HoribaLabViewSDK

Antidoc v2.0.5, Niko Naredi-Rainer

Table of Contents

Chapter 1. Project description

The HORIBA SDK is a powerful tool for anyone working with HORIBA scientific instruments. It simplifies the process of instrument control and data acquisition, allowing users to focus on their research and development tasks. Whether you are conducting experiments, performing quality control, or developing new applications, this SDK provides the necessary tools to streamline your workflow and enhance productivity.

Chapter 2. DQMH® modules

This section describes DQMH® module responsibilities and relationships.

2.1. Preamble

A DQMH module is the main component of an architecture based on DQMH® framework. A DQMH module is used to implement a section of the application that has one responsibility.

DQMH® framework defines two different type of DQMH module.

Singleton:

A Singleton DQMH module can have only one instance running at any given time.

Cloneable:

A Cloneable DQMH module can have one or multiple instances running in parallel.

DQMH® framework defines two different ways to carry data throughout the application and with both other DQMH modules and non-DQMH based code.

Request events:

A request is a code that fires an event requesting the DQMH module to do something. Multiple locations in the code can send events to the DQMH module.

Request events are many-to-one.

Requests are usually named using imperative tense.

Broadcast events:

A broadcast is a code that fires an event broadcasting that the DQMH module did something. Multiple Event Structures can register to handle the Broadcast Events.

Broadcast Events are one-to-many.

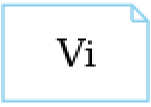
Broadcasts are usually named using past tense or passive voice.

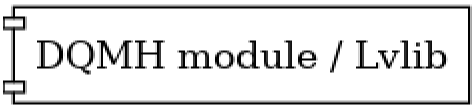
NOTE Refer to the DQMH® framework official documentation to find more details on how the framework works

The following section gives you details on the project architecture relying on this framework. It gives you an overview of the modules' interaction and detailed information on each module.

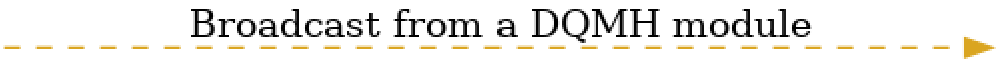
Graphs used in this section have the following legend:

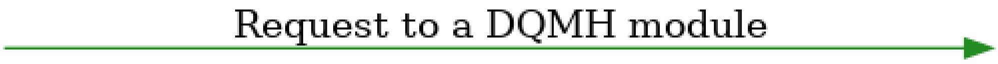
Components:

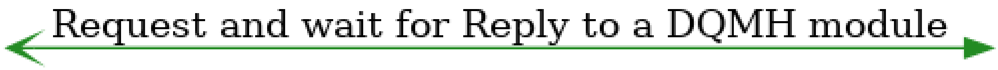




Events:



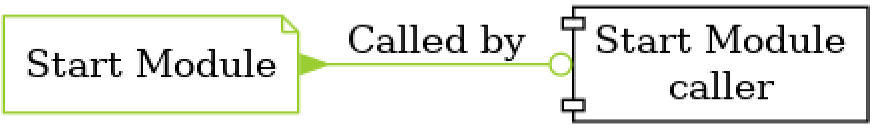


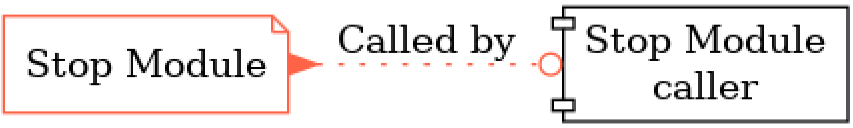


NOTE One arrow can represent one or more events between two components

NOTE Request and Request and wait for Reply are represented by only one arrow. If there is no Request and wait for Reply, Request representation is used. Otherwise Request and wait for Reply is used

Start and Stop module callers:





