between two halves) of the probability distribution of an input waveform. For periodic signals it is advisable to use Cyclic = true.

Cyclic	Bool

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")
app.Measure.P1.ParamEngine = "Median"
' Set the measurement for a periodic signal
app.Measure.P1.Operator.Cyclic = true
```

Cyclic ***Bool***

Description

Sets/Queries whether the median parameter Px is to be measured over a number of complete cycles.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set parameter P2 to median.
app.Measure.P2.ParamEngine = "Median"
' Set the median parameter for cyclic measurements.
app.Measure.P2.Operator.Cyclic = True
```

MINIMUM

app.Measure.Px.Operator (ParamEngine = "Minimum")

Calculates the minimum value of a waveform

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

app.Measure.P1.ParamEngine = "Minimum"
```

NPOINTS

app.Measure.Px.Operator (ParamEngine = "npoints")

UsePointsInFrame	Bool

UsePointsInFrame ***Bool***

Description

Choose if the returned value is only points inside the displayed frame, or if all points in the result are reported.

OVERSHOOTNEGATIVE

app.Measure.Px.Operator (ParamEngine = "OvershootNegative")

OVERSHOOTPOSITIVE

app.Measure.Px.Operator (ParamEngine = "OvershootPositive")

PEAKTOPEAK

app.Measure.Px.Operator (ParamEngine = "PeakToPeak")

PERCENTILE

app.Measure.Px.Operator (ParamEngine = "Percentile")

HPctPop	Double
PctRes	DoubleLockstep

HPctPop ***Double***

Range From 0 to 100 step 1

Description

Sets/Queries the percentage of the population which falls to the left (or below) the desired percentile. For example, the median is the 50th percentile, or the horizontal coordinate of the histogram at which 50% of the population falls to the left.

PctRes ***DoubleLockstep***

Range From 1e-006 to 1 step 0.01, locked to 1 2 5, fine grain allowed=false, on=false

Description

This control allows you to control the precision or resolution in the percentage. The default is 1%. But you can set the resolution to as low as 1e-6 % (one part in 1e8). This is useful for finding approximate confidence limits.

PERSISTDCD

app.Measure.Px.Operator (ParamEngine = "PersistDCD")

Aperture	Double
HighPct	Double
LowPct	Double

LowPct ***Double***

Range From 10 to 40 step 1

Description

High level in percent.

HighPct ***Double***

Range From 60 to 90 step 1

Description

High level in percent.

Aperture ***Double***

Range From 0 to 100 step 0.1

Description

For eye-diagram parameters which have an "aperture" setting, this defines the region over which the eye digrams vertical information is analyzed.
It specifies the percentage of the central region of the eye (relative to 1 Unit Interval) which is used in the analysis.

OutputType	Enum
RefFindLevel	Action
RefHysteresis	Double
RefLevelType	Enum
RefPercentLevel	Double
RefSlope	Enum
SigFindLevel	Action
SigHysteresis	Double
SigLevelType	Enum
SigPercentLevel	Double
SigSlope	Enum

OutputType *Enum*

Description

Sets/Queries the output type for Phase Px.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set parameter P1 to phase difference.
app.Measure.P1.ParamEngine = "Phase"
' Set the output unit as radians.
app.Measure.P1.Operator.OutputType = "Radians"
```

Values

Degrees	
Percent	
Radians	

RefLevelType *Enum*

Description

Sets/Queries the unit of measurement for the test level of the reference trace.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set parameter P1 to phase difference.
app.Measure.P1.ParamEngine = "Phase"
' Set the reference level to be measured in absolute units.
app.Measure.P1.Operator.RefLevelType = "Absolute"
```

Values

Absolute	
Percent	

SigLevelType *Enum*

Description

Sets/Queries which level to use "Percent" or "Absolute" for transitions on the signal

Values

Absolute	
Percent	

RefHysteresis *Double*

Range From 0 to 10 step 0.1

Description

Sets/Queries the hysteresis range for the reference trace.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set parameter P1 to phase difference.
app.Measure.P1.ParamEngine = "Phase"
' Set the reference hysteresis in graticule divisions.
app.Measure.P1.Operator.RefHysteresis = 0.7
```

SigHysteresis *Double*

Range From 0 to 10 step 0.1

Description

Sets/Queries the hysteresis range for the signal.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set parameter P3 to phase difference.
app.Measure.P3.ParamEngine = "Phase"
' Set the signal hysteresis in graticule divisions.
app.Measure.P3.Operator.SigHysteresis = 0.7
```

RefFindLevel *Action*

Description

Find the test level for the reference trace.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set parameter P3 to phase difference.
app.Measure.P3.ParamEngine = "Phase"
' Find the test level for the reference trace.
app.Measure.P3.Operator.RefFindLevel
```

SigFindLevel *Action*

Description

Causes the engine to find a suitable level for either SigLevelType ("Absolute"or "Percent")

RefPercentLevel *Double*

Range From 0 to 100 step 1

Description

Sets/Queries the test level for the reference trace in percent.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set parameter P3 to phase difference.
app.Measure.P3.ParamEngine = "Phase"
' Set the reference test level in percent.
app.Measure.P3.Operator.RefPercentLevel = 55
```

SigPercentLevel *Double*

Range From 0 to 100 step 1

Description

Sets/Queries the test level for the signal in percent.

RefSlope *Enum*

Description

Sets/Queries the polarity of the measured reference transitions.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Set parameter P1 to phase difference.
app.Measure.P1.ParamEngine = "Phase"
' Set the reference slope to negative.
app.Measure.P1.Operator.RefSlope = "Neg"
```

Values

Both	
Neg	
Pos	

SigSlope *Enum*

Description

Sets/Queries the polarity of the measured signal transitions.

Values

Both	
Neg	
Pos	

POPULATIONATX

app.Measure.Px.Operator (ParamEngine = "PopulationAtX")

CursorShape	Enum
HorValue	Double

HorValue *Double*

Range From -1.79769e+308 to 1.79769e+308 step 0

CursorShape *Enum*

Values

Absolute	
Difference	
Reference	

RANGE

app.Measure.Px.Operator (ParamEngine = "Range")

BER	DoubleLockstep
ErrorBias	Enum
PctRes	DoubleLockstep
Percent1	Double
Percent2	Double
SetPctFromBER	Bool
TxDensity	Double
UsePercentiles	Bool
VarBER	Bool

Percent1 *Double*

Range From 0 to 100 step 0.001

Description

Leftmost percentile setting (default 0.0), and always 0.0 is "UsePercentiles" is false. Otherwise you can use this control directly or indirectly to set the leftmost percentile (confidence level expressed as a percent), or this can be set automatically from BER and ErrorBias.

Percent2 *Double*

Range From 0 to 100 step 0.001

Description

Rightmost percentile setting (default 0.0), and always 0.0 is "UsePercentiles" is false. Otherwise you can use this control directly or indirectly to set the rightmost percentile (confidence level expressed as a percent), or this can be set automatically from BER and ErrorBias.

PctRes *DoubleLockstep*

Range From 1e-010 to 1 step 1e-005, locked to 1 2 4 5, fine grain allowed=false, on=false

Description

When UsePercentiles is set to true, this control allows control over the resolution of percentile settings (down to 1 part in 10 to the 8th)

BER *DoubleLockstep*

Range From 1e-016 to 0.2 step 0.001, locked to 1 2 5, fine grain allowed=true, on=false

Description

The perdentile settings (left and right) can be set automatically (if SetPctFromBER is true) to correspond to a this Bit Error Ratio (BER). If "SetPctFromBER" is not true, then this control variable is ignored.

ErrorBias *Enum*

Description

When setting percentiles from BER, This control variable permits a symmetric error (i.e. errors to either side of mean are equally important), or a left biased (from median to left confidence limit) or right biased percentile (from median to right confidence limit. All three of these ranges can be of interest depending on what variations can cause an "error"

Values

medianLeft	
medianRight	
symmetric	

UsePercentiles *Bool*

Description

If false (the default) then the range calculated by this processor is really from the leftmost populated bin to the end of the rightmost populated bin. If true, then the range can be calcualted using percentiles. Once this choice is set to true, you can then further set the percentiles indirectly by specifying a BER. See SetPctFromBER control variable.

SetPctFromBER *Bool*

Description

If "UsePercentile" is set to true, and this control is set to true, then the percentiles are set using BER and the error bias settings. If "UsePercentile" is false (default) this setting is ignored.

TxDensity *Double*

Range From 0.01 to 1 step 0.01

Description

When setting percentile limits from BER, this control variable permits an adjustment to the confidence interval to account for
other statistical weighting (such as trnasition density in serial data timing analysis).

VarBER *Bool*

Description

When BER is used to set the percentile levels for left and right, this control allows the BER to be much more finely adjusted.

RISEATLEVEL

app.Measure.Px.Operator (ParamEngine = "RiseAtLevel")

HighPct	Double
LevelsAre	Enum
LowPct	Double
SetLevel1090	Action
SetLevel2080	Action

LevelsAre *Enum*

Description

Type of level: absolute, percent, %PkPk or %0-Min with EMC option.

Values

Absolute	
Percent	

LowPct *Double*

Range From 5 to 90 step 1

Description

High level in percent.

HighPct *Double*

Range From 10 to 95 step 1

Description

High level in percent.

SetLevel1090 *Action*

Description

Set the levels to 10% and 90% of full amplitude.

SetLevel2080 *Action*

Description

Set the levels to 20% and 80% of full amplitude.

ROOTMEANSQUARE

app.Measure.Px.Operator (ParamEngine = "RootMeanSquare")

Cyclic	Bool

Cyclic *Bool*

Description

If true, the calculation is limited to a whole number of cycles detected in the input.

STANDARDDEVIATION

app.Measure.Px.Operator (ParamEngine = "StandardDeviation")

Cyclic	Bool

Cyclic ***Bool***

Description

If true, the calculation is limited to a whole number of cycles detected in the input.

