OphirLMMeasurement COM Object

1 Revision History

14	18-Dec-24	1. Support for connecting to Ariel wirelessly
13	31-Jul-22	1. Mention Visual C++ redistributable
12	05-Jan-21	1. Support for Ariel 2. Support for VS 2019 3. Added GetData Status Codes: Temperature, Alert Hot, Pulse Width, and PfP Energy
11	25-Mar-20	1. Removed XP support 2. Added status to GetData method for reporting pulse frequency 3. Added information about the new interface `ICoLMMeasurement2` 4. Added functions for calibration date 5. Added functions for pulsed power pulse width 6. Added functions for low frequency power pulse frequency
10	11-Apr-19	1. Added Centauri and Juno+ 2. Listed Python and Java as environments we supply a demo for 3. Added Creating an Object and Threading issues sections
09	22-Mar-17	1. Added C++ demo information
08	30-May-16	1. Added StarLite and StarBright 2. Added status report to GetData method for reporting Filter status for sensors that self-detect their Filter State 3. Remove component for 32-bit applications on 64-bit Windows
07	07-Aug-14	1. Added Section Troubleshooting Regsvr32
06	30-Apr-14	1. Added true 64-bit version of OphirLMMeasurement a. Updated Section Additional Registrations b. Added Section Versions of OphirLMMeasurement
05	23-Apr-12	1. For 64-bit Windows the redistributable folder is renamed from **x64** to **x86_on_x64** 2. Updated explanation of using OphirLMMeasurement in 64-bit applications
04	13-Dec-11	1. Added to Section Additional Registrations description of uninstall of drivers as well as silent install/uninstall
03	26-Jul-11	1. Fix status numbers in Section DataReady_Handler. 2. BeamTrack sensor support added to COM object 3. Added section Configuring the Sensor's Measurement Mode 4. Moved list of status codes from section DataReady_Handler to new section GetData Status Codes and expanded the list to include codes returned for BeamTrack sensors 5. Listed MatLab as one of the IDE's that we supply a demo for
02	24-Nov-10	1. Fix Offset Measurement error for new pyroelectric sensors.
01	11-Nov-10	1. Initial Revision

2 Overview

2.1 Introduction

This document describes OphirLMMeasurement. This is the COM object provided by Ophir that allows user applications to interface with all Ophir USB speaking devices (Centauri, StarLite, StarBright, Juno, Juno+, Nova-II, all Pulsar models, USBI, Vega, and Ariel).

Besides their use as standalone, fully featured laser power/energy meters, Ophir devices can also be used through OphirLMMeasurement. This COM object allows system integrators to integrate the measurement expertise of Ophir equipment with in-house legacy analysis packages.

Demo client applications in VB.NET 2010/2019, VC# 2010/2019, VC++ 2010/2019, Python, Java, MatLab, and LabVIEW 8.6 are provided with the StarLab installation and can be found in C:\Program Files\Ophir Optronics\StarLab 3.xx\Automation Examples\Com object.

In addition, OphirLMMeasurement has been tested with Delphi 5. In practice, OphirLMMeasurement 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).

Special note for C++ users: Due the intricacy involved in using COM objects in C++, the C++ demo includes a wrapper class which provides a standard C++ interface to the COM object. To use this wrapper class, add OphirLMMeasurement.h and OphirLMMeasurement.cpp to your project, and #include "OphirLMMeasurement.h". The file program.cpp is a demo of how to use the class. See the comments in OphirLMMeasurement.h for more details.

Note: We provide a COM object and not a .NET automation because many clients' applications do not use .NET and it is easy to use a COM object from .NET.

Note: As of StarLab 3.20, the OphirFastX and OphirUsbX ActiveX packages are no longer included with the StarLab installation.

2.2 Improvements in OphirLMMeasurement

2.2.1 Support for All Devices in One Package

Centauri, StarLite, StarBright, Juno, Juno+, Nova-II, Vega, USBI, Pulsar, and Ariel are all supported by OphirLMMeasurement.

OphirLMMeasurement provides a common interface with all of the devices. Therefore devices with similar features should be drop-in compatible in the code.

2.2.2 Console Applications

OphirLMMeasurement doesn't need a window and can therefore work with a console client. This includes Microsoft based applications and other IDE's as well such as MatLab scripts.

2.2.3 Immediate Response

OphirLMMeasurement provides a mode in which each measurement is reported immediately when taken at frequencies of up to 100Hz.

2.2.4 Turbo Mode

Due to improvements in the underlying code, Turbo Mode is no longer necessary when measuring with Pulsar devices.

2.3 USB Details of Devices

The following lists the differences between the various Ophir devices vis-à-vis their USB communications. OphirLMMeasurement hides all this from the client application but the system integrator must be aware of this to ensure he sets up the system with the appropriate hardware support.

• Centauri	Up to 2 channels,	High Speed,	Self-powered,	one interrupt IN endpoint
• Juno	One channel,	Full Speed,	Bus Powered,	up to 5 interrupt IN endpoints

• Juno+	One channel,	Full Speed,	Bus Powered,	up to 5 interrupt IN endpoints
• Nova-II	One channel,	Full Speed,	Self powered,	one interrupt IN endpoint
• Pulsar	Up to 4 channels,	High Speed,	Self powered,	up to 2 interrupt IN endpoints
• USBI	One channel,	Full Speed,	Bus Powered,	one interrupt IN endpoint
• Vega	One channel,	Full Speed,	Self powered,	one interrupt IN endpoint
• StarLite	One channel,	High Speed,	Self powered,	one interrupt IN endpoint
• StarBright	One channel,	High Speed,	Self powered,	one interrupt IN endpoint
• Ariel	One channel,	Full Speed,	Self powered,	one interrupt IN endpoint

3 OphirLMMeasurement: Detailed Description

This section describes registration of OphirLMMeasurement as well as its methods, events, and error codes.

3.1 Registration

StarLab PC software (2.10 and higher) registers OphirLMMeasurement as part of the installation process of the StarLab application. Registration is also possible via the Menu of the StarLab application.