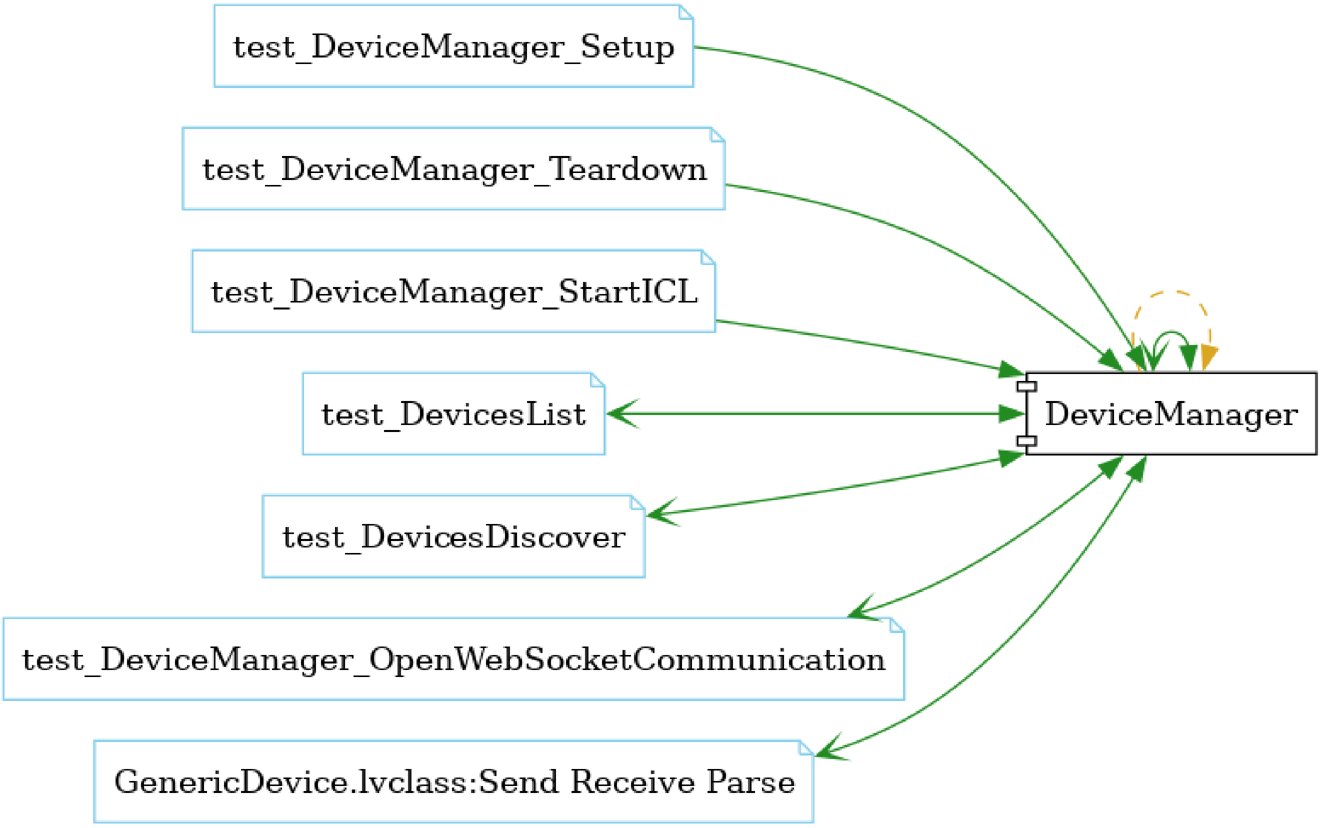
2.2. Modules overview

This project contains 1 singleton module and 0 cloneable module.

Table 1. Modules list

Singleton	Cloneable
DeviceManager.lvlib	

This graph represents the links between all DQMH modules.


test_DeviceManager_Setup
test_DeviceManager_Teardown
test_DeviceManager_StartICL
test_DevicesList
DeviceManager
test_DevicesDiscover
test_DeviceManager_OpenWebSocketCommunication
GenericDevice.lvclass:Send Receive Parse

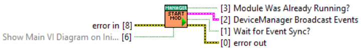
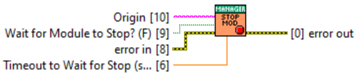
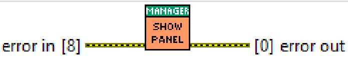
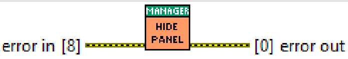
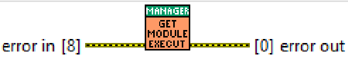
2.3. DeviceManager.lvlib

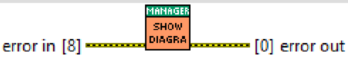
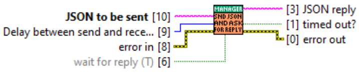
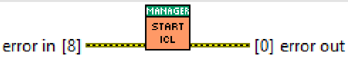
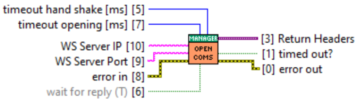
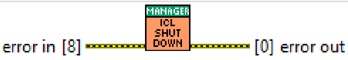
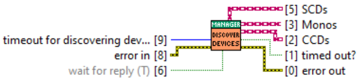
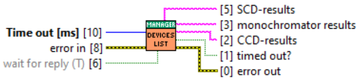
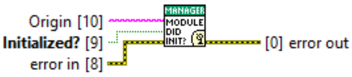
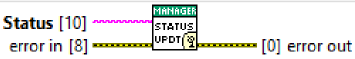
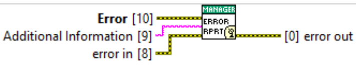
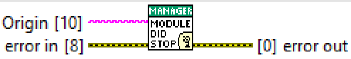
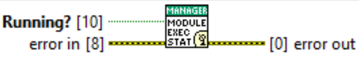
Type: Singleton

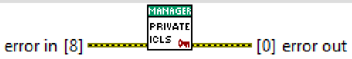
Responsibility: This DQMH module takes care of starting and stoping the ICL as well as monitoring any output from the ICL. It also handles any communication to and from the ICL via Websocket requests.