TDRCAPIND

app.Measure.Px.Operator (ParamEngine = "TDRCapInd")

CursorTimePos1	Double
CursorTimePos2	Double
Vi	Double
Z0	Double

Z0 ***Double***

Range From 0 to 1000 step 0.1

Description

The characteristic impedance of the medium under observation

Vi ***Double***

Range From -1 to 1 step 0.001

Description

Incident voltage (which is set internally)

CursorTimePos1 ***Double***

Range From -1.79769e+308 to 1.79769e+308 step 0

Description

This (one of two) time positions, delimits the range of the feature in the TDR signal to be analyzed for the purpose of estimatingof capacitance and inductance.

CursorTimePos2 *Double*

Range From -1.79769e+308 to 1.79769e+308 step 0

Description

This (one of two) time positions, delimits the range of the feature in the TDR signal to be analyzed for the purpose of estimatingof capacitance and inductance.

TIE

app.Measure.Px.Operator (ParamEngine = "TIE")

TIE is "Time Interval Error", or the error in expected arrival time of trnasotions in either a data stream or a clock signal. It is the heart of most jitter measurments (where only one signal is under analysis). The Skew processor is closely related to this function in cases where measurements are using a separate reference clock.

WARNING: The TIE processor is appropriate for analysis of "real-time" acquired waveforms for jitter and timing variations. It will give incorrect results for equivalent-time type waveforms.

BaseFrequency	Double
DataIsNRZ	Bool
FindBaseFrequency	Action
FindLevel	Action
FrequencyMultiplier	Double
Hysteresis	Double
IncludeVirtualEdges	Bool
IntervalsEdgeEdge	Integer
IntervalType	Enum
LevelType	Enum
MaxComboIntervals	Integer
PercentLevel	Double
SignalType	Enum
Slope	Enum
Summary	String
UseAllEdges	Bool
UseBaseFrequency	Enum
UseMultiEdgeCombos	Bool
UsePLL	Bool

LevelType *Enum*

Values

Absolute	
Percent	

PercentLevel *Double*

Range From 0 to 100 step 1

FindLevel *Action*

Description

Activate this control to find the vertical level for 50% for the waveform presented at the input to this processor. (Only for LevelType = "Absolute"

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Automatically find the level for 50% crossing as an absolute (vertical
units) value
app.Measure.P3.Operator.FindLevel
```

Hysteresis *Double*

Range From 0 to 10 step 0.1

Description

This setting establishes the zone around the level (or threshold for level crossings) which must be traversed by the signal in order for the transition to be "qualified".

Slope *Enum*

Values

Both	
Neg	
Pos	

FindBaseFrequency *Action*

Description

When activatred starts an automatic process to attempt to learn the base frequency of the clock or data signal (see DataIsNRZ) and set the BaseFrequency contorl to the found value. Warning: be sure to provide as many clock or data cycles as is reasonably possible for maximum precision in the result.

Example

```
' Visual Basic Script
Set app = CreateObject("LeCroy.XStreamDSO")