3.1.1 Additional Registrations

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

1. Install the Visual Studio 2019 Visual C++ redistributable, available at https://docs.microsoft.com/en-us/cpp/windows/latest-supported-vc-redist.
2. Copy the necessary files
   - For 32-bit applications (running on 32-bit or 64-bit Windows): From **C:\Program Files\Ophir Optronics\StarLab 3.xx\Automation Examples\Com object\Redistributable\x86** to the target PC
   - For 64-bit applications: From **C:\Program Files\Ophir Optronics\StarLab 3.xx\Automation Examples\Com object\Redistributable\x64** to the target PC.
3. Run **installophircom.bat** file on the target PC. This will install the USB drivers as well as register the OphirLMMeasurement COM object and all other components necessary for full functionality. The following is the contents of **installophircom.bat**
   - rem 1) install COM (this is Ophir specific)
   - regsvr32 OphirLMMeasurement.dll
   - rem 2) install Ophir device drivers (Ophir specific INF files)
   - swapinf StarlabDrivers
4. If there is a need to remove the software from the target PC and to clean its registry, run **RemoveOphirCOM.bat** that is provided. The following is the contents of **RemoveOphirCOM.bat**
   - rem 1) remove ophir devices drivers (Ophir specific INF files)
   - swapinf StarlabDriversRemove
   - rem 2) remove COM (this is Ophir specific)
   - regsvr32 /u OphirLMMeasurement.dll
5. The default COM object installation includes Ophir dialog windows informing the user to disconnect Ophir devices. If it is necessary to hide this message, then the install and uninstall batch files can be run silently. To do this, modify **installophircom.bat** and **RemoveOphirCOM.bat** as follows
   - **installophircom.bat**
     1. regsvr32 **/s** OphirLMMeasurement.dll
     2. swapinf StarlabDrivers **--silent**
   - **RemoveOphirCOM.bat**
     1. swapinf StarlabDriversRemove **--silent**
     2. regsvr32 /u **/s** OphirLMMeasurement.dll
   - Note: If you chose to install and remove silently, you must verify that the Ophir device has been disconnected before starting the process. Otherwise the installation will fail may leave unwanted information in the OC and you won't know about it

3.1.2 Troubleshooting Regsvr32

Microsoft reports that there are occasions where Regsvr32 doesn't execute properly even though it is being run as an Administrator (see http://support.microsoft.com/kb/249873). The way to overcome this is to re-run the Regsvr32 command from an elevated command prompt.

To open an elevated command prompt, follow these steps:

- ***Windows 8.1 and Windows 8***

  Swipe in from the right edge of the screen, and then tap **Search**. Or, if you are using a mouse, point to the lower-right corner of the screen, and then click **Search**. Type **Command Prompt** in the **Search** box, right-click **Command Prompt**, and then click **Run as administrator**. If you are prompted for an administrator password or for a confirmation, type the password, or click **Allow**.

- ***Windows 7 and Windows Vista***

  Click **Start**, type **Command Prompt** or **cmd** in the **Search** box, right-click **Command Prompt**, and then click **Run as administrator**. If you are prompted for an administrator password or for a confirmation, type the password, or click **Allow**.

- ***Windows 10 and Windows 11***

  Click the **Start** menu, type **Command Prompt** or **cmd** in the **Search** box, right-click **Command Prompt**, and then click **Run as administrator**. If you are prompted for an administrator password or for a confirmation, type the password, or click **Allow**.

3.1.3 Versions of OphirLMMeasurement

The OphirLMMeasurement component works in Windows Vista, 7, 8, 8.1, 10, and 11. We provide 2 variations.

- 32-bit component for 32-bit applications.
- 64-bit component for 64-bit applications.

As of StarLab 3.20, we no longer supply a component for 32-bit applications running on 64-bit machines. The ordinary 32-bit component will work with 32-bit applications on both 32-bit and 64-bit Windows.

3.2 Creating an Object

In order to use the library an object must be created. This object has methods which allow controlling the Ophir measuring devices. To create the object, the ProgID `OphirLMMeasurement.CoLMMeasurement` or the GUID `{B180613C-E514-4739-AADC-CAD4493910D7}` can be used. How precisely to create a COM object depends on your development environment; see the provided samples for some possibilities.

Some environments require an interface name to be specified. For version 10.0 of the COM object (which comes with StarLab 3.50) the interface is `ICoLMMeasurement2`. The older interface `ICoLMMeasurement` will still work, but will not allow accessing the functions added in this version. Environments that do not require specifying the interface name will automatically use the newer interface.

3.3 Threading issues

The COM object provided is Apartment-threaded. This means (approximately) that it can be only used directly from the thread it was created in. The simplest way to use this library from multiple threads is to create a separate object in each thread. Since the state of the instruments is shared between objects, this has the same effect.

3.4 Error Codes

- 0x00000000 (S_OK): No Error
- 0x80004001 (E_NOTIMPL): Function Not Implemented
- 0x80070057 (E_INVALIDARG): Invalid Argument
- 0x80004005 (E_FAIL): Unspecified Failure
- 0x80040200: Device not opened
- 0x80040201: Device Already Opened
- 0x80040202: Drivers cannot be loaded
- 0x80040203: Load File Missing
- 0x80040300: Device Failed
- 0x80040301: Device Firmware is Incorrect
- 0x80040302: Sensor Failed
- 0x80040303: Sensor Firmware is Incorrect
- 0x80040304: Bad Device Handle
- 0x80040305: Bad Sensor Channel
- 0x80040306: This Sensor is Not Supported
- 0x80040307: Not Applicable in this Device
- 0x80040308: The Device is no longer Available
- 0x80040400: Save To Sensor Failed
- 0x80040401: Param Error
- 0x80040402: Failed to create Safe Array
- 0x80040403: Not Applicable in this Sensor
- 0x80040404: Value Out of Range
- 0x80040405: Command Failed
- 0x80040500: Stream Mode Not Started
- 0x80040501: A channel is in Stream Mode

The OphirLMMeasurement COM object follows the standard COM practice of returning an HRESULT from its methods. The HRESULT can be passed to GetErrorFromCode to get a descriptive string, or the standard COM GetErrorInfo function can be called (see documentation for your environment). Some client environments (such as VB6 and all .NET languages) do not return this HRESULT from the method call; instead they throw an exception when a method returns a failure HRESULT. This exception will normally contain within it the error number and the descriptive string.

3.5 Methods and Events

3.5.1 Device Communications

Close	CloseAll	GetKnownWirelessDevices
OpenUSBDevice	OpenWirelessDevice	ResetAllDevices
ResetDevice	ScanUSB	ScanWireless

3.5.2 General Information and Diagnostics

GetDeviceCalibrationDueDate	GetDeviceInfo	GetDriverVersion
GetErrorFromCode	GetSensorCalibrationDueDate	GetSensorInfo
GetVersion	IsSensorExists	

3.5.3 Sensor Configuration

AddWavelength	DeleteWavelength	GetDiffuser
GetFilter	GetLowFreqPowerPulseFreq	GetMeasurementMode
GetPulsedPowerPulseWidth	GetPulseLengths	GetRanges
GetThreshold	GetWavelengths	GetWavelengthsExtra
ModifyWavelength	SaveSettings	SetDiffuser
SetFilter	SetLowFreqPowerPulseFreq	SetMeasurementMode
SetPulsedPowerPulseWidth	SetPulseLength	SetRange
SetThreshold	SetWavelength	

3.5.4 Measurement Delivery

ConfigureStreamMode	GetData	StartStream
StopAllStreams	StopStream	

3.5.5 Trigger Settings

GetExtTrigModes	GetExtTrigOnOff	GetExtTrigWindowTime
SetExtTrigMode	SetExtTrigOnOff	SetExtTrigWindowTime