2.3.1. Event list

Table 2. Events

Name	Type	Connector pane	Description	S.	R.	I.
Start Module		 MANAGER START MOD error in [8] Show Main VI Diagram on Ini... [6] [3] Module Was Already Running? [2] DeviceManager Broadcast Events [1] Wait for Event Sync? [0] error out	Launches the module Main VI. After calling this VI, you can optionally register for broadcast events from the module by wiring the broadcast events output of this VI to a <b>Register For Events</b> function. After the optional Register For Events function call, you should always call the <b>Synchronize Module Events.vi</b> for this module with the 'Wait for Event Sync?' output of this VI to the corresponding input of the Synchronize Module Events.vi. To see an example of the proper wiring pattern, see the "Start Module: Value Change" event frame in the API Tester VI for this module.			
Stop Module		 MANAGER STOP MOD Origin [10] Wait for Module to Stop? (F) [9] error in [8] Timeout to Wait for Stop (s... [6] [0] error out	Send the Stop request to the Module's Main.vi. If <b>Wait for Module to Stop?</b> is TRUE, this VI will wait until the module main VI stops, and will timeout at the <b>Timeout to Wait for Stop</b> value. This value defaults to "-1", which means the VI will not timeout, and will always wait until the module main VI stops before completing execution. Note: The <b>Timeout to Wait for Stop</b> value is ignored if 'Wait for Module to Stop?' is set to FALSE.			
Show Panel	○→		Send the Show Panel request to the Module's Main.vi.			
Hide Panel	○→		Send the Hide Panel request to the Module's Main.vi.			
Get Module Execution Status	○→	 MANAGER GET MODULE EXECUT error in [8] [0] error out	Fire the Get Module Execution Status request.			

Name	Type	Connector pane	Description	S.	R.	I.
Show Diagram	○→		This VI tells the Module to show its block diagram to facilitate troubleshooting (add probes, breakpoints, highlight execution, etc).			
SendJSONandAskForReply	○→ ↻	 JSON to be sent [10] Delay between send and rece... [9] error in [8] wait for reply (T) [6] MANAGER SND JSON AND ASK FOR REPLY [3] JSON reply [1] timed out? [0] error out	This request asks in a synchronous manner to send a JSON string to the ICL, waits the specified time in [ms] and querries the websocket for a reply.			
StartICL	○→		This command starts the ICL.exe and its monitoring			
OpenWebSocketCommunication	○→ ↻	 timeout hand shake [ms] [5] timeout opening [ms] [7] WS Server IP [10] WS Server Port [9] error in [8] wait for reply (T) [6] MANAGER OPEN COMS [3] Return Headers [1] timed out? [0] error out	This event opens the websocket communcation from the DeviceManager to the ICL.exe			
ICLshutdown	○→		This request sends the command to shutdown the ICL.exe via websocket communication.			
DiscoverDevices	○→ ↻	 timeout for discovering dev... [9] error in [8] wait for reply (T) [6] MANAGER DISCOVER DEVICES [5] SCDs [3] Monos [2] CCDs [1] timed out? [0] error out	Requests from the ICL to discover monochromators, cameras and single channel detectors.			
DevicesList	○→ ↻	 Time out [ms] [10] error in [8] wait for reply (T) [6] MANAGER DEVICES LIST [5] SCD-results [3] monochromator results [2] CCD-results [1] timed out? [0] error out	This event calls mono_list, ccd_list, and scd_list.			
Module Did Init	📡	 Origin [10] Initialized? [9] error in [8] MANAGER MODULE DID INIT? [0] error out	Send the Module Did Init event to any VI registered to listen to this module's broadcast events.			
Status Updated	📡	 Status [10] error in [8] MANAGER STATUS UPDT [0] error out	Send the Status Updated event to any VI registered to listen to events from the owning module.			
Error Reported	📡	 Error [10] Additional Information [9] error in [8] MANAGER ERROR RPRT [0] error out	Send the Error Reported event to any VI registered to listen to events from the owning module.			
Module Did Stop	📡	 Origin [10] error in [8] MANAGER MODULE DID STOP [0] error out	Send the Module Did Stop event to any VI registered to listen to this module's broadcast events.			
Update Module Execution Status	📡	 Running? [10] error in [8] MANAGER MODULE EXEC STAT [0] error out	Broadcast event to specify whether or not the module is running.			