' Automatically find the frequency for clock or data signal at the input
of the TIE processor.
app.Measure.P3.Operator.FindBaseFrequency
```

SignalType *Enum*

Values

Clock	
Data	

BaseFrequency *Double*

Range From 1 to 1e+011 step 1000

Description

This is the frequency of used to provide expected times for TIE. If a PLL is being used, this frequency must be within the capture range for the PLL (usually quite close) or you will encounter unexpected results. Typically the find frequency operation is quite good for setting this value, unless the input source is "data" and the signal is very stressed (closed eye or nearly closed eye).

FrequencyMultiplier *Double*

Range From 0.001 to 1000 step 0.001

UseBaseFrequency *Enum*

Values

Custom	
Standard	

IncludeVirtualEdges *Bool*

Description

When this control is set to true (default is false), the TIE values the result at the output have "virtual edges" included in the output (i.e. values which are linearly interpolated) corresponding to edges which did NOT transit. This allows a relatively uniform in time strm of values. This feature is mostly obviated by the processor "ParamUpSample"

DataIsNRZ *Bool*

Description

This should be set to "true" for TIE analysis of an NRZ data stream. It should be set to false for TIE analysis of a clock signal

IntervalType *Enum*

Description

Timing Analysis can either performed using the edge-edge timing methodology (as was developed for Time-Interval-Analyzers", or edge-ref, as is common for real-time oscilloscopes. Edge-Ref is highly recommended.

Values

EDGEEDGE	
EDGEREF	

IntervalsEdgeEdge *Integer*

Range From 1 to 500 step 1

Description

For edge-edge methodology, this control sets the number of UI (unit intervals) between edges to be analyzed.

Summary *String*

Range Any number of characters

UseMultiEdgeCombos *Bool*

MaxComboIntervals *Integer*

Range From 1 to 20000 step 1

UsePLL *Bool*

UseAllEdges *Bool*

TIMEATLEVEL

app.Measure.Px.Operator (ParamEngine = "TimeAtLevel")

FindLevel	Action
Hysteresis	Double
LevelType	Enum
PercentLevel	Double
Slope	Enum
Summary	String

LevelType *Enum*

Description

Level type in absolute, percent and %Pkpk, %0-min, %0-max with EMC option.

Values

Absolute	
Percent	

PercentLevel *Double*

Range From 0 to 100 step 1

Description

Level in percent.

FindLevel *Action*

Description

When in absolute level, finds the level at 50%.

Hysteresis *Double*

Range From 0 to 10 step 0.1

Description

Hysteresis around level in units of divisions.

Slope *Enum*

Description

Slope of the detected transitions.

Values

Both	
Neg	
Pos	

Summary *String*

Range Any number of characters

Description

Summary of functionality and settings of processor.

TOP

app.Measure.Px.Operator (ParamEngine = "Top")

XATMAXIMUM

app.Measure.Px.Operator (ParamEngine = "XAtMaximum")

HystDiv	Double
Method	Enum

Method ***Enum***

Values

LeftmostMax	
LocalMaxima	
RightmostMax	

HystDiv ***Double***

Range From 0.1 to 5 step 0.05

XATMINIMUM

app.Measure.Px.Operator (ParamEngine = "XAtMinimum")

HystDiv	Double
Method	Enum

Method ***Enum***

Description

Method to use for finding Minima.

Values

LeftmostMin	
LocalMinima	
RightmostMin	

HystDiv ***Double***

Range From 0.1 to 5 step 0.05

XATPEAK

app.Measure.Px.Operator (ParamEngine = "XAtPeak")

PeakNumber	Integer

PeakNumber ***Integer***

Range From 1 to 10000 step 1

Description

Peak number for which the X value is returned.

DCOM

The following items must be known and considered when creating DCOM connections from PC's to LeCroy oscilloscope:

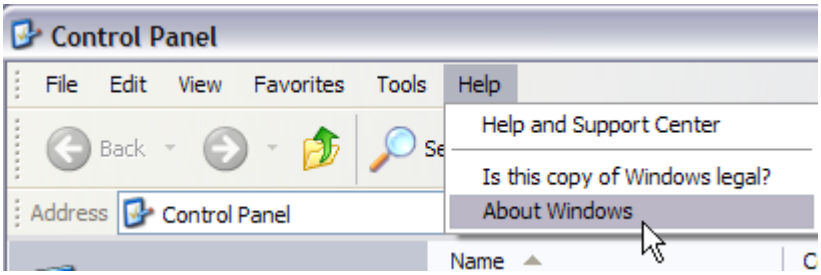
- The Operating System running on both the PC and Oscilloscope
- The level of visibility on an NT domain (if PC and Oscilloscope are even running on the same domain)
- Users must have administrative rights for both PC and Oscilloscope network locations

Note: Be sure to look for your particular Operating System based instructions from **Windows XP or Vista**, **Windows 2000**, and **Windows Embedded**.

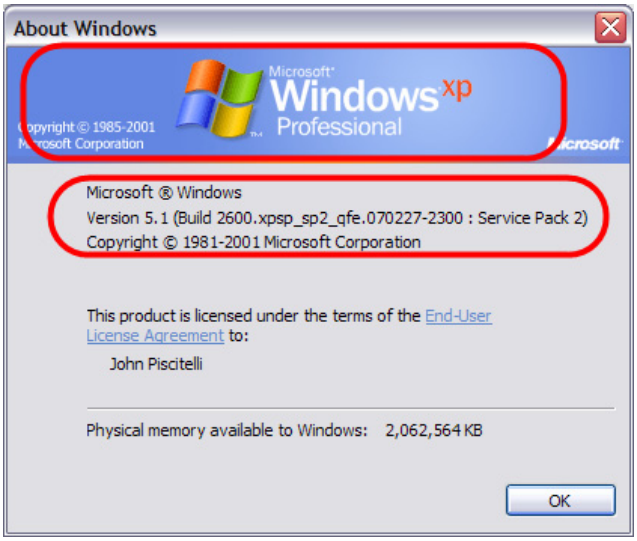
Prepare the PC to Permit DCOM Connections to the Oscilloscope

First, find out what Operating System is running on the PC by performing the following steps:

1. Click **Start →Settings → Control Panel**.