3.5.6 Events

DataReady	PlugAndPlay	

3.5.7 Legacy Methods

Read	Write	

3.5.8 Alphabetic Listing

Method	Communications	Diagnostics	Configuration	Measurement	Trigger	Legacy
AddWavelength			•			
Close	•					
CloseAll	•					
ConfigureStreamMode				•		
DataReady (event)						
DeleteWavelength			•			
GetData				•		
GetDeviceCalibrationDueDate		•				
GetDeviceInfo		•				
GetDiffuser			•			
GetDriverVersion		•				
GetErrorFromCode		•				
GetExtTrigModes					•	
GetExtTrigOnOff					•	
GetExtTrigWindowTime					•	
GetFilter			•			
GetKnownWirelessDevices	•					
GetLowFreqPowerPulseFreq			•			
GetMeasurementMode			•			
GetPulsedPowerPulseWidth			•			
GetPulseLengths			•			
GetRanges			•			
GetSensorCalibrationDueDate		•				
GetSensorInfo		•				
GetThreshold			•			
GetVersion		•				
GetWavelengths			•			
GetWavelengthsExtra			•			
IsSensorExists		•				
ModifyWavelength			•			
OpenUSBDevice	•					
OpenWirelessDevice	•					
PlugAndPlay (event)						
Read						•
ResetAllDevices	•					
ResetDevice	•					
SaveSettings			•			
ScanUSB	•					
ScanWireless	•					
SetDiffuser			•			
SetExtTrigMode					•	
SetExtTrigOnOff					•	
SetExtTrigWindowTime					•	
SetFilter			•			
SetLowFreqPowerPulseFreq			•			
SetMeasurementMode						
SetPulsedPowerPulseWidth			•			
SetPulseLength			•			
SetRange			•			
SetThreshold			•			
SetWavelength			•			
StartStream				•		
StopAllStreams				•		
StopStream				•		
Write						•

3.6 Detailed Explanations of Methods

3.6.1 Conventions

3.6.1.1 Sensor Identification

The set of devices that OphirLMMeasurement supports includes multichannel devices which may have more than one sensor attached. In order to provide a common interface across all devices, OphirLMMeasurement uniquely identifies a sensor based on the device that it is attached to (hDevice) and the channel to which it is attached. For devices with only one channel <channel> should be 0. For a 2-channel device it can be 0 or 1, and for a 4-channel device 0, 1, 2, or 3.

3.6.1.2 Strings

In order to guarantee compatibility with the various Development Environments, OphirLMMeasurement returns strings in the BSTR type. Arrays of strings are returned as VARIANT *.

3.6.1.3 Common Parameters

Parameter	Description
[in] LONG hDevice	A handle to an open device, received from OpenUSBDevice or OpenWirelessDevice.
[in] LONG channel	A channel number, 0 to 3, as applicable to the device.
[out] VARIANT* options	A list of values that a setting can take, in the form of an array of BSTR. If the setting is not applicable to the chosen sensor, the array will be empty.
[out] LONG* index	The 0-based index of the currently selected value of a setting. If the setting is not applicable to the chosen sensor, -1.
[in] LONG index	The 0-based index of a value of a setting. This must be in the range of the options valid for the setting, as returned in the <options> parameter of a Get… method.

3.6.2 Device Communications

These methods are responsible for managing the communications with Ophir devices. This includes finding and registering the device(s) within OphirLMMeasurement, starting a communications session, and closing down communications with the device(s).

All USB speaking Ophir devices are supported. The Quasar wireless device and the EA-1 ethernet is not supported. The methods GetKnownWirelessDevices, OpenWirelessDevice, and ScanWireless are infrastructure stubs for future support and at present return error codes.

3.6.2.1 Close

Name	**Close**
Parameters	• [in] LONG hDevice
Use	This method ends the communication session with <hDevice>.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040304: Bad Device Handle • 0x80040501: A channel is in Stream Mode
See Also:	Device Communications, Alphabetic Listing, Common Parameters

3.6.2.2 CloseAll

Name	**CloseAll**
Parameters	None
Use	Equivalent to calling Close on all open devices.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040501: A channel is in Stream Mode
See Also:	Device Communications, Alphabetic Listing

3.6.2.3 GetKnownWirelessDevices

Name	**GetKnownWirelessDevices**
Parameters	• [out] VARIANT* serialNumbers
Use	This method is an infrastructure stub for future support and at present only returns an error code
Return Codes	• 0x80004001 (E_NOTIMPL): Function Not Implemented
See Also:	Device Communications, Alphabetic Listing

3.6.2.4 OpenUSBDevice

Name	**OpenUSBDevice**
Parameters	• [in] BSTR serialNumber • [out] LONG* hDevice
Use	Given <serialNumber> that was found with the ScanUSB method, will open a communication session with a device. <hDevice> is a handle which will be used for all communication with the device.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80070057 (E_INVALIDARG): Invalid Argument • 0x80040201: Device Already Opened • 0x80040300: Device Failed • 0x80040301: Device Firmware is Incorrect • 0x80040302: Sensor Failed • 0x80040303: Sensor Firmware is Incorrect • 0x80040306: This Sensor is Not Supported • 0x80040308: The Device is no longer Available • 0x80040405: Command Failed
See Also:	Device Communications, Alphabetic Listing, ScanUSB

3.6.2.5 OpenWirelessDevice

Name	**OpenWirelessDevice**
Parameters	• [in] BSTR name • [out] LONG* hDevice
Use	Given <name> that was found with the ScanWireless method, will open a communication session with a device. <hDevice> is a handle which will be used for all communication with the device.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80070057 (E_INVALIDARG): Invalid Argument • 0x80040201: Device Already Opened • 0x80040300: Device Failed • 0x80040301: Device Firmware is Incorrect • 0x80040302: Sensor Failed • 0x80040303: Sensor Firmware is Incorrect • 0x80040306: This Sensor is Not Supported • 0x80040308: The Device is no longer Available • 0x80040405: Command Failed
See Also:	Device Communications, Alphabetic Listing, ScanWireless

3.6.2.6 ResetAllDevices

Name	**ResetAllDevices**
Parameters	None
Use	Closes and restarts all open devices. After calling this function ScanUsb should be called to redetect the devices. Note: This function is only needed for the USBI instrument. All other instruments detect automatically when the sensor is changed. Note: This method will only function when no devices are streaming data (i.e. any channels which were put into Stream mode with the StartStream method have been taken out of stream mode with StopStream or StopAllStreams).
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040501: A channel is in Stream Mode
See Also:	Device Communications, Alphabetic Listing

3.6.2.7 ResetDevice

Name	**ResetDevice**
Parameters	• [in] LONG hDevice
Use	Restarts <hDevice>. Used after changing sensor attached to the device Note: This function is only needed for the USBI instrument. All other instruments detect automatically when the sensor is changed. Note: This command method will only function when the device isn't streaming data (i.e. any channels which were put into Stream mode with the StartStream method have been taken out of stream mode with StopStream or StopAllStreams).
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040308: The Device is no longer Available • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Device Communications, Alphabetic Listing, Common Parameters

