

Pulsar-4 OphirFastX ActiveX Control

Revision History

09	19-Oct-10	1. Mention release of the OphirLMMeasurement COM object and recommendation that new designs work with that
08	16-May-10	1. Added section on OphirFastXBeta
07	29-Dec-08	1. Modify Registration instructions 2. PC application is now StarLab
06	16-Nov-08	1. Fix section on Registration 2. Added StartCS2 and GetData methods 3. Added DataReady event
05	11-Aug-08	1. Add LabVIEW 7.0 and 8.5 to list of IDE's that this ActiveX has been tested with
04	21-May-08	1. Added explanation of HS command 2. Added pseudo-code for Write Read protocol 3. Minor fix to Registration instructions
03	27-Jan-08	1. Added table of methods and which heads they apply to
02	16-Jan-08	1. Added support for Thermopile and Photodiode heads in USBI 2.30 2. Changed name of device from USBI-4 to Pulsar-4 3. Updated name of INF file 4. Added list of commands that can be applied to Thermopile and Photodiode heads
01		1. Initial Revision

Overview

This document is divided into four sections

- 1) [Overview](#). Introduction and general explanation how to communicate with Ophir Pulsar-4, Pulsar-2, and Pulsar-1 devices using the OphirFastX ActiveX control (this section)
- 2) [ActiveX: Detailed Description](#). Detailed description of the methods, events, and error codes of the OphirFastX ActiveX control.
- 3) [External Trigger Settings](#)
- 4) [Measurement Pseudo Code](#)
- 5) [Appendix A: OphirFastXBeta](#)

Introduction

This document describes Ophir's ActiveX support of the Pulsar-4, Pulsar-2, and Pulsar-1 devices. Please see the companion document "USBI ActiveX Control" for details on the ActiveX support for the USBI and Nova-II devices.

Besides their use as standalone, fully featured laser power/energy meters, the Ophir Pulsar-4, Pulsar-2, and Pulsar-1 devices can also be used through an ActiveX control. This allows system integrators to integrate the measurement capabilities of the Pulsar devices with legacy analysis packages.

The OphirFastX ActiveX control has been tested in VB.NET 2005, VC#.NET 2005, and in Excel (Microsoft). It has also been tested in LabVIEW 7.0 and 8.5. Demo client applications are provided with the StarLab installation. In practice, OphirFastX can be used in any environment that allows interaction with COM automation servers (although it hasn't been tested with tools other than those mentioned).

Note: As part of the 2.10 release of StarLab, Ophir now provides a COM object interface (OphirLMMeasurement) that supports all Ophir USB-speaking devices. The ActiveX packages are included in the release so as to not disrupt legacy OEM installations by customers. For new designs, we highly recommend using OphirLMMeasurement.

Registration

StarLab PC software (1.10 and higher) registers OphirFastX as part of the installation process of the StarLab application. Registration is also possible via the Options Menu of the StarLab application.

After installing the StarLab PC application on one PC, it's possible to register the ActiveX control on additional PC's without installing the full StarLab package. To do so, use the following steps

1. Copy the following files to C:\ophirx (all the files are in the installation directory C:\program files\ophir optronics\StarLab 1.10)
 - OphirFastX.ocx
 - wdreg.exe
 - ophdev.inf
 - ophdev.cat
 - pulsar.inf
 - pulsar.cat
 - windrvr6.inf
 - windrvr6.sys
 - wd901.cat
 - dixfapi.dll
 - FU4Axxx.hex
 - FU4Bxxx.hex
 - FU4Fxxx.ttf
2. Build a batch file with the following contents in C:\ophirx (rem means remark)
 - rem 1) install jungo driver (our 3rd party tool)
 - wdreg -inf windrvr6.inf install
 - rem 2) install ActiveX (this is Ophir specific)
 - regsvr32 OphirFastX.ocx
 - rem 3) install pulsar driver (Ophir specific INF file)
 - wdreg -inf pulsar.inf install
3. Copy C:\ophirx to your target PC and run the batch file

Examples

Examples of ActiveX containers using OphirFastX in VB.NET 2005, VC#.NET 2005, and Excel are provided with the installation package. All examples assume a rudimentary knowledge of the respective development platforms as well as an understanding of ActiveX controls.

For the fast head (Pyroelectric and nanoJoule) examples, look in C:\Program Files\Ophir Optronics\StarLab 1.10\Automation Examples\OphirFastX Pulsar demos\Pyroelectric and Nanojmeter

For the slow head (Thermopile and Photodiode) examples, look in C:\Program Files\Ophir Optronics\StarLab 1.10\Automation Examples\OphirFastX Pulsar demos\Thermopile and Photodiode

USB Details

The Ophir Pulsar-4, Pulsar-2, and Pulsar-1 devices are high-speed, self-powered, high-powered USB devices. Besides a control endpoint (over which most of the communication takes place), it also has one interrupt IN endpoint. It's used by the [StartCS](#), [StartCS2](#), and [StartTurboMode](#) methods (described later) to allow reporting of measurements with less USB protocol overhead and at a higher data rates.

Device Communication Details

In order to take full advantage of their high-speed capabilities, the Pulsar-4, Pulsar-2, and Pulsar-1 devices report raw, binary data to the PC. The methods provided by OphirFastX allow full control of the head(s) attached to the device(s) as well as formatting of the data throughput into a form that is Client-Application friendly.

For the slow heads, Pulsar devices report the data in ASCII strings, closely matching the standard of other Ophir devices. For these heads too, Ophir supplies data-formatting methods similar to those already in use for the faster heads. OphirFastX also allows direct communication between the application and the Pulsar devices using the Write and Read methods (although this methodology is not recommended)

ActiveX: Detailed Description

This section describes the methods, events, and error codes of the OphirFastX ActiveX control.

Note: Only once instance of OphirFastX can be active. If a second instance is opened, the [OpenUSB](#) method will return a warning code, reminding the Application Programmer of this limitation.

Error Codes

- 1) 0x00000000: No Error.
- 2) 0x80040100: OphirFastX has not been opened.
- 3) 0x80040101: OpenWarning
- 4) 0x80040102: OphirFastX has already been opened.
- 5) 0x80040103: OphirFastX drivers cannot be loaded.
- 6) 0x80040104: Load File Missing
- 7) 0x80040105: Load Device Failed
- 8) 0x80040106: Failed to detect devices.
- 9) 0x80040107: Failed to refresh device(s).
- 10) 0x80040200: Device index out of range.
- 11) 0x80040201: Invalid device handle.
- 12) 0x80040210: Invalid head channel.
- 13) 0x80040211: Head not supported
- 14) 0x80040212: No head is connected to channel.
- 15) 0x80040300: Save to head failed.
- 16) 0x80040301: Param Error
- 17) 0x80040302: Failed to create Safe Array
- 18) 0x80040303: Not Applicable in this head.
- 19) 0x80040310: Not a Continuous Spectrum Head
- 20) 0x80040311: Not a Discrete Spectrum Head
- 21) 0x80040312: Wavelength Out of Range
- 22) 0x80040320: No Diffuser
- 23) 0x80040321 No Filter
- 24) 0x80040322 Command Failed
- 25) 0x80040400: Stream Mode not started.
- 26) 0x80040401: Stream Mode already started.
- 27) 0x80040402: At least one device is in Stream Mode.
- 28) 0x80040410: Turbo not supported
- 29) 0x80040411: Turbo not configured
- 30) 0x80040412: Expected Frequency out of range
- 31) 0x80040413: Expected Number of pulses out of range
- 32) 0x80040414: No turbo data
- 33) 0x80040415: Packet number error
- 34) 0x80040416: Not in turbo mode
- 35) 0x80040417: Requested more data than can be handled