2. On the File Explorer window shown, click **Help →About Windows** from the menu bar.



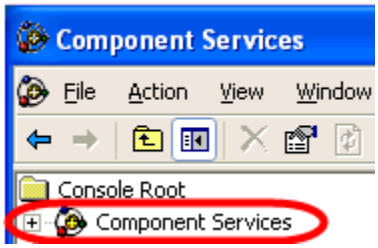
3. The operating system name is shown on the logo displayed in the pop-up. The service pack is shown underneath the logo.



Installation on Windows XP or Vista

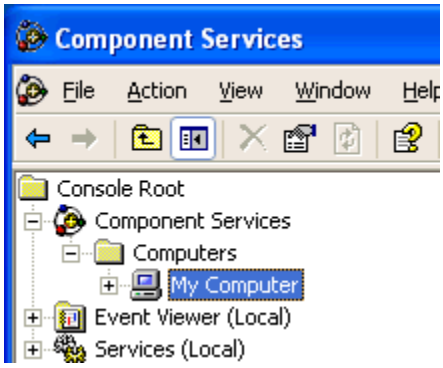
Note: For installation on a PC running Windows 2000, refer to the **Installation on Windows 2000** topic found later in this manual.

1. Install **XStreamBrowserInstall** version 1.0.4 or higher
   - On Windows XP, click **Start** → **Run**, enter **dcomcnfg.exe** and click **OK**.
   - On Windows Vista, open a command prompt console and run **dcomcnfg.exe**.
2. Expand **Component Services** by clicking the + plus sign (this may take a few minutes).

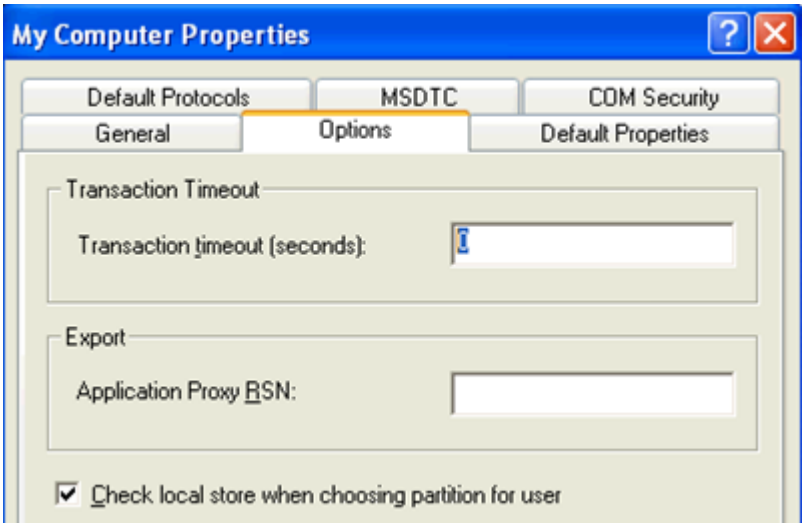


3. If a Windows Security Alert pop-up is shown, click the **Unblock** button.

4. Expand **Computers**, right-click **My Computer**, and select **Properties**.



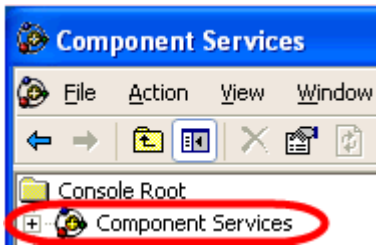
5. Click the **Options** tab on the Properties display, set the **Transaction timeout** to **0**, and click the **Apply** button.



Close the **Properties** display and the **Component Services** window.

Installation on Windows 2000 SP4

1. Install **XStreamBrowserInstall** version 1.0.4 or higher
2. Expand **Component Services** by clicking the + plus sign (this may take a few minutes).

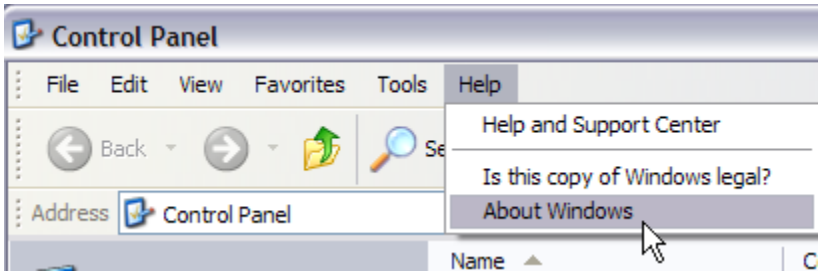


3. If a Windows Security Alert pop-up is shown, click the **Unblock** button.

Oscilloscope Preparation on Windows XP or Vista

First, find out what Operating System is running on the PC by performing the following steps:

1. Click **Start →Settings → Control Panel**.
2. On the File Explorer window shown, click **Help → About Windows** from the menu bar.



3. The operating system name is shown on the logo displayed in the pop-up. The service pack is shown underneath the logo.

Note: For installation on a PC running Windows 2000 or Windows Embedded, refer to the **Oscilloscope Preparation on Windows 2000** or **Oscilloscope Preparation on Windows Embedded** topics found later in this manual.

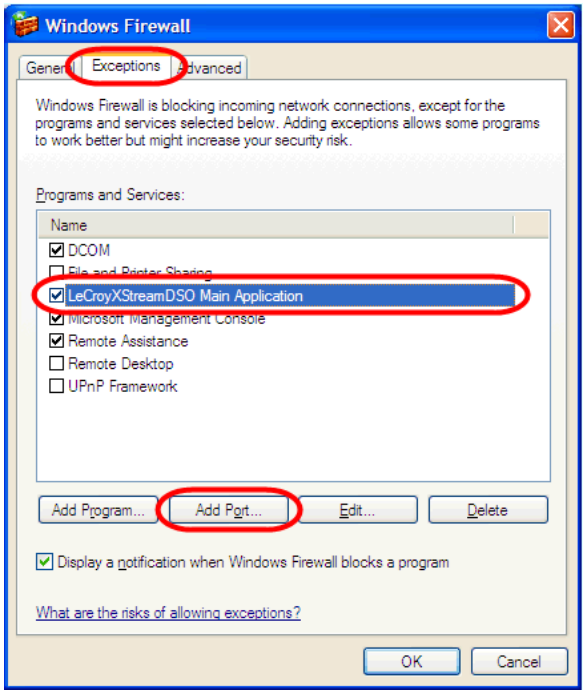
Creating a System Restore Point on Windows XP or Vista

It's always a good idea to create a system restore point before changing any Windows XP system settings. Create one by going to **Start → Programs → Accessories → System Tools → System Restore**.

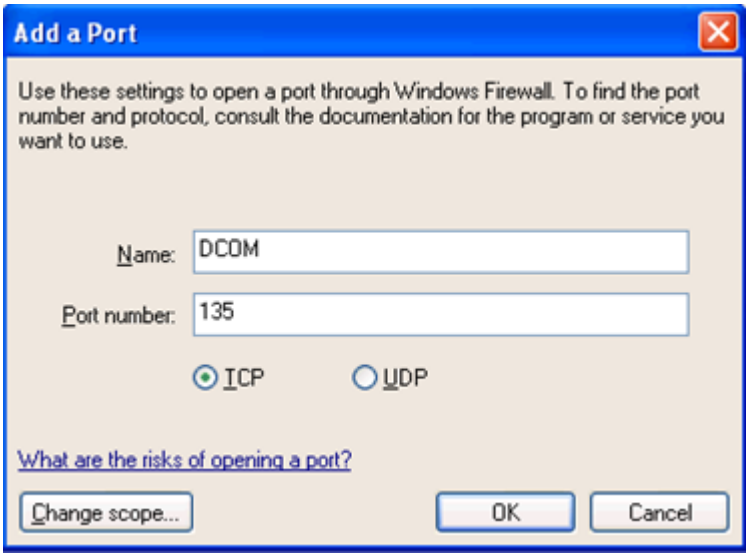
Opening the Firewall on Windows XP or Vista

Now, we have to make sure the Windows XP firewall is open.

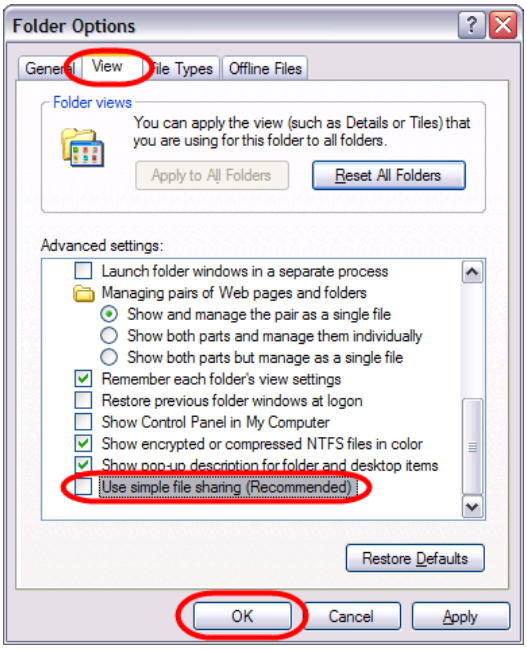
1. Go to **Start → Control Panel → Windows Firewall**.
2. Click the **Exceptions** tab on the **Windows Firewall** display, verify that the **LeCroyXStreamDSO Main Application** item is checked, and click the **Add Port** button as follows.



3. On the Add a Port pop-up, provide port Name (DCOM), Number (135), select the TCP radio button, and click **OK**.



4. Now, if the PC and Oscilloscope are on the same NT domain, turn off simple file sharing by going to **Start** → **Settings** → **Control Panel**. Double-click on **Folder Options**. Click the **View** tab on the **Folder Options** window, scroll to the bottom, **uncheck** the **Use simple file sharing (Recommended)**, and click **OK**.



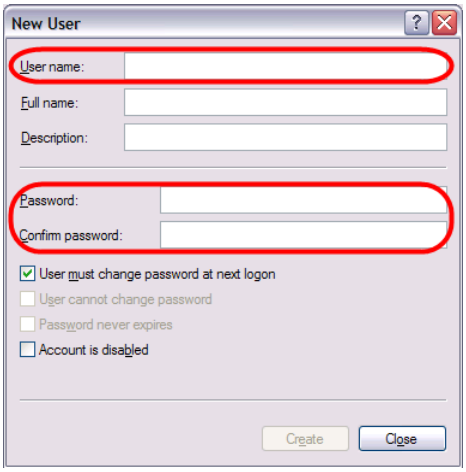
Creating a User Account on Oscilloscopes Running Windows XP or Vista

Note: If the PC and the Oscilloscope reside on the same NT domain, there is no need to create a user account on the Oscilloscope and this topic may be skipped.

If the PC and Oscilloscope do not reside on the same NT domain, a user account must be created on the Oscilloscope. This user account must match the username and password of the one on the PC (the same one on the PC, with administrative rights, etc.). Create a new user account on the oscilloscope as follows:

Right-click on **MyComputer** and select **Manage**. Open **Local Users and Groups** → **Users**. Now, right-click on a blank area in the list of users on the right and select **New User…**

Be sure to match the **username** and **password** fields from the PC's account on the oscilloscope's **New User** window.



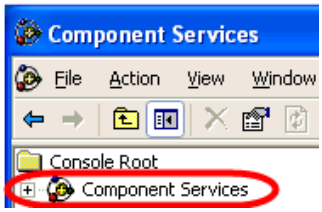