3.6.2.8 ScanUSB

Name	**ScanUSB**
Parameters	• [out] VARIANT* serialNumbers
Use	Scans the USB for all Ophir devices that are presently attached to the PC. Note: If the device is a Pulsar, this method will load it with its firmware. The firmware files are located under the directory where OphirLMMeasurement.dll is located, in the subdirectory **firmware**. Updated firmware files, with the names **FU4Axxx.hex**, **FU4Bxxx.hex**, and **FU4Fxxx.ttf**, should be placed in the same location. <serialNumbers> will contain an array of BSTR.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040202: Drivers cannot be loaded • 0x80040203: Load File Missing • 0x80040402: Failed to create Safe Array • 0x80040501: A channel is in Stream Mode
See Also:	Device Communications, Alphabetic Listing, OpenUSBDevice

3.6.2.9 ScanWireless

Name	**ScanWireless**
Parameters	• [out] VARIANT* names
Use	Scans for wireless Ophir devices. Currently the only wireless device supported is the Ariel. <names> will contain an array of BSTR.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040202: Drivers cannot be loaded • 0x80040203: Load File Missing • 0x80040402: Failed to create Safe Array • 0x80040501: A channel is in Stream Mode
See Also:	Device Communications, Alphabetic Listing, OpenWirelessDevice

3.6.3 General Information and Diagnostics

These methods provide general purpose information about the devices, sensors, and software that are presently attached and participating in the present communication session

3.6.3.1 GetDeviceInfo

Name	**GetDeviceInfo**
Parameters	• [in] LONG hDevice • [out] BSTR* deviceName • [out] BSTR* romVersion • [out] BSTR* serialNumber
Use	Gets name, ROM version and serial number of the device.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040308: The Device is no longer Available
See Also:	General Information and Diagnostics, Alphabetic Listing, Common Parameters

3.6.3.2 GetDeviceCalibrationDueDate

Name	**GetDeviceCalibrationDueDate**
Parameters	• [in] LONG hDevice • [out] DATE* dueDate
Use	Get the date this instrument is due to be calibrated. Note: Some instruments do not yet report the calibration due date and will return error 0x80040307.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040307: Not Applicable in this Device • 0x80040308: The Device is no longer Available
See Also:	General Information and Diagnostics, Alphabetic Listing, Common Parameters, GetSensorCalibrationDueDate

3.6.3.3 GetDriverVersion

Name	**GetDriverVersion**
Parameters	• [out] BSTR* info
Use	Get version of drivers that OphirLMMeasurement uses.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80004005 (E_FAIL): Unspecified Failure
See Also:	General Information and Diagnostics, Alphabetic Listing

3.6.3.4 GetErrorFromCode

Name	**GetErrorFromCode**
Parameters	• [in] LONG errorCode • [out] BSTR* info
Use	Given an error code, will pass back a string explanation of the error.
Return Codes	• 0x00000000 (S_OK): No Error
See Also:	General Information and Diagnostics, Alphabetic Listing

3.6.3.5 GetSensorCalibrationDueDate

Name	**GetSensorCalibrationDueDate**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] DATE* dueDate
Use	Get the date this sensor is due to be calibrated. Note: Some instruments do not yet report the calibration due date and will return error 0x80040307.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040307: Not Applicable in this Device • 0x80040308: The Device is no longer Available
See Also:	General Information and Diagnostics, Alphabetic Listing, Common Parameters, GetDeviceCalibrationDueDate

3.6.3.6 GetSensorInfo

Name	**GetSensorInfo**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] BSTR* serialNumber • [out] BSTR* sensorType • [out] BSTR* sensorName
Use	Gets serial number, type (thermopile, photodiode, pyroelectric), and name of sensor.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available
See Also:	General Information and Diagnostics, Alphabetic Listing, Common Parameters

3.6.3.7 GetVersion

Name	**GetVersion**
Parameters	• [out] LONG* version
Use	Get version of the OphirLMMeasurement COM object.
Return Codes	• 0x00000000 (S_OK): No Error
See Also:	General Information and Diagnostics, Alphabetic Listing

3.6.3.8 IsSensorExists

Name	**IsSensorExists**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] VARIANT_BOOL* exists
Use	Check if a sensor is attached to <hDevice> <channel>.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040308: The Device is no longer Available
See Also:	General Information and Diagnostics, Alphabetic Listing, Common Parameters

3.6.4 Sensor Configuration

These methods are used to configure the measurement parameters of the sensor. Some of these methods apply to all types of sensors (e.g. Range methods) and some are sensor-type specific (e.g. Filter methods apply to Photodiode sensors only, Diffuser methods are only for Pyroelectric and PD Energy sensors, etc.)

3.6.4.1 AddWavelength

Name	AddWavelength
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] LONG wavelength
Use	Add <wavelength> to list of favorite wavelengths of the selected sensor. Wavelength: Must be between the lower and upper wavelength limits as returned by the GetWavelengthsExtra method. Note: This method is only applicable for sensors with a continuous spectrum. The set of lasers cannot be altered in sensors with a discrete set of lasers settings.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040403: Not Applicable in this Sensor • 0x80040404: Wavelength Out of Range • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, DeleteWavelength, GetWavelengths, GetWavelengthsExtra, ModifyWavelength, SetWavelength

3.6.4.2 DeleteWavelength

Name	**DeleteWavelength**
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] LONG index
Use	Delete wavelength of selected sensor specified in <index>. Note: If the currently selected wavelength is deleted, a different wavelength is automatically chosen. If there is only one wavelength left it cannot be deleted. Note: This method is only applicable for sensors with a continuous spectrum. The set of lasers cannot be altered in sensors with a discrete set of lasers settings
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040401: Param Error • 0x80040403: Not Applicable in this Sensor • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, AddWavelength, GetWavelengths, GetWavelengthsExtra, ModifyWavelength, SetWavelength

3.6.4.3 GetDiffuser

Name	**GetDiffuser**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] LONG* index • [out] VARIANT* options
Use	Get the diffuser state of the selected sensor and what the diffuser options are. For sensors that are not equipped with an adjustable diffuser, <index> will always return 0. This method applies to Pyroelectric and PD Energy sensors only.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040402: Failed to create Safe Array
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, SetDiffuser

3.6.4.4 GetFilter

Name	**GetFilter**
Parameters	• in] LONG hDevice • [in] LONG channel • [out] LONG* index • [out] VARIANT* options
Use	Get the filter state of the selected sensor and what the filter options are. For sensors that are not equipped with an adjustable filter, <index> will always return 0. This method applies to Photodiode sensors only.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040402: Failed to create Safe Array
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, SetFilter

3.6.4.5 GetLowFreqPowerPulseFreq

Name	**GetLowFreqPowerPulseFreq**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] DOUBLE* value • [out] DOUBLE* min • [out] DOUBLE* max
Use	Get the current pulse frequency for low frequency power measurement mode, and the maximum and minimum supported values. This method applies to Photodiode sensors only, and is only supported on some instruments.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040403: Not Applicable in this Sensor
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, SetLowFreqPowerPulseFreq