Methods

The methods of OphirFastX are divided into several categories

- USB Device Communications

CloseUSB	DetectDevices	GetDeviceHandle
GetNumberOfDevices	OpenUSB	RefreshAllDevices

- General Information and Diagnostics

GetChannelInfo	GetDeviceInfo	GetDriverVersion
GetErrorFromCode	IsChannelExists	SetDiagnosticMode

- Head Configuration

DeleteWavelength	GetContinuousWavelengths	GetDiffuser
GetDiscreteWavelengths	GetFilter	GetMeasurementMode
GetPulseLengths	GetRanges	GetThreshold
GetWavelengthsType	ModifyWavelength	SaveSettings
SetDiffuser	SetFilter	SetMeasurementMode
SetPulseLength	SetRange	SetThreshold
SetWavelengthIndex		

- Trigger Settings

GetExtTrigModes	GetExtTrigOnOff	GetExtTrigWindowTime
SetExtTrigMode	SetExtTrigOnOff	SetExtTrigWindowTime

- Measurement Delivery

ConfigChannelForTurbo	EnableDisableChannelForCS	GetData
GetMaxFrequency	GetTurboNumberOfPackets	GetTurboPacket
StartCS	StartCS2	StartTurboMode
StopAllCS	StopCS	StopTurboMode

- Legacy Methods

Read	Write	
----------------------	-----------------------	--

Alphabetic Listing - Summary

The following table is the list of all methods and to which heads they apply.

Method	Device and Driver	Thermopile	Photodiode	Pyroelectric and nanoJoule
CloseUSB	•			
ConfigChannelForTurbo				•
DeleteWavelength		•	•	•
DetectDevices	•			
EnableDisableChannelForCS				
GetChannelInfo		•	•	•
GetContinuousWavelengths		•	•	•
GetData		•	•	•
GetDeviceHandle	•			
GetDeviceInfo	•			
GetDiffuser				•
GetDiscreteWavelengths		•		•
GetDriverVersion	•			
GetExtTrigModes				•
GetExtTrigOnOff				•
GetExtTrigWindowTime				•
GetErrorFromCode	•			
GetFilter			•	
GetMaxFrequency				•
GetMeasurementMode		•	•	•
GetNumberOfDevices	•			
GetPulseLengths				•
GetRanges		•	•	•
GetThreshold		•		
GetTurboNumberOfPackets				•
GetTurboPacket				•
GetWavelengthsType		•	•	•
IsChannelExists	•			
ModifyWavelength		•	•	•
OpenUSB	•			
Read		•	•	
RefreshAllDevices	•			
SaveSettings	•	•	•	•
SetDiagnosticMode	•			
SetDiffuser				•
SetExtTrigMode				•
SetExtTrigOnOff				•
SetExtTrigWindowTime				•
SetFilter			•	
SetMeasurementMode		•	•	•
SetPulseLength				•
SetRange		•	•	•
SetThreshold		•		
SetWavelengthIndex		•	•	•
StartCS		•	•	•
StartCS2		•	•	•
StartTurboMode				•
StopAllCS		•	•	•
StopCS		•	•	•
StopTurboMode				•
Write		•	•	

Alphabetic Listing - Details

The following is an alphabetical listing of the methods of OphirFastX. Note, although Pulsar-4 is mentioned, these methods apply to all three devices (Pulsar-4, Pulsar-2, and Pulsar-1) unless otherwise noted.

Name	CloseUSB
Parameters	None
Use	Closes Ophir USB drivers and disables working with OphirFastX. This method is the last one to be called, and ends the communication session with the Pulsar device.
Return Codes	1) 0x000000: No Error 2) 0x80040100: OphirFastX has not been opened.
See Also:	DetectDevices , GetDeviceHandle , GetNumberOfDevices , OpenUSB , RefreshAllDevices ,

Name	ConfigChannelForTurbo
Parameters	1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) short nExpectedFreq 4) long lStopAfterNumber 5) short nEnable
Use	This must be called for each channel of the device before StartTurboMode nExpectedFreq is the expected pulse frequency of the laser to be measured on that channel lStopAfterNumber is the desired number of pulses on that channel nEnable for setting this specific channel to report Turbo Mode readings or not
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040410: Turbo not supported. 7) 0x80040412: Expected Frequency out of range. 8) 0x80040413: Expected Number of pulses out of range.
See Also:	GetMaxFrequency , GetTurboNumberOfPackets , GetTurboPacket , StartTurboMode , StopTurboMode , EventTurboCompleted , Working with Turbo Mode

Name	DeleteWavelength
Parameters	1) short nDeviceHandle 2) SHORT nHeadChannel 3) SHORT nWavelengthIndex
Use	Delete wavelength specified in nWavelengthIndex. Note: Do not delete the currently selected wavelength.
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel.

	6) 0x80040322: Command Failed.
See Also:	GetContinuousWavelengths , GetDiscreteWavelengths , GetWavelengthsType , ModifyWavelength , SaveSettings , SetWavelengthIndex

Name	DetectDevices
Parameters	None
Use	Detect if any Ophir Pulsar-4 devices are attached to the PC.
Return Codes	<ul style="list-style-type: none"> 1) 0x00000000: No Error 2) 0x80040100: OphirFastX has not been opened 3) 0x80040106: Failed to detect any devices. 4) 0x80040402: At least one device is in Stream Mode.
See Also:	CloseUSB , GetDeviceHandle , GetNumberOfDevices , OpenUSB , RefreshAllDevices ,

Name	EnableDisableChannelForCS
Parameters	<ul style="list-style-type: none"> 1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) short nEnable
Use	Enable or Disable channel for sending data when the StartCS or StartCS2 methods are called.
Return Codes	<ul style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040301: Param Error
See Also:	StartCS , StartCS2 , StopAllCS , StopCS , MeasureArrived , Working with CS

Name	GetChannelInfo
Parameters	<ul style="list-style-type: none"> 1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) long* lSerialNum (Out) 4) VARIANT* varIDString (Out) 5) VARIANT* varNameString (Out)
Use	Gets information about head attached to specific channel in a device. Note: varIDString, varNameString are VARIANT pointers to a BSTR. This was done in order to make the strings easily accessible in all languages.
Return Codes	<ul style="list-style-type: none"> 1) 0x00000000: No Error 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel.
See Also:	GetDeviceInfo , IsChannelExists

Name	GetContinuousWavelengths
Parameters	<ul style="list-style-type: none"> 1) short nDeviceHandle 2) short nHeadChannel (0..3)

	<ul style="list-style-type: none"> 3) short* nMinWavelength (Out) 4) short* nMaxWavelength (Out) 5) short* nWavelengthIndex (Out) 6) VARIANT* varWavelengthsStrings (Out)
Use	<p>Gets all wavelength information of the head attached to the selected channel in a specific device (for heads with a continuous spectrum)</p> <p>The currently selected wavelength is returned in nWavelengthIndex.</p> <p>The range of allowable wavelengths is from nMinWavelength to nMaxWavelength.</p> <p>varWavelengthsStrings contains string representations of the presently configured favorite wavelengths (up to 6)</p> <p>Note: varWavelengthsStrings is VARIANT pointer to an array of BSTR. This was done in order to make the strings easily accessible in all languages.</p>
Return Codes	<ul style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040310: Not a Continuous Spectrum Head 7) 0x80040302: Failed to create Safe Array (for varWavelengthsStrings). 8) 0x80040322: Command Failed
See Also:	DeleteWavelength , GetDiscreteWavelengths , GetWavelengthsType , ModifyWavelength , SaveSettings , SetWavelengthIndex