Note: This new account is created to provide DCOM access to the oscilloscope and is not necessarily needed for any other use.

DCOM System Configuration on Windows XP or Vista

Now, on the **PC**, make the following configurations:

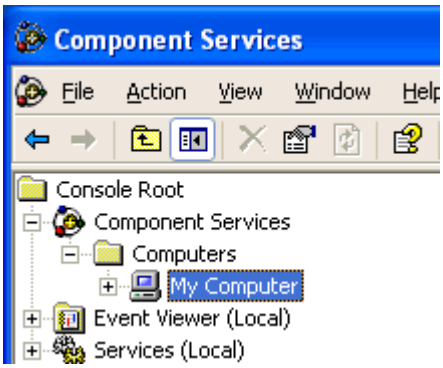
Note: Only proceed with this topic when both the PC and Oscilloscope have already been set with the previous DCOM configuration steps.

1. Click **Start** → **Run**, enter **dcomcnfg.exe** and press **Enter**.
2. Expand **Component Services** by clicking the + plus sign (this may take a few minutes).

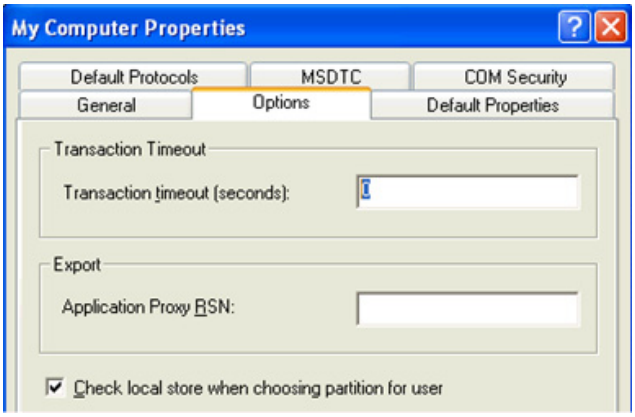


3. If a Windows Security Alert pop-up is shown, click the **Unblock** button.

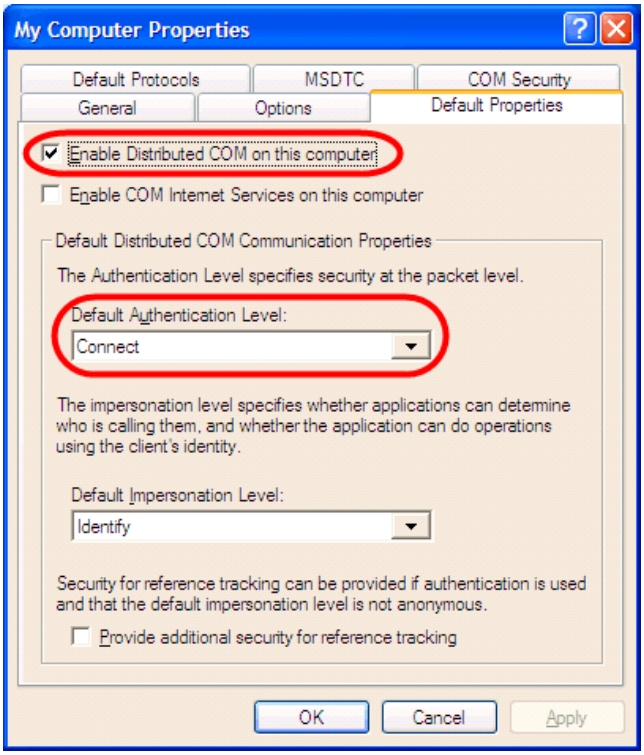
4. Expand **Computers**, right-click **My Computer**, and select **Properties**.



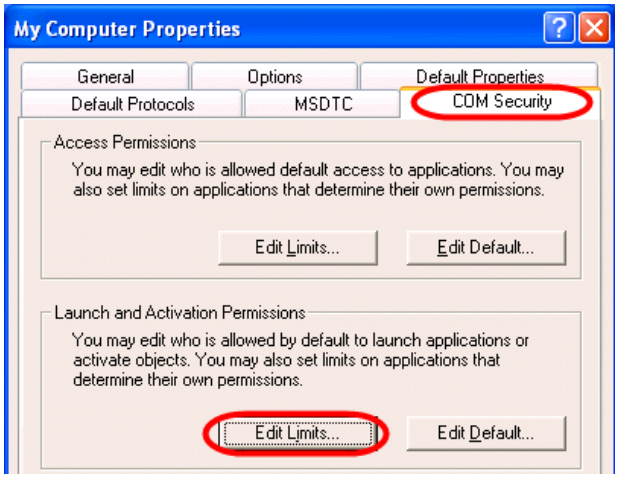
5. Click the **Options** tab on the Properties display, set the **Transaction timeout** to **0**, and click the **Apply** button.



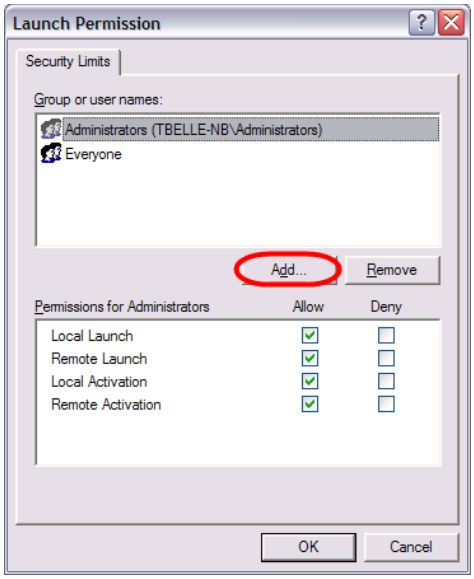
6. Click the **Default Properties** tab and make sure the **Enable Distributed COM on this computer** check-box is checked and **Connect** is selected from the **Default Authentication Level** drop-down.



7. Now, on the COM Security tab, click the Edit Limits button on the Launch and Activation Permissions section.

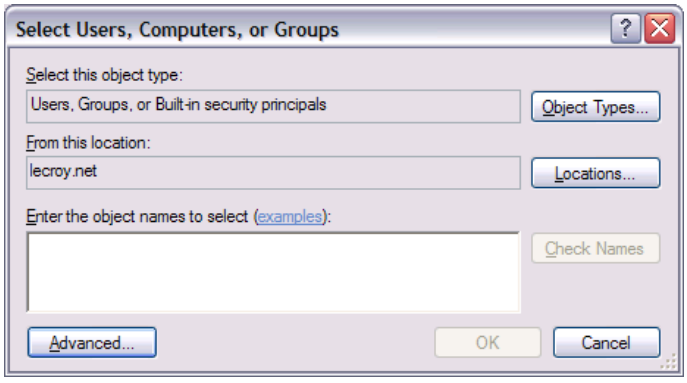


8. Click the **Add** button on the **Launch Permission** dialogue.



- **If the PC and Oscilloscope ARE on the same NT domain:**

  Select the **exact user account used on the PC** on the **Select Users, Computers, or Groups** dialogue and click **OK**.



- **If the PC and Oscilloscope ARE NOT on the same NT domain:**

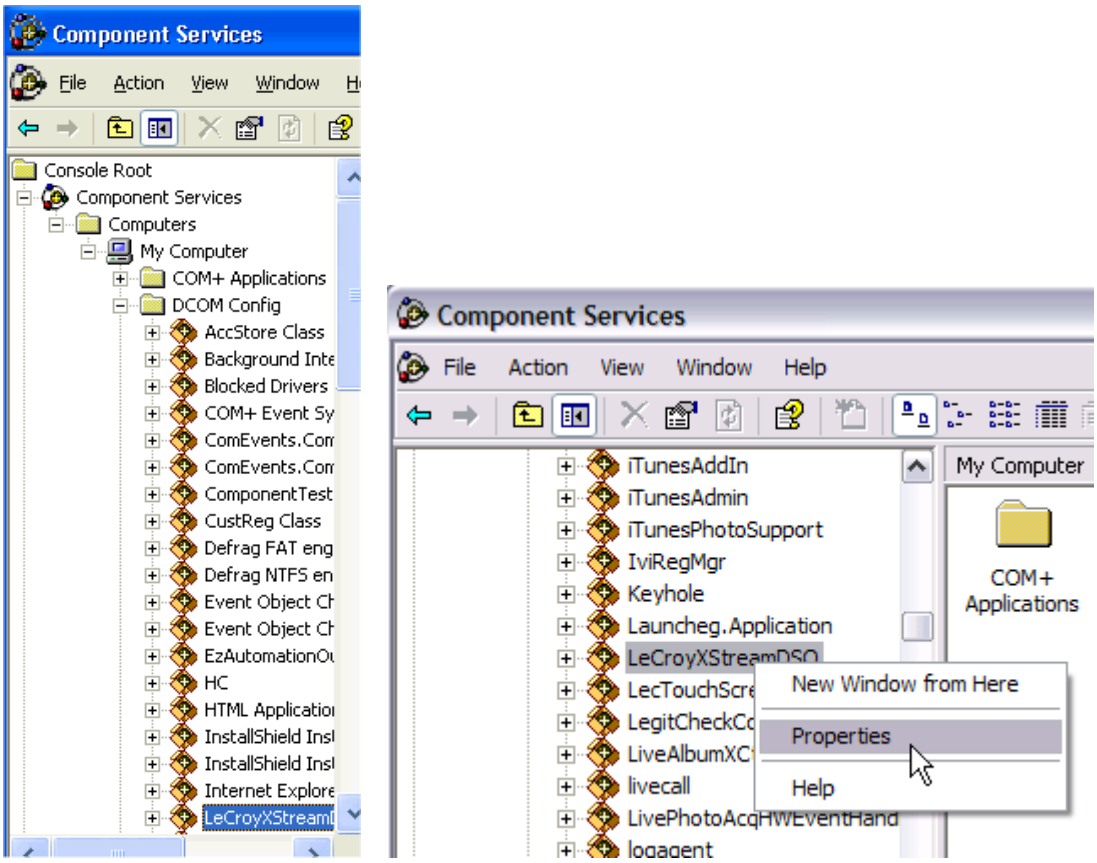
  Select the **user account created on the Oscilloscope** (the one created to match the one on the PC during the previous **Creating a User Account on the Oscilloscope** topic) on the **Select Users, Computers, or Groups** dialogue and click **OK**.

9. Click **OK** to close the **Launch Permissions** dialogue. Now, click **Apply**, and then **OK** on the **COM Security** tab in the **Computer Properties** dialogue.

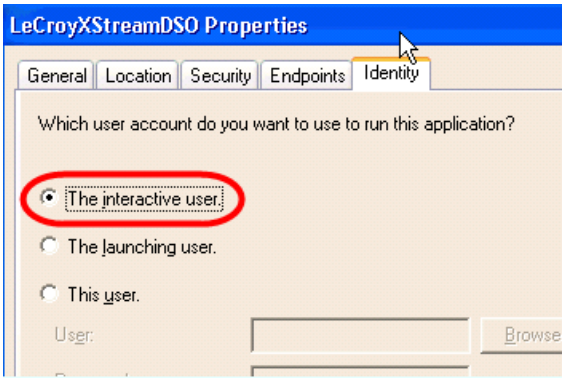
   With all dialogues closed from the previous steps, the **Component Services** window should be shown.

10. Open **My Computer** → **DCOM Config**.

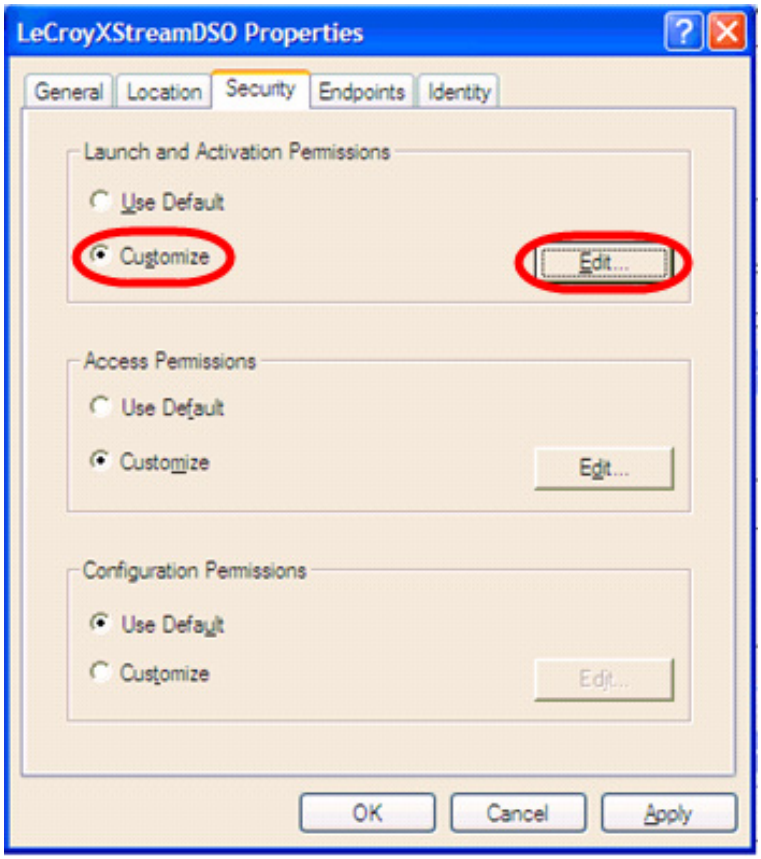
11. Right-click on the **LeCroyXStreamDSO** item and select **Properties**.



12. Click the **Identify** tab on the **LeCroyXStreamDSO Properties** dialog and choose **The interactive user** from the radio buttons.

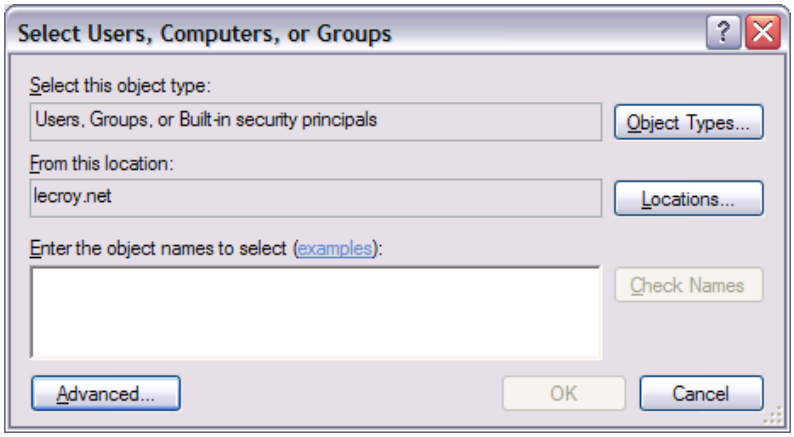


13. On the **Security** tab, select the **Customize** radio button and click the **Edit** button on the **Launch and Activation Permissions** section.



- If the PC and Oscilloscope **ARE** on the same NT domain:

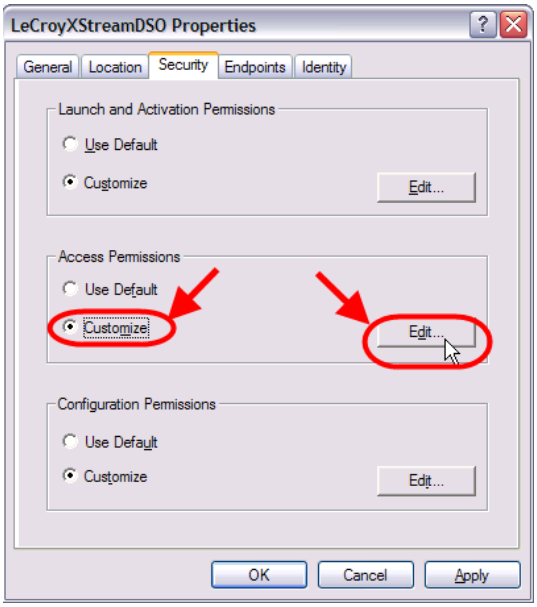
  Select the **exact user account used on the PC** on the **Select Users, Computers, or Groups** dialogue and click **OK**.



- If the PC and Oscilloscope **ARE NOT** on the same NT domain:

  Select the **user account created on the Oscilloscope** (the one created to match the one on the PC during the previous **Creating a User Account on the Oscilloscope** topic) on the **Select Users, Computers, or Groups** dialogue and click **OK**.

14. Now, repeat the process while editing the Access Permission settings from the LeCroyXStreamDSO Properties dialog.



15. Now, click **Start → Shut Down**, select **Restart** and click **OK** to reboot the oscilloscope.
16. When the instrument has rebooted, we can now test the connection by proceeding to the **Testing the DCOM Connection from the PC** topic at the end of this manual.

Oscilloscope Preparation on Windows 2000

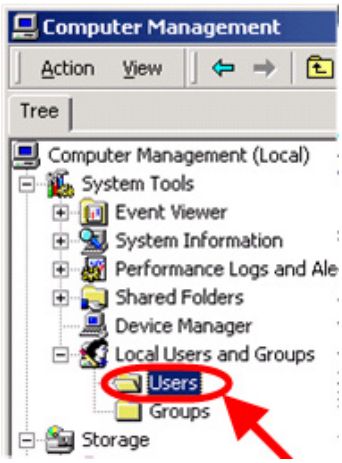
First, we'll prepare the oscilloscope by figuring out whether or not a new username must be configured. If the user account is necessary, we'll create it.

Creating a User Account on Oscilloscopes Running Windows 2000

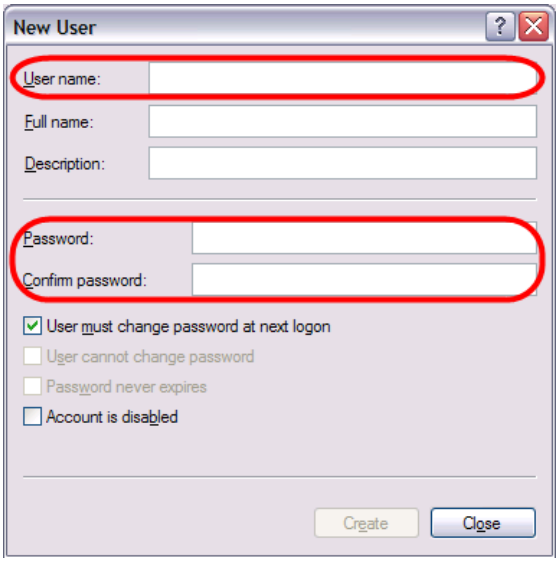
Note: If the PC and the Oscilloscope reside on the same NT domain, there is no need to create a user account on the Oscilloscope and this topic may be skipped.