Name	Type	Connector pane	Description	S.	R.	I.
ICLstartNotification	○→		This private event is used to tell the ICLcommunication loop that the ICL is running and a communcation via websocket can be established			

Type: ○→ → Request | ⇄ → Request and Wait for Reply | 📡 → Broadcast

Scope: 🔑 → Protected | 🔑 → Community

Reentrancy: 🄿 → Preallocated reentrancy | 🅂 → Shared reentrancy

Inlining: ⇉ → Inlined

2.3.2. Module relationship

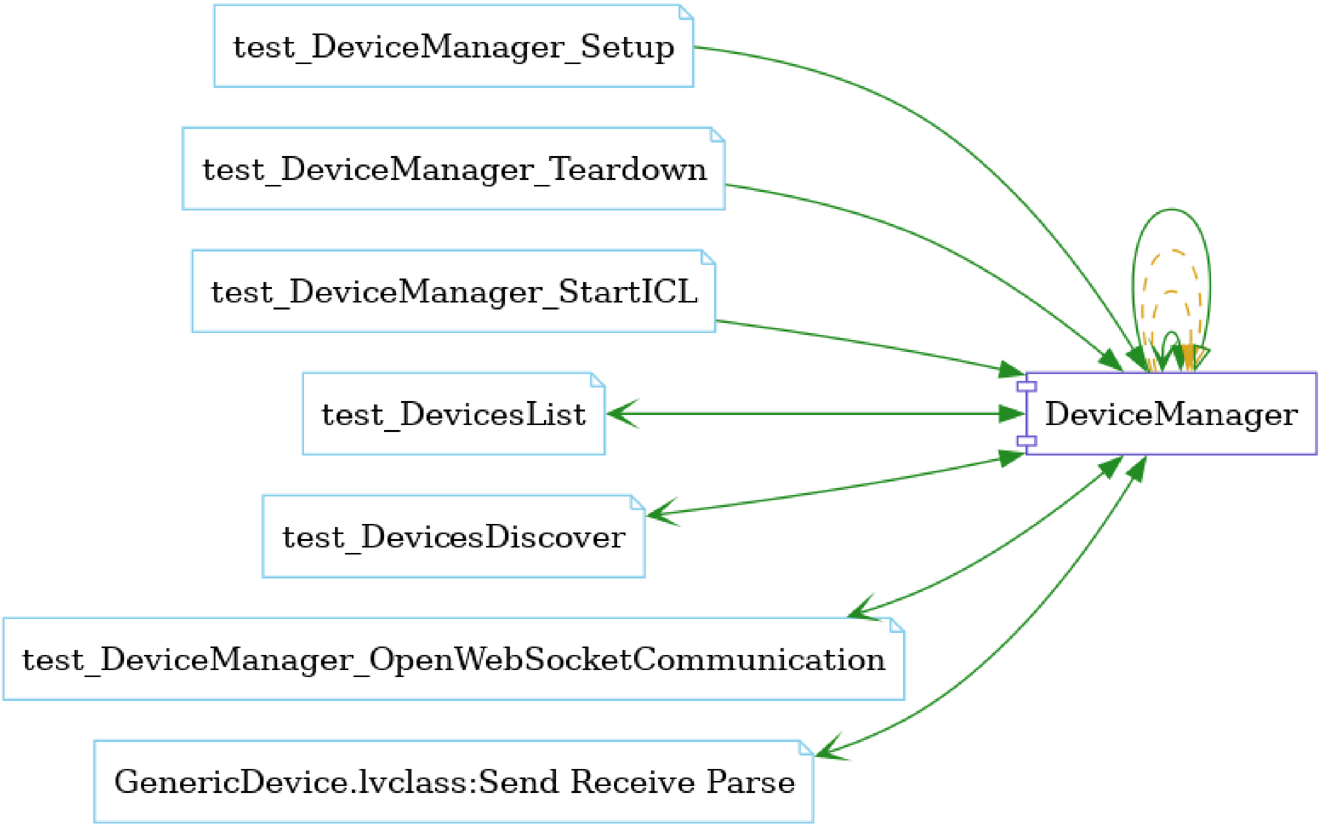

test_DeviceManager_Setup
test_DeviceManager_Teardown
test_DeviceManager_StartICL
test_DevicesList
test_DevicesDiscover
test_DeviceManager_OpenWebSocketCommunication
GenericDevice.lvclass:Send Receive Parse
DeviceManager

Table 3. Requests callers

Request Name	Callers
DevicesList	DeviceManager.lvlib:Test DeviceManager API.vi test_DevicesList.vi
DiscoverDevices	DeviceManager.lvlib:OpenConnectionWithDevice.vi DeviceManager.lvlib:Test DeviceManager API.vi test_DevicesDiscover.vi