Name	GetData
Parameters	1) VARIANT* parray (Out)
Use	<p>This method uploads the data payload that the Pulsar device delivered to the ActiveX. It should be called after the DataReady event has been received.</p> <p>If called when no new data is available, then parray will be empty</p>
Return Codes	1) 0x00000000: No Error.
See Also:	DataReady , EnableDisableChannelForCS , StartCS2 , StopAllCS , StopCS , Working with CS2

Name	GetDeviceHandle
Parameters	<ul style="list-style-type: none"> 1) short nDeviceIndex 2) short *nDeviceHandle (Out)
Use	<p>Having called the GetNumberOfDevices method, this method is used to get a Device Handle for every Pulsar-4 device that the Application Programmer wants to work with.</p> <p>OphirFastX maintains a zero-based array of indexes. Therefore nDeviceIndex must be a value less than what is returned by GetNumberOfDevices.</p>
Return Codes	<ul style="list-style-type: none"> 1) 0x00000000: No Error 2) 0x80040100: OphirFastX has not been opened 3) 0x80040200: Device index out of range.
See Also:	CloseUSB , DetectDevices , GetNumberOfDevices , OpenUSB , RefreshAllDevices ,

Name	GetDeviceInfo
Parameters	<ol style="list-style-type: none"> 1) short nDeviceHandle 2) long* ISerialNum (Out) 3) VARIANT* varNameString (Out)
Use	<p>Gets information about the device.</p> <p>Note: varNameString is VARIANT pointer to a BSTR. This was done in order to make the strings easily accessible in all languages.</p>
Return Codes	<ol style="list-style-type: none"> 1) 0x00000000: No Error 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle.
See Also:	GetChannelInfo , IsChannelExists

Name	GetDiffuser
Parameters	<ol style="list-style-type: none"> 1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) short *nDiffuserExist (Out) 4) short *nDiffuserState (Out)
Use	To query if head in selected channel has a Diffuser (1 for yes, 0 for no) and if yes what its present state is (1 for on 0 for Off).
Return Codes	<ol style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel.
See Also:	SaveSettings , SetDiffuser

Name	GetDiscreteWavelengths
Parameters	<ol style="list-style-type: none"> 1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) short * nWavelengthIndex (Out) 4) VARIANT* varWavelengthsStrings (Out)
Use	<p>Gets all wavelength information of the head attached to the selected channel in a specific device (for heads with a discrete spectrum)</p> <p>The currently selected wavelength is returned in nWavelengthIndex.</p> <p>varWavelengthsStrings contains string representations of the selectable wavelengths (up to 5)</p> <p>Note: varWavelengthsStrings is VARIANT pointer to an array of BSTR. This was done in order to make the strings easily accessible in all languages.</p>
Return Codes	<ol style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040311: Not a Discrete Spectrum Head 7) 0x80040302: Failed to create Safe Array (for varWavelengthsStrings). 8) 0x80040322: Command Failed

See Also:	DeleteWavelength , GetContinuousWavelengths , GetWavelengthsType , ModifyWavelength , SaveSettings , SetWavelengthIndex
-----------	---

Name	GetDriverVersion
Parameters	1) short *nDriverVersion (Out)
Use	Used for debugging purposes. If there is an error when using OphirFastX, use this method to get the version of Ophir Pulsar-4 drivers installed on this PC (returned in the <i>nDriverVersion</i> parameter).
Return Codes	1) 0x00000000: No Error 2) 0x80040100: OphirFastX has not been opened
See Also:	GetErrorFromCode , SetDiagnosticMode

Name	GetErrorFromCode
Parameters	1) long lErrorCode 2) VARIANT *varErrorString (Out)
Use	Given an error code, will pass back a string explanation of the error. Note: <i>varErrorString</i> is VARIANT pointer to a BSTR. This was done in order to make the string easily accessible in all languages.
Return Codes	1) 0x00000000: No Error
See Also:	GetDriverVersion , SetDiagnosticMode

Name	GetExtTrigModes
Parameters	1) short nDeviceHandle 2) short * nExtTrigModelIndex (Out) 3) VARIANT* varExtTrigModesStrings (Out)
Use	To get the set of external trigger modes of a specific device. See Section External Trigger Modes for a full description of the various modes
Return Codes	1) 0x00000000. No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040302: Failed to create Safe Array (for varExtTrigModesStrings).
See Also:	GetExtTrigOnOff , GetExtTrigWindowTime , SetExtTrigMode , SetExtTrigOnOff , SetExtTrigWindowTime

Name	GetExtTrigOnOff
Parameters	1) short nDeviceHandle 2) short nHeadChannel 3) short* nOnOff (Out)
Use	To get the external trigger state, of a specific channel in a specific device: On (1) or Off (0).
Return Codes	1) 0x00000000: No Error.

	<ul style="list-style-type: none"> 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel.
See Also:	GetExtTrigModes , GetExtTrigWindowTime , SetExtTrigMode , SetExtTrigOnOff , SetExtTrigWindowTime

Name	GetExtTrigWindowTime
Parameters	<ul style="list-style-type: none"> 1) short nDeviceHandle 2) short* nExtTrigWindow (Out)
Use	<p>To get the external trigger window time, of a specific device. Window time can be between 1 and 65535 uS.</p> <p>See Section External Trigger Window for a full explanation of the use of this window of time</p>
Return Codes	<ul style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle.
See Also:	GetExtTrigModes , GetExtTrigOnOff , SetExtTrigMode , SetExtTrigOnOff , SetExtTrigWindowTime

Name	GetFilter
Parameters	<ul style="list-style-type: none"> 1) short nDeviceHandle 2) SHORT nHeadChannel (0..3) 3) SHORT* nFilterExist (0..1) 4) SHORT* nFilterState (0..1)
Use	<p>To query if the head has more than one Filter State and if yes, what is the present Filter State</p> <p>nFilterExist 0 or 1(exists) nFilterState: 0:Out 1:In</p>
Return Codes	<ul style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040303: Not Applicable in this head.
See Also:	SetFilter , SaveSettings

Name	GetMaxFrequency
Parameters	<ul style="list-style-type: none"> 1) short nDeviceHandle 2) short nHeadChannel 3) short * nMaxFrequency (Out)
Use	<p>To get the max frequency that can be measured by a specific head. This should be called before ConfigChannelForTurbo to ensure that the channel is not configured for a frequency higher than the head that is attached to it is capable of measuring</p>
Return Codes	<ul style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel.

	5) 0x80040212: No head is connected to channel.
See Also:	ConfigChannelForTurbo , GetTurboNumberOfPackets , GetTurboPacket , StartTurboMode , StopTurboMode , TurboCompleted , Working with Turbo Mode

Name	GetMeasurementMode
Parameters	1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) short *nMode (Out)
Use	To get the measurement mode: energy (0) or power (1).
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel.
See Also	SaveSettings , SetMeasurementMode

Name	GetNumberOfDevices
Parameters	1) short *nDeviceNumber (Out)
Use	Called after the OpenUSB method to get count (in nDeviceNumber) of Ophir Pulsar-4 devices attached to the PC.
Return Codes	1) 0x00000000: No Error 2) 0x80040100: OphirFastX has not been opened
See Also:	CloseUSB , DetectDevices , GetDeviceHandle , OpenUSB , RefreshAllDevices

Name	GetPulseLengths
Parameters	1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) short* nPulseLengthIndex (Out) 4) VARIANT* varPulseLengthsStrings (Out)
Use	To get Pulse Length information of the head in a specific channel in a specific device. The currently selected pulse length is returned in nPulseLengthIndex (0 for short pulse mode, 1 for long pulse mode). The first string returned is the maximum pulse length (in time) when in short pulse mode. The second string returned is the maximum pulse length when in long pulse mode Not all heads support short and long pulse modes. When one of the settings s unavailable the relevant string will be "N/A". Note: varPulseLengthsStrings is VARIANT pointer to an array of BSTR. This was done in order to make the strings easily accessible in all languages.
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel.