If the PC and Oscilloscope do not reside on the same NT domain, a user account must be created on the Oscilloscope. This user account must match the username and password of the one on the PC (the same one on the PC, with administrative rights, etc.). Create a new user account on the oscilloscope as follows:

Right-click on **MyComputer** and select **Manage**. Open **Local Users and Groups → Users**. Now, right-click on a blank area in the list of users on the right and select **New User…**



Be sure to match the **username** and **password** fields from the PC's account on the oscilloscope's **New User** window.



Note: This new account is created to provide DCOM access to the oscilloscope and is not necessarily needed for any other use.

If the scope is in the same NT domain as the client PC, you will have to set the **Hostname** (meaning, the scope name, i.e. LCRY0901N11751) to the scope itself.

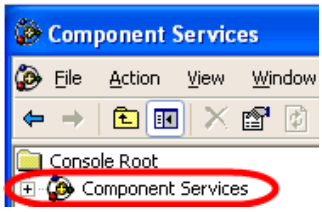
Now, close all open dialogues.

DCOM System Configuration on Windows 2000

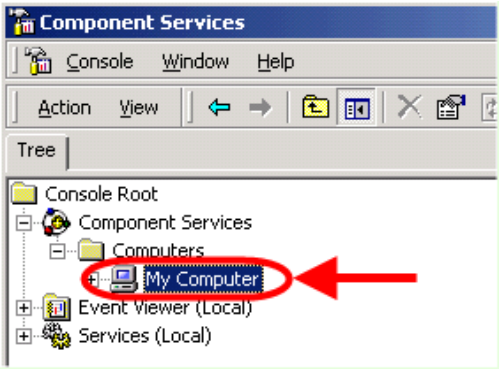
Now, on the **Oscilloscope**, make the following configurations:

Note: Only proceed with this topic when both the PC and Oscilloscope have already been set with the previous DCOM configuration steps.

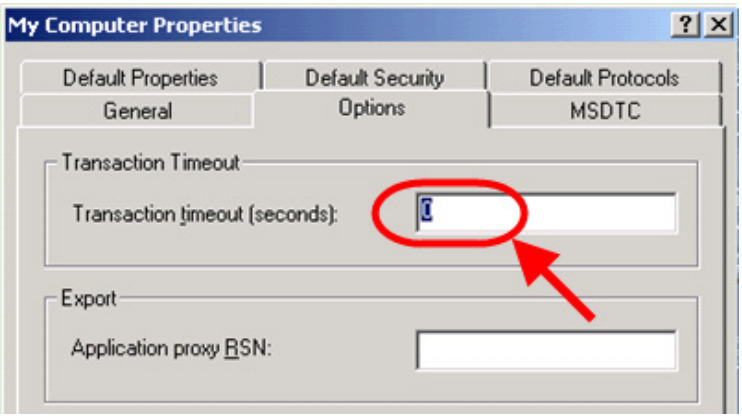
1. Click **Start** → **Settings** → **Control Panel**. On the Control Panel screen, double-click **Administrative Tools** and then **Component Services**.
2. Expand **Component Services** by clicking the + plus sign (this may take a few minutes).



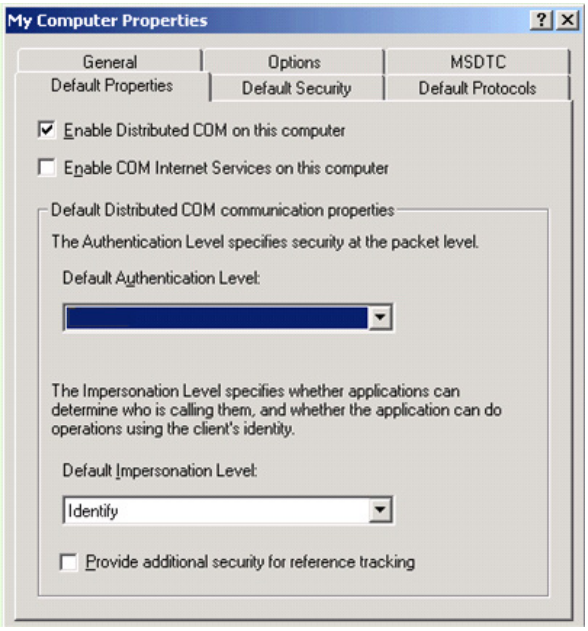
3. Expand **Computers**, right-click **My Computer**, and select **Properties**.



4. Click the **Options** tab on the Properties display, set the **Transaction timeout** to **0**, and click the **Apply** button.



5. Now, click the **Default Properties** tab and determine how you should configure your system from the following choices:



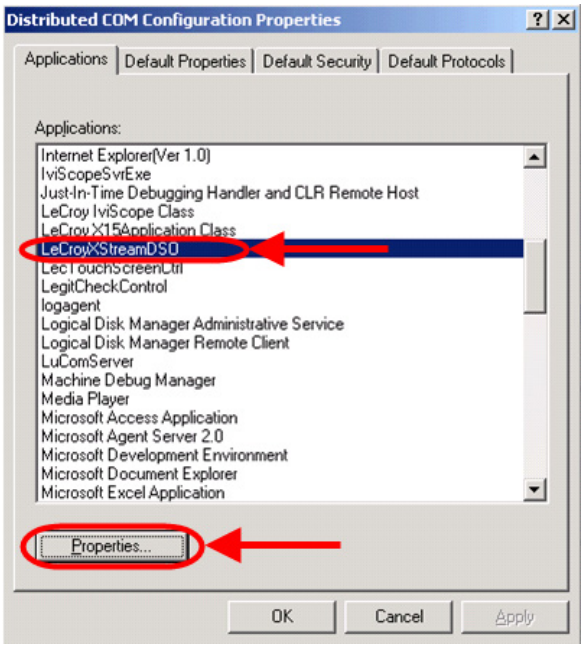
PC and Oscilloscope ARE on the Same NT Domain

- Select **Connect**, **None**, or **Default** on the **Default Impersonation Level** field.

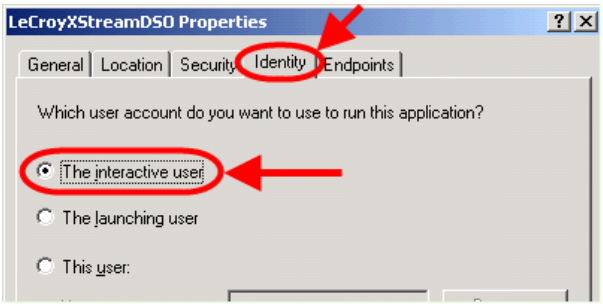
PC and Oscilloscope ARE NOT on the Same NT Domain

- Select **None** on the **Default Impersonation Level** field.

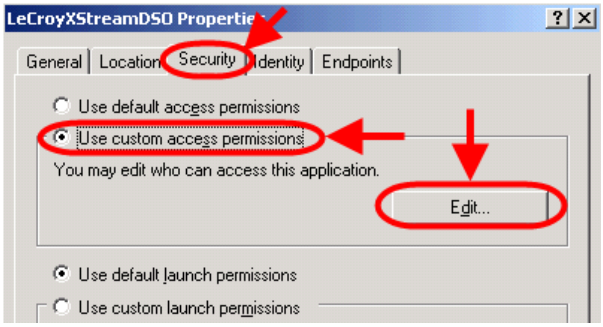
6. Click the **Apply** button, and then close all dialogs.
7. Now, click **Start** → **Run**, enter **dcomcnfg.exe** and press **Enter**.
8. Select **LeCroyXStreamDSO** and click the **Properties** button.



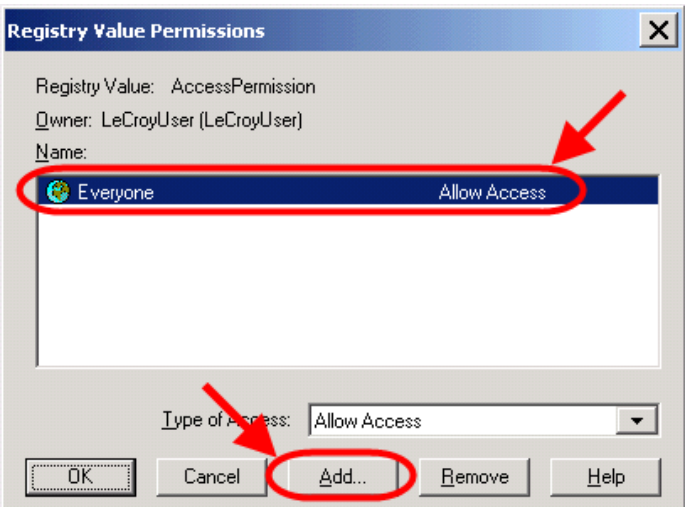
9. On the **Properties** dialog, click the **Identity** tab, and select **The interactive user** radio button.



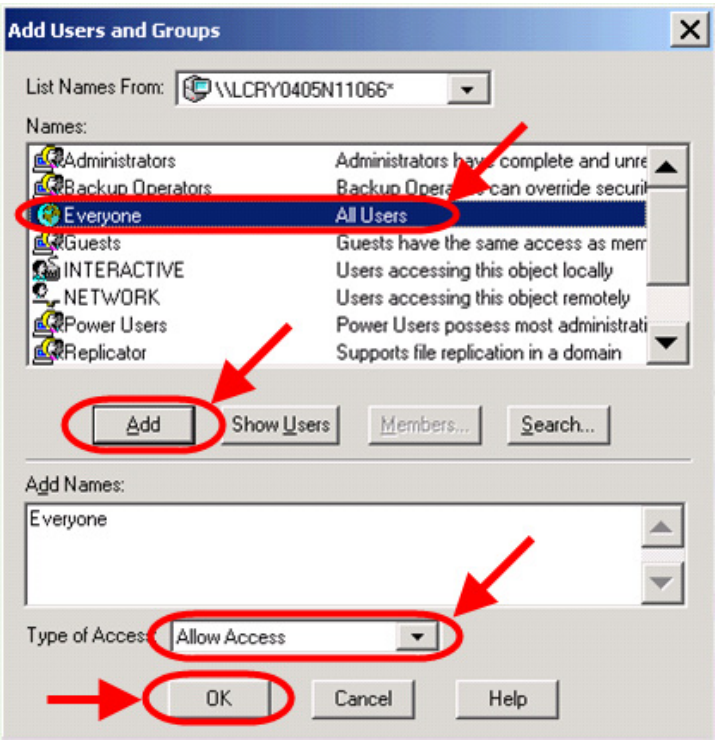
10. Now, click the **Security** tab. Select the **Use custom access permissions** radio button and click the corresponding **Edit** button.



11. The **Registry Value Permissions** dialog is shown. If the **Everyone** name **DOES NOT** have **Allow Access** permission, click the **Add** button.



12. The **Add Users and Groups** dialog is shown.
    - Select **Everyone**
    - click the **Add** button
    - Select **Allow Access**
    - Click **OK**



13. Be sure to click the **Apply** button while closing all dialogs.
14. Now, click **Start** → **Shut Down**, select **Restart** and click **OK** to reboot the oscilloscope.
15. When the instrument has rebooted, we can now test the connection by proceeding to the **Testing the DCOM Connection from the PC** topic at the end of this manual.

Oscilloscope Preparation on Windows Embedded

To prepare the oscilloscope, we first need to open the XP firewall.

Note: Please be advised that after some research we have discovered that the XP Firewall does not work well with DCOM (if at all). One of the problems is that a first connection is established using the default endpoint (port 135), and when a second port is opened it's forced into the 3000-4000 range. Also, XP's SP1 firewall does not allow specified port ranges.

1. Click **Start** → **Control Panel**. Double-click **Network Connections** and then **Local Area Connections**.
2. On the **Local Area Connection Status** dialog, the General tab should be shown.