Request Name	Callers
Get Module Execution Status	DeviceManager.lvlib:Obtain Broadcast Events for Registration.vi DeviceManager.lvlib:Start Module.vi
Hide Panel	DeviceManager.lvlib:Test DeviceManager API.vi
ICLshutdown	DeviceManager.lvlib:CloseConnectionWithDevice.vi DeviceManager.lvlib:Test DeviceManager API.vi
ICLstartNotification	DeviceManager.lvlib:Main.vi
OpenWebSocketCommunication	DeviceManager.lvlib:OpenConnectionWithDevice.vi DeviceManager.lvlib:Test DeviceManager API.vi test_DeviceManager_OpenWebSocketCommunication.vi
SendJSONandAskForReply	DeviceManager.lvlib:Test DeviceManager API.vi GenericDevice.lvclass:Send Receive Parse.vi
Show Diagram	DeviceManager.lvlib:Test DeviceManager API.vi
Show Panel	DeviceManager.lvlib:Test DeviceManager API.vi
StartICL	DeviceManager.lvlib:OpenConnectionWithDevice.vi DeviceManager.lvlib:Test DeviceManager API.vi test_DeviceManager_StartICL.vi

Table 4. Broadcasts Listeners

Broadcast Name	Listeners
Error Reported	DeviceManager.lvlib:Test DeviceManager API.vi
Module Did Init	DeviceManager.lvlib:Test DeviceManager API.vi
Module Did Stop	DeviceManager.lvlib:Test DeviceManager API.vi
Status Updated	DeviceManager.lvlib:Test DeviceManager API.vi
Update Module Execution Status	DeviceManager.lvlib:Test DeviceManager API.vi

Table 5. Used requests

Module	Requests
DeviceManager.lvlib	ICLstartNotification.vi Stop Module.vi

Table 6. Registered broadcast

Module	Broadcasts
DeviceManager.lvlib	Error Reported.vi Module Did Init.vi Module Did Stop.vi Status Updated.vi Update Module Execution Status.vi

2.3.3. Module Start/Stop calls

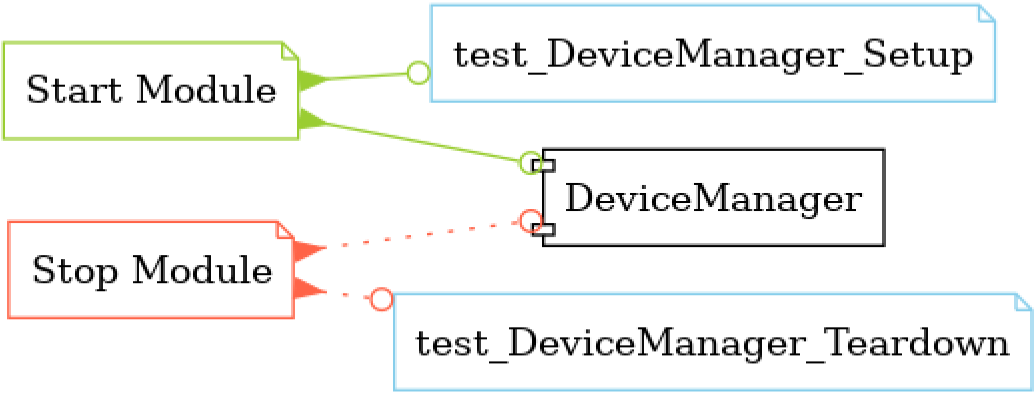


Table 7. Start and Stop module callers

Function	Callers
Start Module	DeviceManager.lvlib:OpenConnectionWithDevice.vi DeviceManager.lvlib:Test DeviceManager API.vi test_DeviceManager_Setup.vi
Stop Module	DeviceManager.lvlib:Handle Exit.vi DeviceManager.lvlib:CloseConnectionWithDevice.vi DeviceManager.lvlib:Test DeviceManager API.vi test_DeviceManager_Teardown.vi

2.3.4. Module custom errors

TIP Custom errors are added to the module via vi named `*--error.vi`.

Module DeviceManager.lvlib use the following custom errors:

Table 8. Custom errors

Name	Code	Description
Module Not Running	0	
Module Not Stopped	0	
Module Not Synced	0	