	<ul style="list-style-type: none"> 5) 0x80040212: No head is connected to channel. 6) 0x80040302: Failed to create Safe Array (for varPulseLengthsStrings).
See Also:	GetPulseLengths , SaveSettings , SetPulseLength ,

Name	GetRanges
Parameters	<ul style="list-style-type: none"> 1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) short *nRangeIndex (Out) 4) VARIANT *varRangesStrings (Out)
Use	<p>To get all range information of the head in a specific channel in a specific device.</p> <p>The currently selected range is returned in nRangeIndex.</p> <p>varRangeStrings contains string representations of all the available ranges for this head</p> <p>Note: varRangesStrings is VARIANT pointer to an array of BSTR. This was done in order to make the strings easily accessible in all languages.</p>
Return Codes	<ul style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040302: Failed to create Safe Array (for varRangesStrings). 7) 0x80040322: Command Failed
See Also:	SaveSettings , SetRange

Name	GetThreshold
Parameters	<ul style="list-style-type: none"> 1) short nDeviceHandle 2) SHORT nHeadChannel (0..3) 3) SHORT* nThresholdIndex (0..2) 4) VARIANT* varThresholdStrings
Use	<p>Get list of threshold settings available and the present setting. This is for Thermopile heads measuring energy</p> <p>The currently selected threshold setting is returned in nThresholdIndex</p> <p>The set of available thresholds is returned in varThresholdStrings. This will generally be "LOW MEDIUM HIGH"</p> <p>Note: varThresholdStrings is a VARIANT pointer to an array of BSTR. This was done in order to make the strings easily accessible in all languages.</p>
Return Codes	<ul style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040303: Not Applicable in this head. 7) 0x80040322: Command Failed. 8) 0x80040302: Failed to create Safe Array (for varThresholdStrings).
See Also:	SetThreshold , SaveSettings

Name	GetTurboNumberOfPackets
Parameters	<ol style="list-style-type: none"> 1) short nDeviceHandle 2) short nHeadChannel 3) long* INumber (Out)
Use	<p>Returns number of data packets that were received in Turbo Mode</p> <p>Must be called after turbo mode is finished.</p>
Return Codes	<ol style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040212: No head is connected to channel. 5) 0x80040210: Invalid head channel. 6) 0x80040414: No turbo data
See Also:	ConfigChannelForTurbo , GetMaxFrequency , GetTurboPacket , StartTurboMode , StopTurboMode , EventTurboCompleted , Working with Turbo Mode

Name	GetTurboPacket
Parameters	<ol style="list-style-type: none"> 1) short nDeviceHandle 2) short nHeadChannel 3) long IPacketNumber 4) VARIANT* parrMeasurement (Out)
Use	<p>Get packet of turbo data (first packet number is 0).</p> <p>Must be called after turbo mode is finished.</p> <p>Note: Call GetTurboNumberOfPackets to determine how many packets of data were received that need to be processed with the GetTurboPacket method</p>
Return Codes	<ol style="list-style-type: none"> 7) 0x00000000: No Error. 8) 0x80040100: OphirFastX has not been opened. 9) 0x80040201: Invalid device handle. 10) 0x80040212: No head is connected to channel. 11) 0x80040210: Invalid head channel. 12) 0x80040414: No turbo data 13) 0x80040415: Packet number error
See Also:	ConfigChannelForTurbo , GetMaxFrequency , GetTurboNumberOfPackets , StartTurboMode , StopTurboMode , EventTurboCompleted , Working with Turbo Mode

Name	GetWavelengthsType
Parameters	<ol style="list-style-type: none"> 1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) VARIANT * varWavelengthsType (Out)
Use	<p>To get type of wavelengths that the head is programmed for ("Continuous" or "Discrete").</p> <p>Note: varWavelengthsType is VARIANT pointer to a BSTR. This was done in order to make the strings easily accessible in all languages.</p>
Return Codes	<ol style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel.

	5) 0x80040212: No head is connected to channel. 6) 0x80040322: Command Failed
See Also:	DeleteWavelength , GetContinuousWavelengths , GetDiscreteWavelengths , ModifyWavelength , SaveSettings , SetWavelengthIndex

Name	IsChannelExists
Parameters	1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) short* nExist (Out)
Use	Check if head exists in the specific channel of the specific device.
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel.
See Also:	GetChannelInfo , GetDeviceInfo

Name	ModifyWavelength
Parameters	1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) short nWavelengthIndex 4) short nNewWavelength
Use	To change the setting of one of the (up to 6) favorite wavelength of the head specific channel in a specific device. Use this to define a new wavelength, modify an already defined wavelength. To delete a wavelength that is no longer necessary, use the DeleteWavelength method Note: For heads with a Continuous Spectrum only
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040310: Not a Continuous Spectrum Head 7) 0x80040312: Wavelength Out of Range. 8) 0x80040301: Param Error (Laser index). 9) 0x80040322: Command Failed
See Also:	DeleteWavelength , GetContinuousWavelengths , GetDiscreteWavelengths , GetWavelengthsType , SaveSettings , SetWavelengthIndex

Name	OpenUSB
Parameters	1) long *IWarning (Out)
Use	Initialize OphirFastX for use. All other methods should be called only after OpenUSB has been called. If OphirFastX has been loaded by a different session OpenUSB will return 0x80040101: Open Warning. This means that it may be possible to communicate with the devices but if not, it's because the other session has loaded and also locked out OphirFastX from use by others

	Note: OpenUSB loads the Pulsar-4 device with its firmware. The firmware files must be in the same directory as the ActiveX. As of version 1.10 the files to use are FU4A125.hex , FU4B125.hex , and FU4F118.ttf . The files to use are always to be found in the StarLab application's installation directory
Return Codes	<ol style="list-style-type: none"> 1) 0x00000000: No Error 2) 0x80040100: Failed to open OphirFastX. 3) 0x80040102: OphirFastX already opened. 4) 0x80040103: OphirFastX drivers cannot be loaded. 5) 0x80040104: Load File Missing 6) 0x80040105: Load Device Failed
See Also:	CloseUSB , DetectDevices , GetDeviceHandle , GetNumberOfDevices , RefreshAllDevices ,

Name	RefreshAllDevices
Parameters	None
Use	<p>Restarts all detected Ophir Pulsar-4 devices. Used after changing at least one head in one channel of one Pulsar-4 device</p> <p>Note: This command method will not function unless no devices are streaming data on the USB (i.e. StopCS, StopAllCS, or StopTurboMode was not called since putting at least one device into Stream mode with the StartCS and StartTurboMode methods)</p>
Return Codes	<ol style="list-style-type: none"> 1) 0x00000000: No Error 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040107: Failed to refresh device(s). 4) 0x80040402: At least one device is in Stream Mode.
See Also:	CloseUSB , DetectDevices , GetDeviceHandle , GetNumberOfDevices , OpenUSB