3. Now, click the **Properties** button and the **Local Area Connection Properties** dialog is shown. Click the **Advanced** tab and make sure the **Protect my computer and network by limiting or preventing access to this computer from the Internet** check-box is **un-checked**.

- If the PC and Oscilloscope **ARE** in the same NT domain, this is a **User Level** configuration and there's **no need to create a new account** on the oscilloscope to allow DCOM connections.
- If the PC and Oscilloscope **ARE NOT** in the same NT domain, this is a **Share Level** configuration and **you must create a new account** on the oscilloscope to allow DCOM connections. The following topic shows you how.

Creating a User Account on Oscilloscopes Running Windows Embedded

Note: If the PC and the Oscilloscope reside on the same NT domain, there is no need to create a user account on the Oscilloscope and this topic may be skipped.

If the PC and Oscilloscope do not reside on the same NT domain, a user account must be created on the Oscilloscope. This user account must match the username and password of the one on the PC (the same one on the PC, with administrative rights, etc.). Create a new user account on the oscilloscope as follows:

1. Click **Start** → **Settings** → **Control Panel**. Double-click on **User Accounts**.
2. Again, when creating this new account, make sure it matches the username and password of the one on the PC (the same one on the PC, with administrative rights, etc.).
3. Select **Limited**, click **Create Account**, and **Change an Account**.
4. Select and highlight the account name just entered and click **Create a Password**. Enter the same password from the PC account twice and click the **Create Password** button.

Note: This new account is created to provide DCOM access to the oscilloscope and is not necessarily needed for any other use.

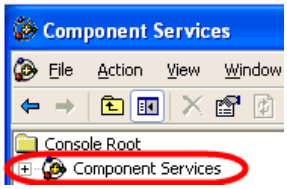
DCOM System Configuration on Windows Embedded (on the Oscilloscope)

Note: Only proceed with this topic when both the PC and Oscilloscope have already been set with the previous DCOM configuration steps.

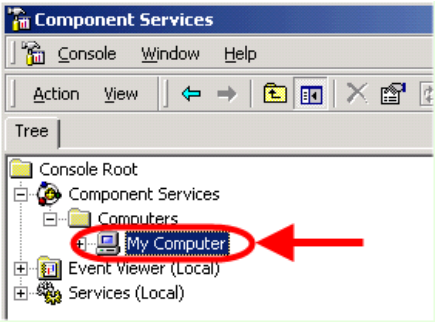
1. Click **Start** → **Run** enter **dcomcnfg.exe** and press **Enter**.

   Alternatively: You can click **Start** → **Control Panel** and double-click **Administrative Tools** and **Component Services**.

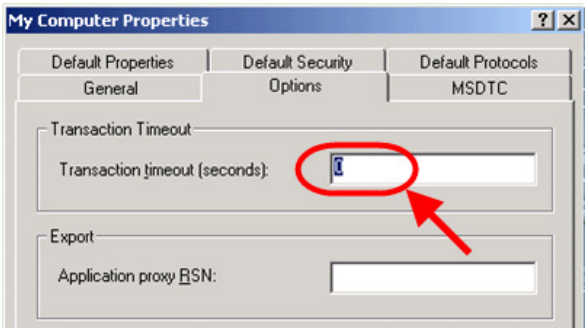
2. Expand **Component Services** by clicking the + plus sign (this may take a few minutes).



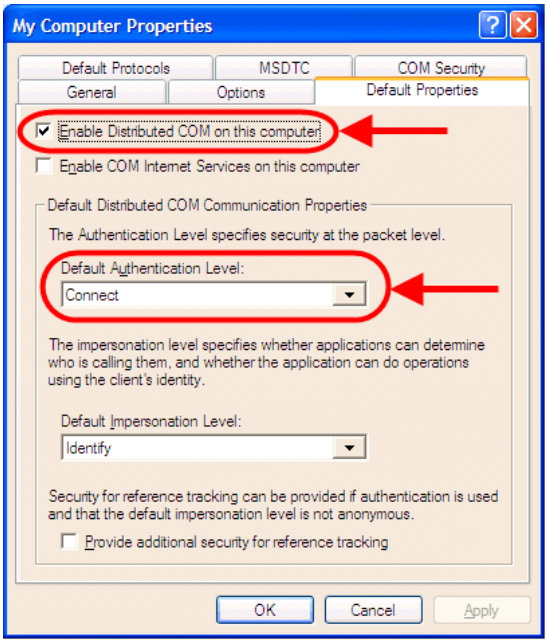
3. Expand **Computers**, right-click **My Computer**, and select **Properties**.



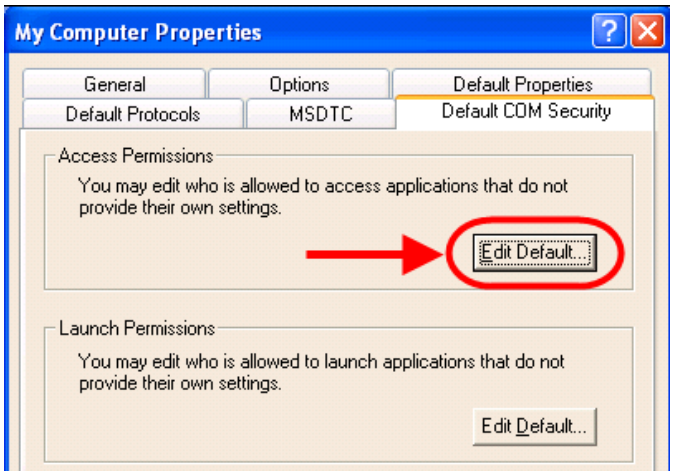
4. Click the **Options** tab on the Properties display, set the **Transaction timeout** to **0**, and click the **Apply** button.



5. Now, click the **Default Properties** tab and make sure the **Enable Distributed COM on this computer** check-box is **checked** and the **Default Authentication Level** drop-down is set to **Connect**.

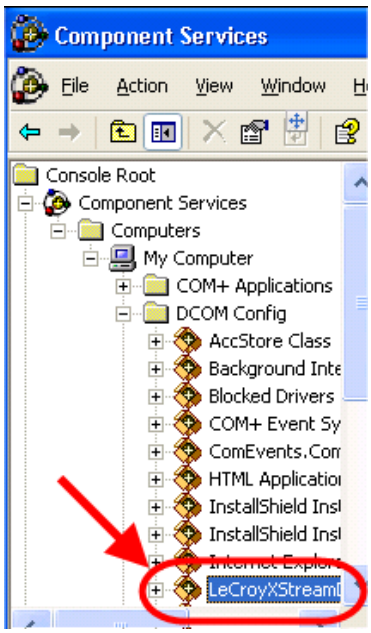


6. Click the **Default COM Security** tab. Click the **Edit Default** button on the **Access Permissions** section of the dialog.

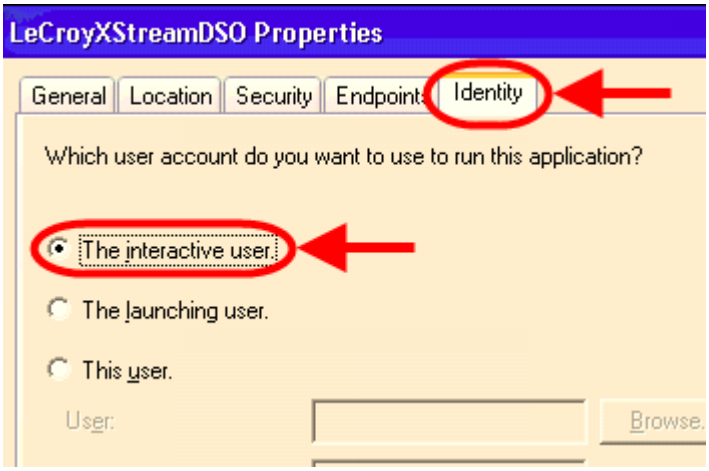


Note: Users sometimes notice the **Default Properties** and **Default COM Security** tab names sometimes are interchanged by Windows.

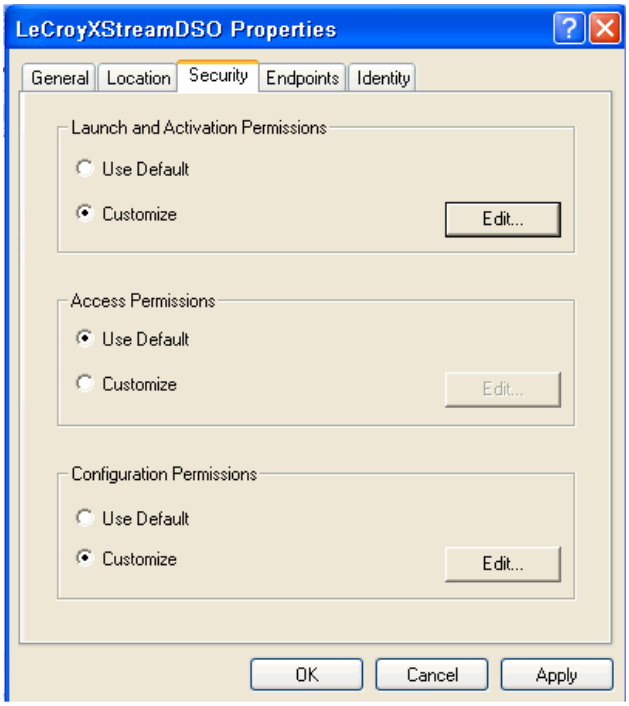
7. The **Access Permissions** dialog is shown. Click the **Add** button.
   - If the PC and Oscilloscope **ARE** in the same NT domain, this is a **User Level** configuration. Provide the exact same user account (username and password) created on the PC (covered previously in this manual).
   - If the PC and Oscilloscope **ARE NOT** in the same NT domain, this is a **Share Level** configuration. Provide the exact same user account (username and password) created on the PC (covered previously in this manual). However, the username must be preceded by the NT **domain name** and a backslash ****. **Example**: LECROY\John.Smith.
8. Verify the **Allow** check-box is checked on the Access Permissions dialog.
9. Click **OK** on the **Access Permissions** dialog and **Apply**, and then **OK** on the **Properties** dialog. The **Component Services** should now be showing.