3.6.4.6 GetMeasurementMode

Name	**GetMeasurementMode**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] LONG* index • [out] VARIANT* options
Use	Get the measurement mode of the selected sensor and what the measurement modes are. This method applies to all types of sensors.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040402: Failed to create Safe Array
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, SetMeasurementMode

3.6.4.7 GetPulsedPowerPulseWidth

Name	**GetPulsedPowerPulseWidth**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] LONG* value • [out] LONG* min • [out] LONG* max
Use	Get the current pulse width for pulsed power measurement mode, and the maximum and minimum supported values, in milliseconds. This method applies to Thermopile sensors only, and is only supported on some instruments.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040403: Not Applicable in this Sensor
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, SetPulsedPowerPulseWidth

3.6.4.8 GetPulseLengths

Name	**GetPulseLengths**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] LONG* index • [out] VARIANT* options
Use	Get the selected pulse length of the sensor in selected channel and what the pulse length options are. This method applies to Pyroelectric and PD Energy sensors only. The strings returned are the maximum pulse length (in time) for each of the various pulse length settings.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040402: Failed to create Safe Array
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, SetPulseLength

3.6.4.9 GetRanges

Name	**GetRanges**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] LONG* index • [out] VARIANT* options
Use	Get the measurement range of the selected sensor and the list of all available measurement ranges. This method applies to all types of sensors.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040402: Failed to create Safe Array
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, SetRange

3.6.4.10 GetThreshold

Name	**GetThreshold**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] LONG* index • [out] VARIANT* options
Use	Get the selected threshold setting of the sensor in selected channel and what the threshold options are. This method applies to Thermopile sensors measuring energy and to some Pyroelectric sensors as well. For Thermopile sensors, the set of available thresholds will generally be "LOW MEDIUM HIGH". For Pyroelectric sensors with adjustable thresholds, there is a threshold setting per pulse length.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040402: Failed to create Safe Array
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, SetThreshold

3.6.4.11 GetWavelengths

Name	**GetWavelengths**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] LONG* index • [out] VARIANT* options
Use	Gets the selected wavelength of selected sensor and list of all available wavelengths. This method applies to all types of sensors. Response is the same format for sensors with discrete and continuous spectrums. For additional spectrum information use GetWavelengthsExtra. For sensors with a continuous spectrum, these strings will be numeric, for sensors with discrete spectrums, these strings may contain non-numeric characters as well (e.g. VIS or NIR).
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040402: Failed to create Safe Array
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, AddWavelength, DeleteWavelength, GetWavelengthsExtra, ModifyWavelength, SetWavelength

3.6.4.12 GetWavelengthsExtra

Name	**GetWavelengthsExtra**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] VARIANT_BOOL*modifiable • [out] LONG* minWavelength • [out] LONG* maxWavelength

Use	Get additional wavelength information of selected channel.
	
	If spectrum is continuous, then
	• modifiable: True
	• minWavelength, maxWavelength: minimum and maximum limits for wavelength adjustment with the AddWavelength and ModifyWavelength methods
	
	If spectrum is discrete then
	• modifiable: False
	• minWavelength, maxWavelength: Not applicable

Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040403: Not Applicable in this Sensor • 0x80040501: A channel is in Stream Mode
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, AddWavelength, DeleteWavelength, GetWavelengths, ModifyWavelength, SetWavelength

3.6.4.13 ModifyWavelength

Name	**ModifyWavelength**
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] LONG index • [in] LONG wavelength
Use	Modify selected sensor's favorite wavelength stored at <index> to the value <wavelength>. Note: <wavelength> must be between the lower and upper wavelength limits as returned by the GetWavelengthsExtra method. Note: This method is only applicable for sensors with a continuous spectrum. The set of lasers cannot be altered in sensors with a discrete set of lasers settings.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040401: Param Error • 0x80040403: Not Applicable in this Sensor • 0x80040404: Wavelength Out of Range • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, AddWavelength, DeleteWavelength, GetWavelengths, GetWavelengthsExtra, SetWavelength

3.6.4.14 SaveSettings

Name	**SaveSettings**
Parameters	• [in] LONG hDevice • [in] LONG channel
Use	Save all configuration parameters of selected sensor.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040400: Save To Sensor Failed • 0x80040501: A channel is in Stream Mode
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters

3.6.4.15 SetDiffuser

Name	**SetDiffuser**
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] LONG index
Use	Set diffuser state of selected sensor to <index> This method applies to Pyroelectric and PD Energy sensors only.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040401: Param Error • 0x80040403: Not Applicable in this Sensor • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, GetDiffuser

3.6.4.16 SetFilter

Name	**SetFilter**
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] LONG index
Use	Set filter state of selected sensor to <index>. This method applies to Photodiode sensors only.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040401: Param Error • 0x80040403: Not Applicable in this Sensor • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, GetFilter

3.6.4.17 SetLowFreqPowerPulseFreq

Name	**SetLowFreqPowerPulseFreq**
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] DOUBLE value
Use	Set the pulse frequency for low frequency power measurement mode. This method applies to Photodiode sensors only, and is only supported on some instruments.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040403: Not Applicable in this Sensor • 0x80040404: Value Out of Range • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, GetLowFreqPowerPulseFreq

3.6.4.18 SetMeasurementMode

Name	**SetMeasurementMode**
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] LONG index
Use	Set measurement mode of selected sensor to <index>. This method applies to all types of sensors.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040401: Param Error • 0x80040403: Not Applicable in this Sensor • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also	Sensor Configuration, Alphabetic Listing, Common Parameters, GetMeasurementMode

3.6.4.19 SetPulsedPowerPulseWidth

Name	**SetPulseLength**
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] LONG value
Use	Get the pulse width for pulsed power measurement mode, in milliseconds. This method applies to Thermopile sensors only, and is only supported on some instruments.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040403: Not Applicable in this Sensor • 0x80040404: Value Out of Range • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, GetPulsedPowerPulseWidth

3.6.4.20 SetPulseLength

Name	**SetPulseLength**
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] LONG index
Use	Set pulse length setting of the selected sensor to <index>. This method applies to Pyroelectric and PD Energy sensors only.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040401: Param Error • 0x80040403: Not Applicable in this Sensor • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, GetPulseLengths

3.6.4.21 SetRange

Name	**SetRange**
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] LONG index
Use	Set measurement range of selected sensor to <index>. This method applies to all types of sensors.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040401: Param Error • 0x80040403: Not Applicable in this Sensor • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, GetRanges

3.6.4.22 SetThreshold

Name	**SetThreshold**
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] LONG index
Use	Set threshold of selected sensor to <index>. This method applies to Thermopile sensors measuring energy and to some Pyroelectric sensors as well.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040401: Param Error • 0x80040403: Not Applicable in this Sensor • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, GetThreshold

3.6.4.23 SetWavelength

Name	**SetWavelength**
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] LONG index
Use	Set wavelength of selected sensor to <index>. This method applies to all types of sensors.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040401: Param Error • 0x80040403: Not Applicable in this Sensor • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Sensor Configuration, Alphabetic Listing, Common Parameters, AddWavelength, DeleteWavelength, GetWavelengths, GetWavelengthsExtra, ModifyWavelength

3.6.5 Measurement Delivery

See section Data Streams for detailed explanations of the various streams of data that come out of the devices and through OphirLMMeasurement, setting up the stream, starting and stopping it, and parsing the response.