Name	SaveSettings
Parameters	<ol style="list-style-type: none"> 1) short nDeviceHandle 2) short nHeadChannel (0..3)
Use	To save all configuration settings for the head in a specific channel in a specific device.
Return Codes	<ol style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040300: Save to head failed.
See Also:	DeleteWavelength , GetDiffuser , GetDiscreteWavelengths , GetFilter , GetMeasurementMode , GetPulseLengths , GetRanges , GetThreshold , GetWavelengthsType , ModifyWavelength , SetDiffuser , SetFilter , SetMeasurementMode , SetPulseLength , SetRange , SetThreshold , SetWavelengthIndex

Name	SetDiagnosticMode
Parameters	None
Use	Put ActiveX in diagnostics mode. ActiveX will print a buffer file that contains

	trace of the Pulsar-4 device's upload process
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened.
See Also:	GetDriverVersion , GetErrorFromCode

Name	SetDiffuser
Parameters	1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) short nDiffuserState (0..1)
Use	To set the head's Diffuser state to On (1) or Off (0).
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040320: No Diffuser. 7) 0x80040301: Param Error.
See Also:	GetDiffuser , SaveSettings

Name	SetExtTrigMode
Parameters	1) short nDeviceHandle 2) short nExtTrigModeIndex
Use	To Set the external trigger mode, of a specific device. See Section External Trigger Modes for a full description of the various modes
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040301: Param Error.
See Also:	GetExtTrigModes , GetExtTrigOnOff , GetExtTrigWindowTime , SetExtTrigOnOff , SetExtTrigWindowTime

Name	SetExtTrigOnOff
Parameters	1) short nDeviceHandle 2) short nHeadChannel 3) short nOnOff
Use	To set the external trigger state, of a specific channel in a specific device: On (1) or Off (0).
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040301: Param Error.
See Also:	GetExtTrigModes , GetExtTrigOnOff , GetExtTrigWindowTime , SetExtTrigMode , SetExtTrigWindowTime

Name	SetExtTrigWindowTime
Parameters	1) short nDeviceHandle 2) short nExtTrigWindow
Use	To Set the external trigger window time, of a specific device. Window time can be between 1 and 65535 uS. See Section External Trigger Window for a full explanation of the use of this window of time
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040301: Param Error.
See Also:	GetExtTrigModes , GetExtTrigOnOff , GetExtTrigWindowTime , SetExtTrigMode , SetExtTrigOnOff

Name	SetFilter
Parameters	1) short nDeviceHandle 2) SHORT nHeadChannel (0..3) 3) SHORT nFilterState (0..1)
Use	To set the head's Filter State to Out (0) or In (1).
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040303: Not Applicable in this head. 7) 0x80040301: Param Error 8) 0x80040321 No Filter
See Also:	GetFilter , SaveSettings

Name	SetMeasurementMode
Parameters	1) short nDeviceHandle 2) short nHeadChannel 3) short nMode
Use	To set the measurement mode to: energy (0) or power (1).
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040301: Param Error.
See Also	GetMeasurementMode , SaveSettings

Name	SetPulseLength
Parameters	1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) short nPulseLengthIndex
Use	To set a pulse length setting of the head a specific channel in a specific device. Set to 0 for short pulse mode and 1 for long pulse mode.

Return Codes	<ol style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040301: Param Error.
See Also:	GetPulseLengths , SaveSettings

Name	SetRange
Parameters	<ol style="list-style-type: none"> 1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) short nRangeIndex
Use	To set the range for the head in a specific channel in a specific device.
Return Codes	<ol style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040301: Param Error (Range index). 7) 0x80040322: Command Failed
See Also:	GetRanges , SaveSettings

Name	SetThreshold
Parameters	<ol style="list-style-type: none"> 1) short nDeviceHandle 2) SHORT nHeadChannel (0..3) 3) SHORT nThresholdIndex (0..2)
Use	To set the threshold for Thermopile heads measuring energy 0: LOW 1: MEDIUM 2: HIGH
Return Codes	<ol style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040303: Not Applicable in this head. 7) 0x80040322: Command Failed. 8) 0x80040301: Param Error
See Also:	GetThreshold , SaveSettings

Name	SetWavelengthIndex
Parameters	<ol style="list-style-type: none"> 1) short nDeviceHandle 2) short nHeadChannel (0..3) 3) short nWavelengthIndex
Use	To set a Wavelength of the head in a specific channel in a specific device.
Return Codes	<ol style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040210: Invalid head channel. 5) 0x80040212: No head is connected to channel. 6) 0x80040301: Param Error (Laser index).

See Also:	DeleteWavelength , GetContinuousWavelengths , GetDiscreteWavelengths , GetWavelengthsType , ModifyWavelength , SaveSettings
-----------	---

Name	StartCS
Parameters	1 short nDeviceHandle
Use	Put device selected by nDeviceHandle into Stream Mode. In this mode, measurements are reported on the Pulsar-4's Interrupt IN endpoint. A detailed description of the format of measurements reported is described in Stream Mode Formats . This method sets the ActiveX to report all the measurements together with the MeasureArrived event.
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040400: Failed to start Stream Mode. 5) 0x80040401: Stream Mode already started.
See Also:	EnableDisableChannelForCS , StopAllCS , StopCS , MeasureArrived , Working with CS

Name	StartCS2
Parameters	1 short nDeviceHandle
Use	Put device selected by nDeviceHandle into Stream Mode. This method sets the ActiveX to inform the client application of data delivery through firing the DataReady event but to not deliver data until the client requests it through the GetData method In this mode, measurements are reported on the Pulsar-4's Interrupt IN endpoint. A detailed description of the format of measurements reported is described in Stream Mode Formats .
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040400: Failed to start Stream Mode. 5) 0x80040401: Stream Mode already started.
See Also:	DataReady , EnableDisableChannelForCS , GetData , StopAllCS , StopCS , Working with CS2

Name	StartTurboMode
Parameters	1) short nDeviceHandle 2) long *IWarning (Out)
Use	Start turbo mode for the device. Starting turbo mode will succeed only when at least one channel is configured for turbo mode with ConfigChannelForTurbo. Turbo Mode will only gather pulses up to the maximum it can handle. If the client application set the ActiveX to gather more pulses than it can handle (as set in calls to ConfigChannelForTurbo), then StartTurboMode will set IWarning to 0x80040417: Requested more data than can be handled. When turbo mode is completed, the TurboCompleted event will be sent by the ActiveX.
Return Codes	1) 0x00000000: No Error.

	<ul style="list-style-type: none"> 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040401: Stream Mode already started. 5) 0x80040411: Turbo not configured. 6) 0x80040400: Failed to start Stream Mode.
See Also:	ConfigChannelForTurbo , GetMaxFrequency , GetTurboNumberOfPackets , GetTurboPacket , StopTurboMode , EventTurboCompleted , Working with Turbo Mode

Name	StopAllCS
Parameters	None
Use	Take all devices out of Stream Mode reporting of measurements (started with the StartCS method)
Return Codes	<ul style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened.
See Also:	EnableDisableChannelForCS , StartCS , StartCS2 . StopAllCS , StopCS , MeasureArrived , Working with CS

Name	StopCS
Parameters	1) short nDeviceHandle
Use	Take selected device out of Stream Mode that was started by the StartCS method.
Return Codes	<ul style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040400: Selected Device not in Stream Mode.
See Also:	EnableDisableChannelForCS , StartCS , StartCS2 , StopAllCS , MeasureArrived , Working with CS

Name	StopTurboMode
Parameters	1) short nDeviceHandle
Use	<p>Stop the Turbo mode manually.</p> <p>The EventTurboCompleted will be sent to signal that Turbo Mode has ended and collected data is ready.</p>
Return Codes	<ul style="list-style-type: none"> 1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040401: Stream Mode already started. 5) 0x80040416: Not in turbo mode
See Also:	ConfigChannelForTurbo , GetMaxFrequency , GetTurboNumberOfPackets , GetTurboPacket , StartTurboMode , EventTurboCompleted , Working with Turbo Mode