Name	Code	Description
Request and Wait for Reply Timeout	0	

Chapter 3. Classes

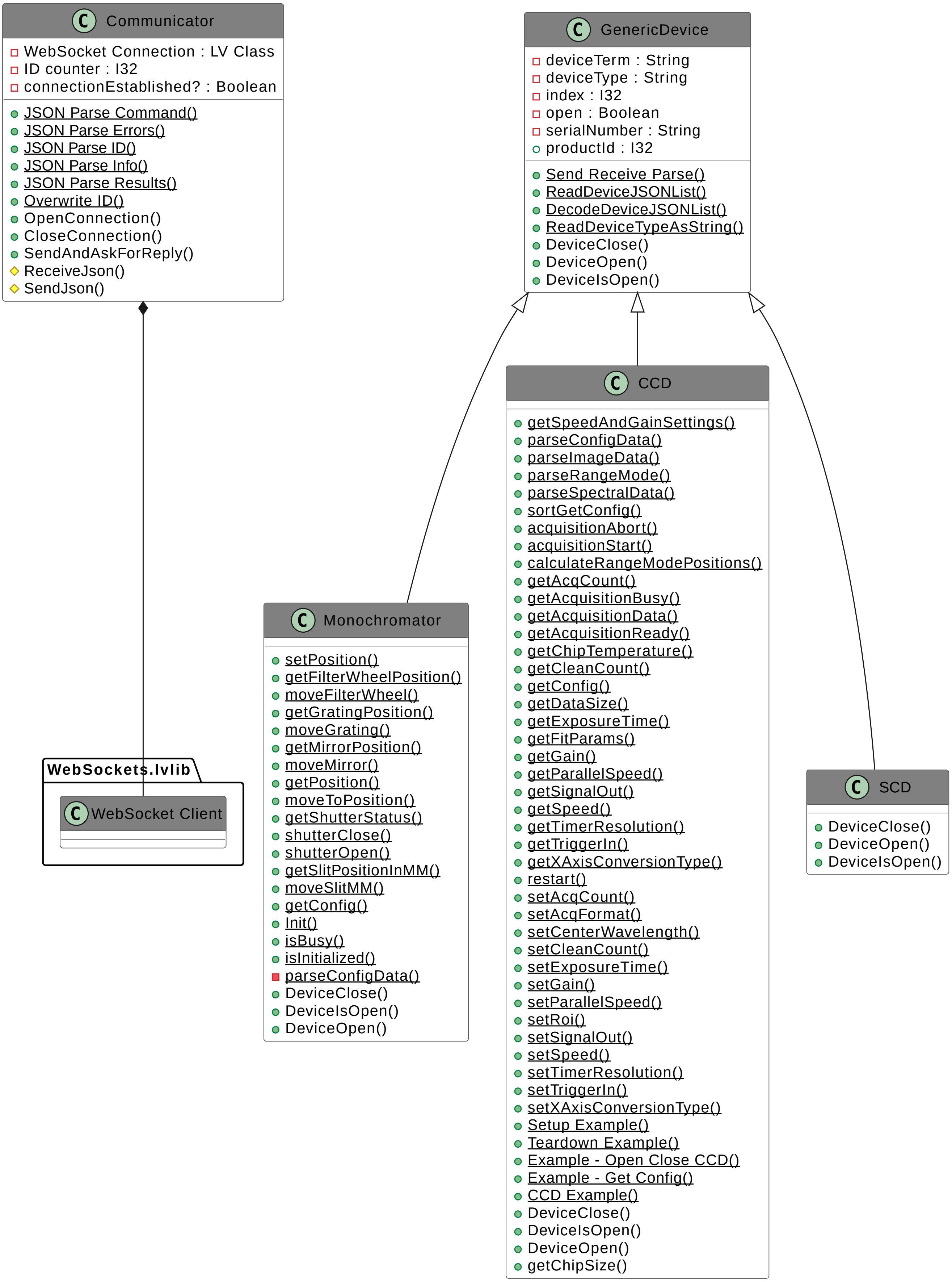
This section describes the classes contained in the project.

3.1. Classes overview

This project contains 5 classes and 0 interface.

Table 9. Classes list

Classes	Interfaces
Communicator.lvclass	
GenericDevice.lvclass	
Monochromator.lvclass	
CCD.lvclass	
SCD.lvclass	



3.2. Communicator.lvclass

Responsibility: This class handles the communication between LV and the ICL via websocket requests.

Version: 1.0.0.4

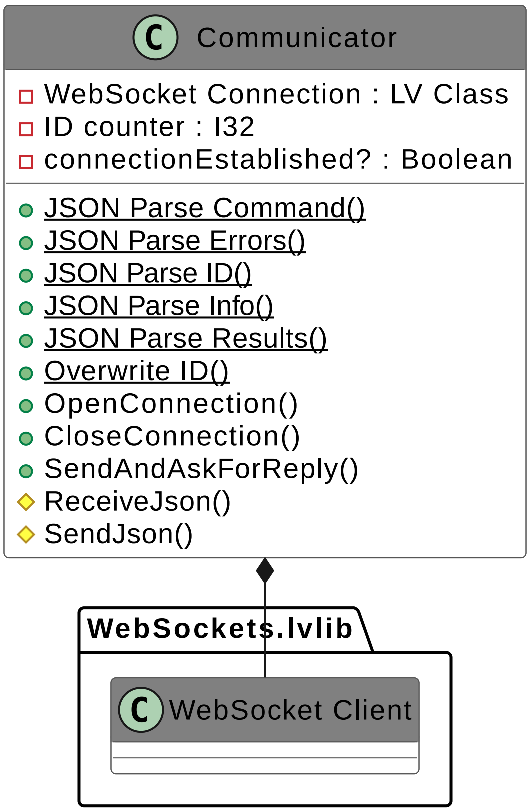
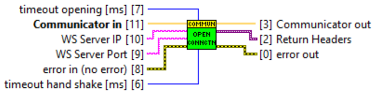
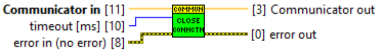
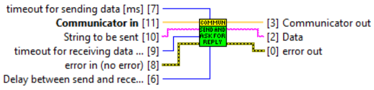
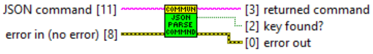
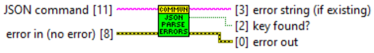
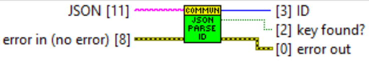
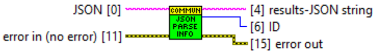
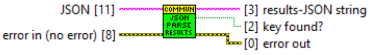
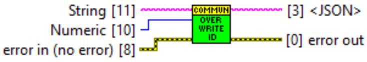
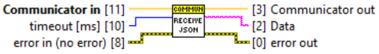
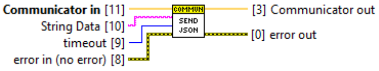