3.6.5.1 ConfigureStreamMode

Name	**ConfigureStreamMode**
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] LONG mode • [in] LONG nValue
Use	Configure measurement delivery for the selected device and channel. See section Data Streams for detailed explanations of setting up the stream, starting and stopping it, and parsing the response. **Value of mode** / **Meaning** / **Acceptable values for nValue** 0 / Turbo mode / 0: off; 1: on 1 / Turbo frequency / Any frequency acceptable to the device 2 / Immediate mode / 0: off; 1: on
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040403: Not Applicable in this Sensor • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Measurement Delivery, Alphabetic Listing, Common Parameters

3.6.5.2 GetData

Name	**GetData**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] VARIANT* arrayValue • [out] VARIANT* arrayTimestamp • [out] VARIANT* arrayStatus
Use	This method uploads the data payload that the Ophir device has delivered to OphirLMMeasurement. It should be called after the DataReady event has been received. If called when no new data is available, then the arrays will be empty. See section Data Streams for detailed explanations of setting up the stream, starting and stopping it, and parsing the response.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040402: Failed to create Safe Array • 0x80040500: Stream Mode Not Started
See Also:	Measurement Delivery, Alphabetic Listing, Common Parameters, Data Streams, GetData Status Codes

3.6.5.3 StartStream

Name	**StartStream**
Parameters	• [in] LONG hDevice • [in] LONG channel
Use	Instruct the selected sensor to begin data streaming measurements to the PC. The format of the stream (buffered, turbo, immediate) can be configured with the ConfigureStreamMode method. When StartStream is called, the COM object flushes the buffer of previously gathered measurements. OphirLMMeasurement will fire the DataReady event when a new data packet has arrived and is ready for reading by the client application. See section Data Streams for detailed explanations of setting up the stream, starting and stopping it, and parsing the response.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80004005 (E_FAIL): Unspecified Failure • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040403: Not Applicable in this Sensor • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Measurement Delivery, Alphabetic Listing, Common Parameters, Data Streams

3.6.5.4 StopAllStreams

Name	**StopAllStreams**
Parameters	None
Use	Stops data streams of all sensors. See section Data Streams for detailed explanations of setting up the stream, starting and stopping it, and parsing the response.
Return Codes	• 0x00000000 (S_OK): No Error
See Also:	Measurement Delivery, Alphabetic Listing, Data Streams

3.6.5.5 StopStream

Name	**StopStream**
Parameters	• [in] LONG hDevice • [in] LONG channel
Use	Stops data streaming from selected sensor. See section Data Streams for detailed explanations of setting up the stream, starting and stopping it, and parsing the response.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040500: Stream Mode Not Started
See Also:	Measurement Delivery, Alphabetic Listing, Common Parameters, Data Streams

3.6.6 Trigger Settings

These methods are for driving the Pulsar's and Centauri's External Trigger mechanisms. External Trigger functionality is provided on the Pulsar with Pyrolectric and PD Energy sensors, and on the Centauri with all sensors.

See section External Trigger Settings for a brief explanation of the External Trigger mechanism. For a more detailed explanation please see **External Triggers and Missing Pulses** in the ***StarLab_User_Manual*** that is included in the StarLab Installation package.

3.6.6.1 GetExtTrigModes

Name	**GetExtTrigModes**
Parameters	• [in] LONG hDevice • [out] LONG* index • [out] VARIANT* options
Use	Gets the selected external trigger mode of the device and the set of available external trigger modes.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040307: Not Applicable in this Device • 0x80040308: The Device is no longer Available • 0x80040402: Failed to create Safe Array
See Also:	Trigger Settings, Alphabetic Listing, Common Parameters, External Trigger Modes

3.6.6.2 GetExtTrigOnOff

Name	**GetExtTrigOnOff**
Parameters	• [in] LONG hDevice • [in] LONG channel • [out] LONG* index • [out] VARIANT* options
Use	To get the external trigger state.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040402: Failed to create Safe Array
See Also:	Trigger Settings, Alphabetic Listing, Common Parameters

3.6.6.3 GetExtTrigWindowTime

Name	**GetExtTrigWindowTime**
Parameters	• [in] LONG hDevice • [out] LONG* extTrigWindow
Use	Get the external trigger window time of <hDevice> in microseconds. Window time can be between 1 and 65535 uS on the Pulsar, and between 1 and 50000 uS on the Centauri.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040307: Not Applicable in this Device • 0x80040308: The Device is no longer Available
See Also:	Trigger Settings, Alphabetic Listing, Common Parameters, External Trigger Window

3.6.6.4 SetExtTrigMode

Name	**SetExtTrigMode**
Parameters	• [in] LONG hDevice • [in] LONG index
Use	Set the external trigger mode of <hDevice>.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040307: Not Applicable in this Device • 0x80040308: The Device is no longer Available • 0x80040401: Param Error • 0x80040405: Command Failed
See Also:	Trigger Settings, Alphabetic Listing, Common Parameters, External Trigger Modes

3.6.6.5 SetExtTrigOnOff

Name	**SetExtTrigOnOff**
Parameters	• [in] LONG hDevice • [in] LONG channel • [in] LONG* index
Use	Set the external trigger state.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040305: Bad Sensor Channel • 0x80040308: The Device is no longer Available • 0x80040401: Param Error • 0x80040403: Not Applicable in this Sensor • 0x80040405: Command Failed • 0x80040501: A channel is in Stream Mode
See Also:	Trigger Settings, Alphabetic Listing, Common Parameters

3.6.6.6 SetExtTrigWindowTime

Name	**SetExtTrigWindowTime**
Parameters	• [in] LONG hDevice • [in] LONG extTrigWindowTime
Use	Set the external trigger window time of <hDevice> in microseconds. Window time can be between 1 and 65535 uS on the Pulsar, and between 1 and 50000 uS on the Centauri.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040307: Not Applicable in this Device • 0x80040308: The Device is no longer Available • 0x80040401: Param Error • 0x80040405: Command Failed
See Also:	Trigger Settings, Alphabetic Listing, Common Parameters, External Trigger Window

3.6.7 Events

3.6.7.1 DataReady

Name	**DataReady**
Parameters	• [in] LONG hDevice • [in] LONG channel
Use	This event is fired to inform the client application that the sensor at <hDevice> <channel> has delivered a new data payload. OphirLMMeasurement will not deliver the data to the client application until it calls the GetData method. Note: Further DataReady events for the specified sensor will not be fired until OphirLMMeasurement's flag has been cleared by the client application calling GetData.
Return Codes	N/A
See Also:	Measurement Delivery, Events, Alphabetic Listing, Common Parameters

3.6.7.2 PlugAndPlay

Name	**PlugAndPlay**
Parameters	None
Use	Fired when a device that has been found by the ScanUSB method has been disconnected from the system. The client application can rescan the USB to discover which device has been disconnected.
Return Codes	N/A
See Also:	Device Communications, General Information and Diagnostics, Events, Alphabetic Listing

3.6.8 Legacy Methods

The communication protocol between the USBI, Vega, Nova-II, StarLite, StarBright, Centauri, and Ariel devices with the PC is based on commands from the PC controller and responses from the device that are ASCII strings. This is also true for Thermopile and Photodiode sensors that are attached to the Juno, Juno+ and Pulsar devices.