Events

Name	DataReady
Parameters	None
Use	This event is fired when data is being reported in CS2 Stream Mode. It informs the client application that the Pulsar-4 device has delivered a new data payload. The ActiveX will not deliver the data to the client application until it calls the GetData method See Working with CS2 for a description of data delivered.
Return Codes	None
See Also:	EnableDisableChannelForCS , GetData , StartCS2 , StopAllCS , StopCS , Working with CS2

Name	MeasureArrived
Parameters	1) VARIANT *parrMeasurement (Out)
Use	This event is fired when data is being reported in CS Stream Mode. It informs the client application that the Pulsar-4 device has delivered a new data payload. It also delivers the data payload to the client application See Working with CS for a description of data delivered.
Return Codes	None
See Also:	EnableDisableChannelForCS , StartCS , StopAllCS , StopCS , Working with CS

Name	TurboCompleted
Parameters	1) short nDeviceHandle (Out)
Use	This event is fired when Turbo mode is completed for specified device. This event is also fired when Turbo mode is stopped manually by StopTurboMode See Working with Turbo Mode for a description of the data delivered
Return Codes	None
See Also:	ConfigChannelForTurbo , GetMaxFrequency , GetTurboNumberOfPackets , GetTurboPacket , StartTurboMode , StopTurboMode , Working with Turbo Mode

Legacy Methods

When working with Thermopile and Photodiode heads, Pulsar devices communicate with the PC in ASCII strings. OphirFastX allows client applications to communicate directly with the Pulsar device by using the Read and Write methods.

Name	Read
Parameters	1) short nDeviceHandle 2) const VARIANT* Answer
Use	Read response from Pulsar device associated with DeviceHandle. The response is in the form of an ASCII string. Note: Data is VARIANT pointer to a BSTR. This was done in order to make the string easily accessible in all languages.
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040322: Command Failed.
See Also:	Write

Name	Write
Parameters	1) short nDeviceHandle 2) const VARIANT frame
Use	Write command to selected Pulsar device.
Return Codes	1) 0x00000000: No Error. 2) 0x80040100: OphirFastX has not been opened. 3) 0x80040201: Invalid device handle. 4) 0x80040322: Command Failed.
See Also:	Read

Pulsar Commands

The following table has been taken from the companion document "*USBI ActiveX Control*" and reduced to Thermopile and Photodiode specific commands. For a full description of these commands, please look there

Command	Meaning	No Head	Thermopile	Photodiode
AR	All Ranges		•	•
AW	All Wavelengths		•	•
CQ	Calibration Query		•	•
EF	Energy Flag		•	
ER	Energy Ready		•	
ET	Energy Threshold		•	
FE	Force Energy		•	
FP	Force Power		•	•
FQ	Filter Query			•
HC	Head Configuration		•	•
HI	Head Information		•	•
IC	Instrument Configuration	•	•	•
II	Instrument Information	•	•	•
MA	MAins	•	•	•

RN	Read range Now		•	•
RQ	Response Query		•	
SE	Send Energy		•	
SI	Send unlts		•	•
SP	Send Power		•	•
VE	VErsion	•	•	•
WD	Wavelength adD		•	•
WE	Wavelength Erase		•	•
WI	Wavelength Index		•	•
WL	WaveLength		•	•
WN	Write range Now		•	•
ZA	Zero Abort	•	•	•
ZE	Zero	•	•	•
ZQ	Zero Query	•	•	•
ZS	Zero Save	•	•	•

Directing commands to specific heads

Unlike the USBI and Nova-II devices, the Pulsar can have up to 4 channels. Except for the MA and IC commands, it is necessary to specify upon which channel to apply the command. This can be done in one of two ways

Option 1: Use of CL command.

Command	CL (Channel select)
Syntax	CL <channel number, 1 to 4 or 0 for query>
Description	Query or set the communication channel of the Pulsar devices Values for <channel number> (if parameter isn't set, default to 0) <ul style="list-style-type: none"> • 0: Query Pulsar for present selected channel • 1: Configure communications for channel 1 • 2: Configure communications for channel 2 • 3: Configure communications for channel 3 • 4: Configure communications for channel 4
Example	Example 1. Pulsar already configured for channel 2 User sent " CL ". Pulsar returns " *2 ". Example 2. Pulsar already configured for channel 2 User sent " CL 0 ". Pulsar returns " *2 ". Example 3. Pulsar already configured for channel 2 and now changed to 3 User sent " CL 3 ". Pulsar returns "***" to signal that he succeeded. Example 4. Pulsar already configured for channel 2 and now changed to 5 User sent " CL 3 ". Pulsar returns " PARAM ERROR " to signal failure. Note: For Pulsar-2 the max channel number is 2. For Pulsar-1, the max channel number is 1
Limitations	None

Options 2: Extra parameter for other commands:

All other commands which refer to a head-specific or channel-specific action, take an extra optional parameter, the channel number 1 to 4. If this parameter is used, the Pulsar changes the active channel to this new channel, and then performs the action requested. If this parameter is NOT used, the Pulsar performs the action on the present active channel, whichever that is.

For example:

“**WN 0**” - changes to head range 0 (the top range) on the presently active channel.

“**WN 0 2**” - changes active channel to 2 and then changes head range for that channel to 0.

Note: The active channel will be permanently changed to this channel, for any subsequent commands.

Enable and Disable of Measurements

Although up to four heads can be attached to a Pulsar device, the user isn’t restricted to simultaneous measurement with each head. Rather the device can be instructed to enable or disable measurement on a per-channel basis.

Command	HS (Head Setting)
Syntax	HS <value 1 or 4, or 0 for Query > <channel number (optional) 1 to 4>
Description	<p>Sets the head measurement mode for the Pulsar device on selected channel</p> <p>Values allowed:</p> <ul style="list-style-type: none"> • 0: Query present setting • 1: Switch off measurements • 2: Switch on measurements for fast heads (pyroelectric, nanoJoule) • 4: Switch on measurements for slow heads (thermopile, photodiode) • Other values should not be used <p>Values for <channel number> (optional parameter)</p> <ul style="list-style-type: none"> • 1: head setting for channel 1 • 2: head setting for channel 2 • 3: head setting for channel 3 • 4: head setting for channel 4 • If omitted, sets head setting for presently active channel
Example	<p>Example 1. Pulsar head setting is off (value 1) on all channels. Present channel is 1 User sent “HS 4”. Pulsar returns “***”. Head measurements are switched on for thermopile/photodiode on channel 1</p> <p>Example 2. Same initial state as example 1. User sent “HS 4 2”. Pulsar returns “***”. Head measurements are switched on for thermopile/photodiode on channel 2; active channel is updated to channel 2.</p> <p>Note: For Pulsar-2 the max channel number is 2. For Pulsar-1, the max channel number is 1</p>
Limitations	None

External Trigger Settings

The Pulsar-4 device has a sophisticated External Trigger mechanism. This allows synchronization between pulse measurements and other inputs (or outputs) in the measurement environment.