Table 10. Functions (non private scope only)

Name	Connector pane	Description	S.	R.	I.
OpenConnection	 timeout opening [ms] [7] Communicator in [11] WS Server IP [10] WS Server Port [9] error in (no error) [8] timeout hand shake [ms] [6] COMMUN OPEN CONNCTN [3] Communicator out [2] Return Headers [0] error out	Opens a connection to the ICL via werbsocket.			
CloseConnection	 Communicator in [11] timeout [ms] [10] error in (no error) [8] COMMUN CLOSE CONNCTN [3] Communicator out [0] error out	Closes the websocket connection.			
SendAndAskForReply	 timeout for sending data [ms] [7] Communicator in [11] String to be sent [10] timeout for receiving data ... [9] error in (no error) [8] Delay between send and rece... [6] COMMUN SEND AND ASK FOR REPLY [3] Communicator out [2] Data [0] error out	Wrapper around send and receive for the websocket communication.			
JSON Parse Command	 JSON command [11] error in (no error) [8] COMMUN JSON PARSE COMMND [3] returned command [2] key found? [0] error out	Parses the original command.			
JSON Parse Errors	 JSON command [11] error in (no error) [8] COMMUN JSON PARSE ERRORS [3] error string (if existing) [2] key found? [0] error out	Parses the returned error into an LV error.			
JSON Parse ID	 JSON [11] error in (no error) [8] COMMUN JSON PARSE ID [3] ID [2] key found? [0] error out	Parses the message ID to follow communication.			
JSON Parse Info	 JSON [0] error in (no error) [11] COMMUN JSON PARSE INFO [4] results-JSON string [6] ID [15] error out	Wrapper around all other VIs that parse an answer from the ICL.			
JSON Parse Results	 JSON [11] error in (no error) [8] COMMUN JSON PARSE RESULTS [3] results-JSON string [2] key found? [0] error out	Parses the results from the ICL.			
Overwrite ID	 String [11] Numeric [10] error in (no error) [8] COMMUN OVER WRITE ID [3] <JSON> [0] error out	Overwrites the message ID if a custom ID is to be used for messaging to the ICL.			

Name	Connector pane	Description	S.	R.	I.
ReceiveJson	 Communicator in [11] timeout [ms] [10] error in (no error) [8] COMMUN RECEIVE JSON [3] Communicator out [2] Data [0] error out	Receives the reply from the synchronous communication to the ICL.	🔑		
SendJson	 Communicator in [11] String Data [10] timeout [9] error in (no error) [8] COMMUN SEND JSON [3] Communicator out [0] error out	Sends the request for a synchronous communication to the ICL.	🔑		

Scope: 🔑 → Protected | 🔑 → Community

Reentrancy: P → Preallocated reentrancy | S → Shared reentrancy

Inlining: ⇢ → Inlined

3.3. GenericDevice.lvclass

Responsibility: This parent class for all devices provides base functionality like discovering, opening and closing devices on the ICL.