10. Access the following path from **Component Services**: **Component Services → Computers → My Computer → DCOM Config**. Find and right-click on **LeCroyXStreamDSO** and select **Properties**.



1. On the Properties dialog, click the **Identity** tab and select **The interactive user** radio button.



2. Click the **Security** tab on the **Properties** dialog and select the **Customize** radio button on the **Launch Permissions** section. Click the **Edit** button.



3. On the resulting dialog, click the **Add** button.
   - If the PC and Oscilloscope **ARE** in the same NT domain, this is a **User Level** configuration. Provide the exact same user account (username and password) created on the PC (covered previously in this manual). Click **OK**.
   - If the PC and Oscilloscope **ARE NOT** in the same NT domain, this is a **Share Level** configuration. Provide the exact same user account (username and password) created on the PC (covered previously in this manual). However, the username must be preceded by the NT **domain name** and a backslash \. **Example**: LECROY\John.Smith.
4. Repeat this process (Steps 6-13) for the **Launch Permissions** (on the **Properties** dialog). When finished, be sure to click **Apply** and **OK** on all dialogs and close the **Component Services** window.
5. Now, click **Start** → **Shut Down**, select **Restart** and click **OK** to reboot the oscilloscope.

6. When the instrument has rebooted, test the connection from the PC using the following topic (Testing the DCOM Connection from the PC).

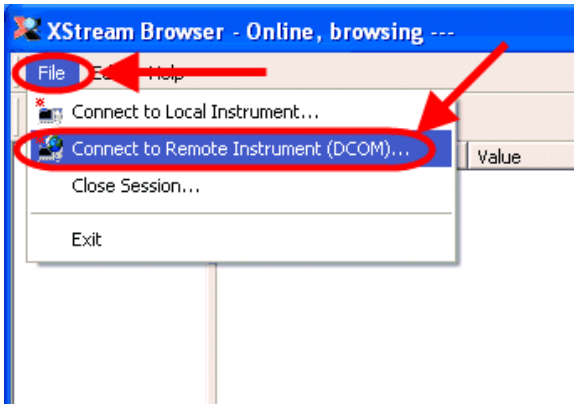
Testing the DCOM Connection from the PC

Note: Only attempt to test the DCOM Connection from the PC after both the PC and Oscilloscope have been set for DCOM configuration.

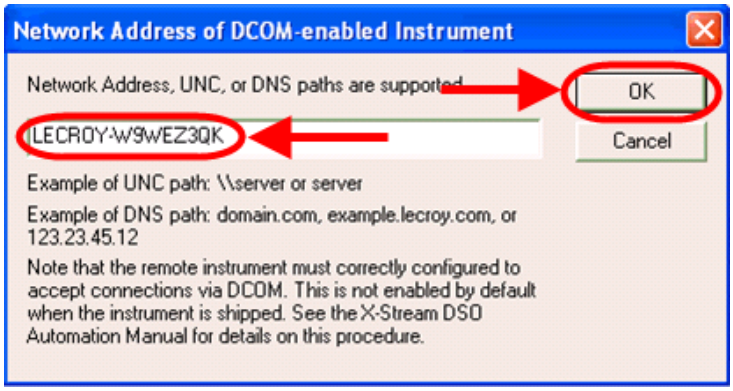
Let's test the DCOM Connection from the PC to the Oscilloscope using the following steps:

1. If not already done, install **XStream Browser**.
2. Launch the **XStream Browser** application on the PC. Make sure that the version of the software is at least 1.0.3 by verifying from the Help/About screen – **Help** (Menu Bar) → **About**.

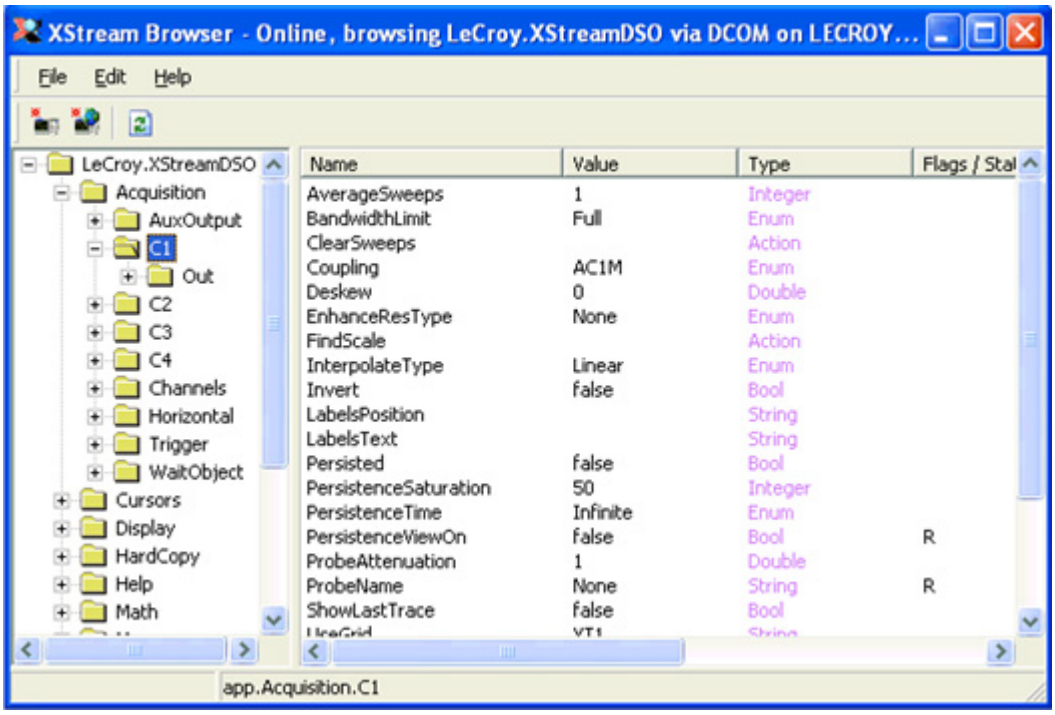
3. On the **XStream Browser** screen, click **File** → **Connect to Remote Instrument (DCOM)**.



4. The **Network Address of DCOM-enabled Instrument** dialog is shown. Provide the **Scope Name** or its **IP Address** and click **OK**.



5. The **XStream Browser** screen eventually shows the **Automation tree** (after clicking **OK** on the **Network Address of DCOM-enabled Instrument** dialog). Click through the tree and access scope values to verify connectivity to the oscilloscope.



GLOSSARY

ActiveX

Microsoft's brand name for the technologies that enable interoperability using the Component Object Model (COM). ActiveX technology includes, but is not limited to, OLE.

Automation

COM-based technology that enables binding at run time, or late binding, to an object's methods and properties and also makes possible cross-application macro programming. Formerly referred to as OLE Automation.

Automation client

An application, programming tool, or scripting language that accesses services provided by Automation objects. Formerly referred to as Automation controller.

Automation object

An instance of a class defined within an application that is exposed for access by other applications or programming tools by Automation interfaces.

Automation server

An application, type library, or other source that makes Automation objects available for programming by other applications, programming tools, or scripting languages.

COM (COMPONENT OBJECT MODEL)

The programming model and binary standard on which OLE is based. COM defines how objects and their clients interact within processes or across process boundaries.

DCOM (DISTRIBUTED COMPONENT OBJECT MODEL)

Distributed form of COM, enables communication between computers using the COM standard.

EARLY BINDING

TODO:

LATE BINDING

The ability to bind member names to dispatch identifiers (IDs) at run time, rather than at compile time. See also ID binding and VTBL binding.

DISPATCH INTERFACES (dispinterface)

An IDispatch interface that responds only to a certain fixed set of names. The properties and methods of the dispinterface are not in the virtual function table (VTBL) for the object.