Note: External Trigger functionality (including Missing Pulses) is available for Pyroelectric and nanoJoule heads only

Typical setup of the External Trigger methods is as follows

1. Set Trigger Mode
2. Set Window (for rising and falling edge modes)
3. Set External Trigger ON

External Trigger Modes

The Pulsar-4 External Trigger can be set to one of 6 modes (4 input modes, 2 output modes)

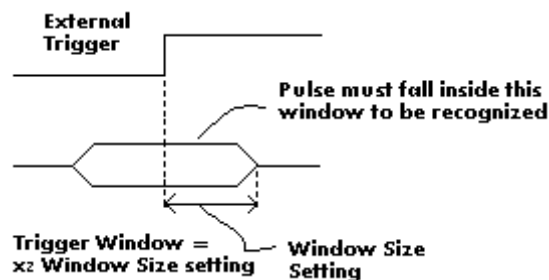
Input Modes	Description
Rising Edge	<ul style="list-style-type: none">• The device is sensitive to a trigger on the RISING EDGE of the input.• The trigger is valid for a pulse arriving during a window of time <i>before</i> or <i>after</i> the active (rising) edge• The inactive (falling) edge of the signal is ignored.• Missing Pulses are recorded when an External Trigger edge is received, but no pulse arrives within the Window Time before or after the active edge.• Pulses are ignored if they arrive outside the Window Time, before or after the active edge.
Falling Edge	<ul style="list-style-type: none">• The device is sensitive to a trigger on the FALLING EDGE of the input.• The trigger is valid for a pulse arriving during a window of time <i>before</i> or <i>after</i> the active (falling) edge• The inactive (rising) edge of the signal is ignored.• Missing Pulses are recorded when an External Trigger edge is received, but no pulse arrives within the Window Time before or after the active edge.• Pulses are ignored if they arrive outside the Window Time, before or after the active edge.
High Level	<ul style="list-style-type: none">• Pulses are recorded only when the input signal is at a HIGH LEVEL.• Any pulse arriving while the signal is high is counted. Any pulse arriving while the signal is low is ignored.• No Missing Pulses are recorded in this mode.

Input Modes	Description
Low Level	<ul style="list-style-type: none"> • Pulses are recorded only when the input signal is at a LOW LEVEL. • Any pulse arriving while the signal is low is counted. Any pulse arriving while the signal is high is ignored. • No Missing Pulses are recorded in this mode.

Output Modes	Description
Active High	<ul style="list-style-type: none"> • Every time a pulse arrives on the head detector the output goes high for 10us and then returns back to low. • The default level of the output with no pulses is low.
Active Low	<ul style="list-style-type: none"> • Every time a pulse arrives on the head detector the output goes low for 10us and then returns back to high. • The default level of the output with no pulses is high.

External Trigger Window

For trigger modes that define edge-triggered measurements (rising or falling), this defines the window of time before and after the edge that a pulse is considered to be in synch with the trigger. If there is no pulse during this window, then the Pulsar-4 device will report a MISSING MEASUREMENT



Pseudo code

The following are pseudo code examples of how to gather data with the Continuous Send and Turbo Mode methods of data delivery. For a live example, see the VB and VC# sample projects included in the StarLab installation directory.

The non-measurement methods are straightforward and can be understood directly from the sample code. Measurement methods involve working with events and need additional explanation

All Measurement Collection can be divided into three components

1. Setup and Start
2. Collection
3. Stop and Close Down

Every measurement delivered contains the following data (each is of type double)

- dTime: Time of measurement (in milliseconds)
- dValue: Measurement (in watts or joules)
- dStatus: One of
 - OK (0)
 - Overage (1)
 - Saturated (2)
 - Missing (3)

Continuous Send and Turbo Mode have slight differences which will be explained below

Working with CS

Most measurement needs are met with the Continuous Send family of methods. Note the MeasureArrived event carries with it the data payload from the Pulsar device. This causes some development environments to hang while processing the event. In such instances it is better to use the CS2 method described in the next section.

Setup and Start

Start of Function

```
/* The following opens the USB and gets the device handle */
```

```
OpenUSB
```

```
GetNumberOfDevices(num)
```

```
GetDeviceHandle(0, nHandle) ' get first device handle
```

```
/* Setup only channel one for measurement */
```

```
EnableDisableChannelForCS(nHandle, 0, 1) ' enable
```

```
EnableDisableChannelForCS(nHandle, 1, 0) ' disable
```

```
EnableDisableChannelForCS(nHandle, 2, 0) ' disable
```

```
EnableDisableChannelForCS(nHandle, 3, 0) ' disable
```

```
/* Start the measurement process */
```

```
StartCS(nHandle)
```

End of Function

MeasureArrived Handler

Start of Handler

```
/* In CS mode, data collection is triggered by the MeasureArrived event being sent to an event handler. The MeasureArrived event carries with it the data payload that was delivered by the Pulsar-4 device. Up to 8000 measurements are delivered with each event. For slow heads this will usually be no more than one or two measurements. */
```

```
/* Each measurement has the following format (all 5 elements are doubles) */
```

```
// dDeviceHandle
```

```
// dChannelIndex
```

```

// dTime:      In milliseconds
// dValue:     In watts or joules
// dStatus:
//           for fast heads will be one of {0: OK, 1: Overrange, 2: Saturated, 3: Missing}
//           for slow heads will be one of {0 = OK, 1 = Overrange,
11 = reset state in energy measurement,
12 = waiting state in energy measurement,
13= summing state in energy measurement,
14= timeout in energy measurement,
15= peak over in energy measurement,
16= energy over in energy measurement}

```

Next = 0

Repeat

```

Allocate (measurement structure)
dDeviceHandle = e.parrMeasurement(Next)
dChannelIndex = e.parrMeasurement(Next + 1)
dTime = e.parrMeasurement(Next + 2)
dValue = e.parrMeasurement(Next + 3)
dStatus = e.parrMeasurement(Next + 4)
Next +=5;

```

Until end of array is reached

End of Handler

Stop and Close Down

Start of Function

```

/* Stop gathering data */
StopCS(nHandle)
/* Release the ActiveX */
CloseUSB

```

End of Function

Working with CS2

Some environments (notably LabVIEW 8.x) hang while handling an event with a data payload. In such environments, it is better to use StartCS2. In this paradigm, the ActiveX fires the DataReady event without any data delivery. To gather the data, the client application calls the GetData method.

Setup and Start

Start of Function

```

/* The following opens the USB and gets the device handle */
OpenUSB
GetNumberOfDevices(num)
GetDeviceHandle(0, nHandle) ' get first device handle

/* Setup only channel one for measurement */
EnableDisableChannelForCS(nHandle, 0, 1) ' enable
EnableDisableChannelForCS(nHandle, 1, 0) ' disable
EnableDisableChannelForCS(nHandle, 2, 0) ' disable
EnableDisableChannelForCS(nHandle, 3, 0) ' disable

```

```

/* Start the measurement process */

```

```

StartCS2 (nHandle)

```

End of Function

DataReady Handler

Start of Handler

```

/* In CS2 mode, data collection is triggered by the DataReady event being sent to an
event handler. The event handler gathers the data payload that was delivered by the
Pulsar-4 device by calling the GetData method. Up to 8000 measurements are

```

delivered with each event. For slow heads this will usually be no more than one or two measurements. */

```
/* Each measurement has the following format (all 5 elements are doubles) */
// dDeviceHandle
// dChannelIndex
// dTime:      In milliseconds
// dValue:     In watts or joules
// dStatus:
//     for fast heads will be one of {0: OK, 1: Overrange, 2: Saturated, 3: Missing}
//     for slow heads will be one of {0 = OK, 1 = Overrange,
//                                     11 = reset state in energy measurement,
//                                     12 = waiting state in energy measurement,
//                                     13= summing state in energy measurement,
//                                     14= timeout in energy measurement,
//                                     15= peak over in energy measurement,
//                                     16= energy over in energy measurement}
```

GetData (parrMeasurement)

Next = 0

Repeat

 Allocate (measurement structure)

 dDeviceHandle = parrMeasurement(Next)

 dChannelIndex = parrMeasurement(Next + 1)

 dTime = parrMeasurement(Next + 2)

 dValue = parrMeasurement(Next + 3)

 dStatus = parrMeasurement(Next + 4)

 Next +=5;

 Until end of array is reached

End of Handler

Stop and Close Down

Start of Function

 /* Stop gathering data */

 StopCS(nHandle)

 /* Release the ActiveX */

 CloseUSB

End of Function

Working with Turbo Mode

Turbo Mode is for users that need to measure every pulse at high frequencies. With the Pulsar-4 it's possible to measure at 20KHz and not miss a pulse.