At Ophir we highly recommend using OphirLMMeasurement for all measurement needs. However, end users that are porting from RS232 based solutions or that are migrating to OphirLMMeasurement from OphirUsbX may want to continue with the ASCII command paradigm. Therefore these methods are included in OphirLMMeasurement.

The function sequence is always a call to Write followed by a call to Read, i.e. every Write must be followed by a Read, and every Read must be preceded by a Write. Failure to adhere to this sequence may cause communication trouble.

Note: For a full description of available commands, please see **"*Ophir User Commands.pdf*"** included in the installation.

3.6.8.1 Read

Name	**Read**
Parameters	• [in] LONG hDevice • [out] BSTR* reply
Use	Read <reply> from <hDevice>. <reply> is in the form of an ASCII string.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040308: The Device is no longer Available • 0x80040405: Command Failed
See Also:	Legacy Methods, Alphabetic Listing, Common Parameters, Working with Legacy Methods

3.6.8.2 Write

Name	**Write**
Parameters	• [in] LONG hDevice • [in] BSTR command
Use	Write <command> to <hDevice>. <command> is in the form of an ASCII string.
Return Codes	• 0x00000000 (S_OK): No Error • 0x80040200: Device not opened • 0x80040304: Bad Device Handle • 0x80040308: The Device is no longer Available • 0x80040405: Command Failed
See Also:	Legacy Methods, Alphabetic Listing, Common Parameters, Working with Legacy Methods

4 Data Streams

See Measurement Delivery for the list of methods that manage the data delivery from the Ophir devices

4.1 Stream Formats

OphirLMMeasurement provides three types of data stream formats from the various devices to the client application.

- Standard
- Turbo
- Immediate

4.1.1 Standard

This is the default data stream that OphirLMMeasurement uses. Data is buffered within OphirLMMeasurement and which then fires the DataReady event to inform the client application that new data is available. This event can be fired as often as once every 50 milliseconds depending on how quickly the data is coming in and how quickly the client application calls the GetData method to read out the data from OphirLMMeasurement.

Besides OphirLMMeasurement's buffering, the Juno and Pulsar devices have hardware buffering mechanisms that enable them to reach very high measurement frequencies (10 KHz and 25 KHz respectively)

4.1.2 Turbo

The USBI, Vega, and Nova-II devices reach 130Hz of measurement that they can deliver every pulse in Standard format. In order to reach higher frequencies (2000Hz) they have to work in Turbo Mode. Using the ConfigureStreamMode method, the client application can tell OphirLMMeasurement that it wants to work in Turbo Mode and what is the expected frequency of the laser being measured.

There is no need to use Turbo Mode in order to reach high frequency measurements with the Juno and Pulsar devices.

4.1.3 Immediate

For clients that want to synchronize their Ophir measurements with something else (for example energy of a pulse together with a beam profile of the pulse captured by a camera in another piece of software such as BeamGage), OphirLMMeasurement provides Immediate Mode streams. Using the ConfigureStreamMode method to put OphirLMMeasurement in Immediate Mode, whenever a new measurement arrives, OphirLMMeasurement immediately fires the DataReady event instead of buffering the reading. Do not use immediate mode with high frequency data as the data delivery mechanism will not be able to keep up.

Note: If ConfigureStreamMode is not called before StartStream, OphirLMMeasurement assumes that the client application is using Standard Mode. However if ConfigureStreamMode is called to setup OphirLMMeasurement for Turbo or Immediate Modes, it must be called again to return to Standard Mode. Note: for the Pulsar, all channels must be closed and reopened before the Stream Mode can be reconfigured.

4.2 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 C# 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

- Setup and Start
- Collection
- Stop and Close Down

4.2.1 Setup and Start

Start of Function

```
/* Scan the USB for Ophir devices that has been attached to computer.
Get array of devices serial numbers */
ScanUSB (snNum)

/* The following opens the first USB device and gets the device handle */
OpenUSBDevice (snNum(0), nHandle)

/* Optional. Set the measurement parameters (i.e. Range, Wavelength, etc) if necessary */
Set Measurement Parameters ()

/* Optional – Configure measurement mode of sensor. The set of possible modes is dependent
on the type of sensor in use. See Configuring the Sensor's Measurement Mode */
SetMeasurementMode ()

/* Optional - Configure mode of stream. This section is different for the 3 different stream
modes. See Configuring the Various Stream Modes */
Configure the Stream Mode ()

/* Start the measurement process */
StartStream (nHandle, nChannel)
```

End of Function

4.2.2 DataReady_Handler

Start of Handler

```
/* This handler wakes up when the DataReady event is fired. The format of the data is identical
in all 3 modes; the only difference being that dataArray.Length should always be 1 if set up for
immediate mode

Measurements are delivered as three arrays containing the following data
    • Timestamp(double) : Time of measurement (in milliseconds)
    • value(double): Measurement (in watts or joules)
    • status(long): See section GetData Status Codes for a detailed explanation of
      the status codes for the various measurement modes

GetData (nHandle, channel, dataArray, timeStampArray, statusArray)
For index = 0 to dataArray.Length - 1
    LabelMeasurements(channel).Text  =arrayValue(index)          // value
    LabelTime(channel).Text      =      arrayTimestamp(index)   // time in milliseconds
    LabelStatus(channel).Text    =      arrayStatus(index)      // status
Next
```

End of Hander

4.2.3 Stop and Close Down

This section is identical across all 3 stream modes

Start of Function

```
/* Stop gathering data */
StopStream (nHandle, nChannel)
/* Close device*/
Close (nHandle)
```

End of Function

4.2.4 Configuring the Various Stream Modes

// values of mode and nValue parameters of the ConfigureStreamMode method

Value of mode	Meaning	Acceptable values for nValue
0	Turbo mode	0: off; 1: on
1	Turbo frequency	Any frequency acceptable to the device
2	Immediate mode	0: off; 1: on