Unlike CS mode, The TurboCompleted event doesn't carry with it the data payload. When the event is triggered and it is up to the handler to gather the data if that is desired. The event handler selects the device and channel it wants data for. As such, OphirFastX will not return the device handle and channel number with the with each measurement in the data payload

Note: Turbo Mode is available for Pyroelectric and nanoJoule heads only

Setup and Start

Start of Function

 /* The following opens the USB and gets the device handle */

 OpenUSB

 GetNumberOfDevices(num)

 GetDeviceHandle(0, nHandle) ' get first device handle

```

/* Setup only channel zero for Turbo Mode measurements. The expected frequency
is 500 Hertz and the desired number of pulses is 5000 (10 seconds of data) */
ConfigChannelForTurbo (nHandle, 0, 500, 5000, 1)
ConfigChannelForTurbo (nHandle, 1, 0, 0, 0) // Disable channel 1
ConfigChannelForTurbo (nHandle, 2, 0, 0, 0) // Disable channel 2
ConfigChannelForTurbo (nHandle, 3, 0, 0, 0) // Disable channel 3

/* Start the measurement process */
StartTurboMode(nHandle)
End of Function

```

TurboCompleted Handler

Start of Handler

```

/* The TurboCompleted event is fired in one of two ways
    • The Pulsar-4 has delivered the pre-defined number of measurements
    • Client Application has called the StopTurboMode method

```

Upon receiving the TurboCompleted event, the Client Application must initiate the data transfer from the ActiveX to itself */

```

/* Get number of packets for first channel */
GetTurboNumberOfPackets(nHandle, 0, IPacketNumber)
/* Obtain the data from OphirFastX's buffer. Each packet contains up to 8000
measurements; with each measurement containing the time, value, and status of the
measurement as described above */

```

Packet Index = 0

Repeat

```

/* Each packet contains the time, value, and status of the measurement */
GetTurboPacket(nHandle, 0, Packet Index , arrDatas)
/* Parse the array to get the time, value, and status of each measurement */
Next = 0
Repeat

```

Allocate (turbo measurement structure)

dTime = arrDatas (Next)

dValue = arrDatas (Next + 1)

dStatus = arrDatas (Next + 2)

Next +=3;

Until end of arrDatas is reached

Increment Packet Index

Until Packet Index = IPacketNumber

End of Handler

Stop and Close Down

Start of Function

```

/* Stop gathering data. Note: this is only necessary for manual termination of Turbo
Mode. To allow Turbo Mode to terminate on its own, leave this line out */

```

StopTurboMode (nHandle)

```

/* Wait for TurboCompleted Handler that it has finished processing the data */

```

```

/* Release the ActiveX */

```

CloseUSB

End of Function

Working with Legacy Methods

The preferred communication protocol between the client application and the Pulsar device is the one described in Working with CS above. However, for customers already familiar with the command and response protocol supported by other devices, we include that option for Thermopile and Photodiode heads.

The command and response protocol is based on ASCII string flow between the PC application and the device. The ActiveX acts as a conduit between the two without any processing in either direction. It is therefore the client application's responsibility to format the string written to the device and parse the string returned appropriately.

Below is a sample session of the command and response flow between the Pulsar device and the PC application. It selects channel 1 on the Pulsar device for power measurement and gathers data until a pre-defined upper bound value has been measured. For simplicity's sake, error-checking is ignored.

Start of Function

```
/* The following opens the USB and gets the device handle */
OpenUSB
GetNumberOfDevices(num)
GetDeviceHandle(0, nHandle) ' get first device handle

/* Turn off CS streaming of measurements from the selected device */
StopCS(nHandle, 0, 1)

/* Set communication channel to 1 */
Write (nHandle, "CL 1");
Read (nHandle, ResponseString);

/* Put head in Power mode */
Write (nHandle, "FP");
Read (nHandle, ResponseString);

/* Start power measurement on selected channel */
Write (nHandle, "HS 4");
Read (nHandle, ResponseString);

/* Gather measurements until upper bound is reached */
Start_Loop
    Write (nHandle, "SP");
    Read (nHandle, ResponseString);
    Response = ConvertToReading (ResponseString)
    If Response > UPPER_LIMIT Then
        Break out of Loop
End_Loop

/* Release the ActiveX */
CloseUSB
```

End of Function

Appendix A: OphirFastXBeta

Several customers complained that OphirFastX could not be used in console applications. This is because the ActiveX fires an event that is based in a window. If the client application is "window-less", then it will not work well with the ActiveX. Also, MatLab based applications couldn't call in to the ActiveX.

OphirFastXBeta fixes these problems, as described below.

1. We have changed the way we deliver data to the calling application. Instead of sending the data together with the event we no longer fire an event. Rather, the application is expected to poll the ActiveX every 50mS for a new batch of data. This solves a problem with calling the ActiveX from within a console application.
 - Measurement: Use only **StartCS2** and **GetData**
 - The following methods and events are not supported in OphirFastXBeta
 - StartCS
 - StartTurboMode
 - ConfigChannelForTurbo
 - StopTurboMode
 - DataReady event
 - MeasureArrived event
 - TurboCompleted event
 - GetTurboPacket
 - GetTurboNumberOfPackets
 - Note: If you needed Turbo Mode performance, there is no longer a need for working in Turbo Mode with this version. Regular mode will also deliver all the data up to 20KHz.
2. To solve the problem of using our ActiveX in Matlab, we had to change the MFC interface of VARIANT * parameters in the following functions. The default interface is [in,out] whereas these parameters should really be [out]
 - GetRanges [out] VARIANT* varRangesStrings
 - GetErrorFromCode [out] VARIANT* varErrorString
 - GetContinuousWavelengths [out] VARIANT* varWavelengthsStrings
 - GetDiscreteWavelengths [out] VARIANT* varWavelengthsStrings
 - GetWavelengthsType [out] VARIANT* varWavelengthsType
 - GetPulseLengths [out] VARIANT* varPulseLengthsStrings
 - GetExtTrigModes [out] VARIANT* varExtTrigModesStrings
 - GetDeviceInfo [out] VARIANT* varNameString
 - GetChannelInfo [out] VARIANT* varIDString,[out] VARIANT* varNameString
 - GetTurboPacket [out] VARIANT* parrMeasurement
 - GetThreshold [out] VARIANT* varThresholdStrings
 - Read [out] VARIANT* Answer
 - GetData [out] VARIANT* parray
3. To distinguish between OphirFastX and OphirFastXBeta, please note the following.
 - It has a different GUID and name (OphirFastXBeta, OphirFastXBetaLib) in order to distinguish it from the officially released version
 - The version of the beta is 1.6
 - OphirFastXBeta.ocx must be registered with regsvr32 " OphirFastXBeta.ocx"
 - OphirFastXBeta.ocx and sample code will be found in the Beta subdirectory of the Automation Examples directory