4.2.4.1 Standard

Start of Function

```
/* OphirLMMeasurement defaults to this state. Only necessary if a different type if stream mode
was configured and now want to return to Standard */
ConfigureStreamMode (nHandle, nChannel, 0, 0)        // Turbo Mode turned off
ConfigureStreamMode (nHandle, nChannel, 2, 0)        // Immediate Mode turned off
```

End of Function

4.2.4.2 Turbo

Start of Function

```
/* Set expected frequency of laser and Turbo mode to on. Reminder: Turbo Mode is
unnecessary for the Juno and Pulsar devices. Also, it is only applicable for Pyroelectric and PD
Energy sensors. If Turbo frequency is set to greater than the max frequency that the sensor
can handle, an error will be returned that contains the max possible Turbo frequency */
ConfigureStreamMode (nHandle, nChannel, 1, 5000)     // Turbo frequency set to 5000Hz
ConfigureStreamMode (nHandle, nChannel, 0, 1)        // Turbo Mode turned on
```

End of Function

4.2.4.3 Immediate

Start of Function

```
/* Set Turbo Mode off and Immediate Mode on */
ConfigureStreamMode (nHandle, nChannel, 0, 0)        // Turbo Mode turned off
ConfigureStreamMode (nHandle, nChannel, 2, 1)        // Immediate Mode turned on
```

End of Function

4.2.5 Configuring the Sensor's Measurement Mode

This section lists the various types of Ophir sensors and which measurement modes they support. To get the appropriate list of modes for the sensor in use, call the GetMeasurementMode() method. To configure the sensor for measurement, call the SetMeasurementMode() method. Note that not all instruments support all modes.

Sensor Type	Measurement Modes
Thermopile Sensor	Power, Energy, Pulsed Power
Thermopile Sensor with Beam Tracking	Power, Energy, Pulsed Power, Power with Track
Photodiode Sensor	Power, Exposure, Low Freq Power
Pyroelectric and PD Energy Sensors	Power, Energy, Exposure

4.2.6 GetData Status Codes

The following is the list of status codes that can be returned with the measurements when calling the GetData() method. Other codes that may be reported are reserved for internal use or future expansion.

Status Code	Meaning	When and Where
0	OK	Power/Energy/Exposure measurements for all sensors
1	Overrange	Power and Energy measurements for all sensors
2	Saturated	Power and Energy measurements for all sensors
3	Missing	Energy with Pyroelectric and PD Energy sensors on a Pulsar or Centauri device
4	Reset state	Single shot energy or Pulsed Power with a Thermopile sensor
5	Waiting	Single shot energy or Pulsed Power with a Thermopile sensor
6	Summing	Single shot energy or Pulsed Power with a Thermopile sensor
7	Timeout	Single shot energy or Pulsed Power with a Thermopile sensor
8	Peak Over	Single shot energy or Pulsed Power with a Thermopile sensor
9	Energy Over	Single shot energy or Pulsed Power with a Thermopile sensor
0x010000	X measurement OK	Track measurements with a BeamTrack sensor
0x010001	X measurement Error	Track measurements with a BeamTrack sensor
0x020000	Y measurement OK	Track measurements with a BeamTrack sensor
0x020001	Y measurement Error	Track measurements with a BeamTrack sensor
0x030000	Size measurement OK	Track measurements with a BeamTrack sensor
0x030001	Size measurement Error	Track measurements with a BeamTrack sensor
0x030002	Size measurement Warning	Track measurements with a BeamTrack sensor
0x040001	Filter State Change	Photodiode sensors with built-in filter state detection reporting a change in filter state (in to out or vice-versa)
0x050000	Pulse frequency	Power and Energy with Pyroelectric and PD Energy sensors
0x100000	Temperature	Thermopile sensor
0x200000	Alert Hot	Thermopile sensor
0x300000	Pulse Width	Pulsed Power with a Thermopile sensor
0x400000	PfP Energy	Pulsed Power with a Thermopile sensor

5 External Trigger Settings

The Pulsar and Centauri devices have a sophisticated External Trigger mechanism. This allows synchronization between pulse measurements and other inputs (or outputs) in the measurement environment.

Note: On the Pulsar, External Trigger functionality (including Missing Pulses) is available for Pyrolectric and PD Energy sensors only. On the Centauri, edge triggers are available for Pyrolectric and PD Energy sensors only; level triggers are available for all sensors.

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 for the relevant sensors

5.1 External Trigger Modes

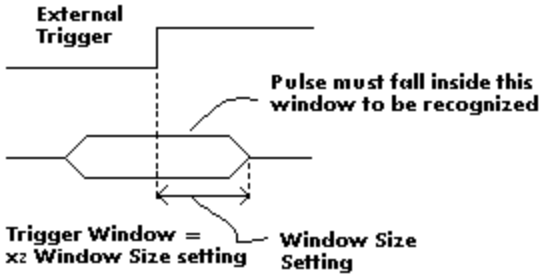
The Pulsar External Trigger can be set to one of 6 modes (4 input modes, 2 output modes). The Centauri External Trigger can be set to one of 4 input modes – the output modes are not supported.

Input Modes (Centauri and Pulsar)	*Description*
Rising Edge	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 after 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. Pulses are ignored if they arrive outside the Window Time.
Falling Edge	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 after 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. Pulses are ignored if they arrive outside the Window Time.
High Level	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.
Low Level	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 (Pulsar only)	*Description*
Active High	Every time a pulse arrives on the sensor 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	Every time a pulse arrives on the sensor detector the output goes low for 10us and then returns back to high. The default level of the output with no pulses is high.

5.2 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 device will report a MISSING MEASUREMENT.



6 Working with Legacy Methods

The preferred communication protocol between the client application and the Ophir device is the one described in Data Streams above. However, for customers already familiar with the command and response protocol supported by Ophir devices, we include that option as well.

The command and response protocol is based on ASCII string flow between the PC application and the device. Use the Read and Write methods, to have OphirLMMeasurement act as a conduit between the two, without any processing in either direction. It then becomes the client application's responsibility to format the string written to the device and to parse the string returned appropriately.

Below is a sample session of the command and response flow between an Ophir device and the PC application. It sets up the sensor for power measurement and gathers data until a pre-defined upper bound value has been measured. For simplicity's sake, most error-checking is ignored.

```
Start of Function
        /* Opens the USB and get the device handle */
        ScanUSB (snArray)
        OpenUSBDevice (snArray[0], hDevice)
        if not (IsSensorExists (hDevice, 0) )       // verify that there is a sensor attached
                goto EXIT

        Write (hDevice, "FP");                      // put sensor in power mode
        Read (hDevice, *reply);

        If reply  != "*"                            // verify acknowledgement from device
                Goto EXIT

        /* Gather measurements until upper bound is reached */
        Start_Loop
                Write (hDevice, "SP");              // tell device to send power measurement
                Read (hDevice, *reply);
                Response = ConvertToReading (reply)
                If Response > UPPER_LIMIT Then
                        Break out of Loop
        End_Loop

        EXIT:
                Close (hDevice);
End of Function
```