Version: 1.0.0.7

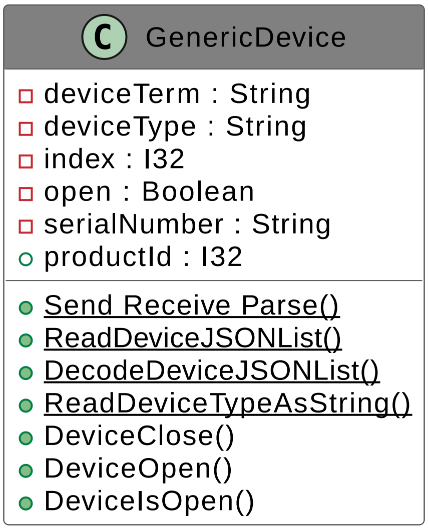
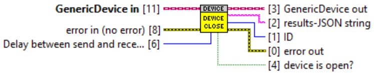
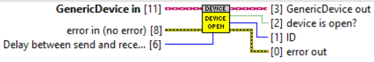
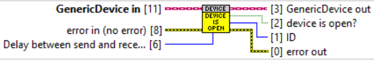
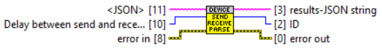
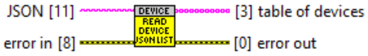
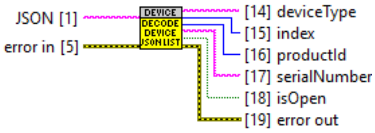
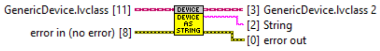
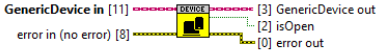
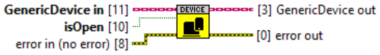
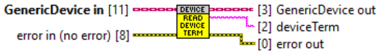
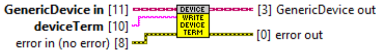
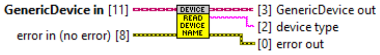
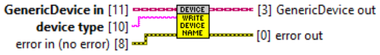
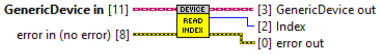
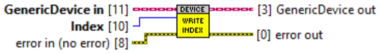
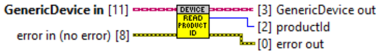
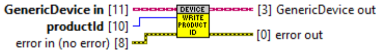
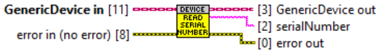
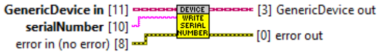


Table 11. Functions (non private scope only)

Name	Connector pane	Description	S.	R.	I.
DeviceClose	 GenericDevice in [11] error in (no error) [8] Delay between send and rece... [6] DEVICE DEVICE CLOSE [3] GenericDevice out [2] results-JSON string [1] ID [0] error out [4] device is open?	Closes communications with the CCD indicated by the index.			
DeviceOpen	 GenericDevice in [11] error in (no error) [8] Delay between send and rece... [6] DEVICE DEVICE OPEN [3] GenericDevice out [2] device is open? [1] ID [0] error out	This command initializes the CCD and gets it’s the CCD configuration from the device. The device is also connected to the API. Since a CCD hardware initialization occurs, all CCD parameters, including any previously set parameters, will be reset to their default values.			
DeviceIsOpen	 GenericDevice in [11] error in (no error) [8] Delay between send and rece... [6] DEVICE DEVICE IS OPEN [3] GenericDevice out [2] device is open? [1] ID [0] error out	Returns true if selected CCD is open.			

Name	Connector pane	Description	S.	R.	I.
Send Receive Parse	 <JSON> [11] Delay between send and rece... [10] error in [8] DEVICE SEND RECEIVE PARSE [3] results-JSON string [2] ID [0] error out	Wrapper around the send and receive calls.			
ReadDeviceJSONList	 JSON [11] error in [8] DEVICE READ DEVICE JSON LIST [3] table of devices [0] error out	Parses the returned devies into a string array.			
DecodeDeviceJSONList	 JSON [1] error in [5] DEVICE DECODE DEVICE JSON LIST [14] deviceType [15] index [16] productId [17] serialNumber [18] isOpen [19] error out	Decodes the answer from the ICL into a device.			
ReadDeviceTypeAsString	 GenericDevice.lvclass [11] error in (no error) [8] DEVICE DEVICE AS STRING [3] GenericDevice.lvclass 2 [2] String [0] error out	Return the device class name as string.			
Read isOpen	 GenericDevice in [11] error in (no error) [8] DEVICE [3] GenericDevice out [2] isOpen [0] error out	Accessor VI for this class property.			
Write isOpen	 GenericDevice in [11] isOpen [10] error in (no error) [8] DEVICE [3] GenericDevice out [0] error out	Accessor VI for this class property.			
Read DeviceTerm	 GenericDevice in [11] error in (no error) [8] DEVICE READ DEVICE TERM [3] GenericDevice out [2] deviceTerm [0] error out	Accessor VI for this class property.			
Write DeviceTerm	 GenericDevice in [11] deviceTerm [10] error in (no error) [8] DEVICE WRITE DEVICE TERM [3] GenericDevice out [0] error out	Accessor VI for this class property.			
Read DeviceType	 GenericDevice in [11] error in (no error) [8] DEVICE READ DEVICE NAME [3] GenericDevice out [2] device type [0] error out	Accessor VI for this class property.			
Write DeviceType	 GenericDevice in [11] device type [10] error in (no error) [8] DEVICE WRITE DEVICE NAME [3] GenericDevice out [0] error out	Accessor VI for this class property.			
Read Index	 GenericDevice in [11] error in (no error) [8] DEVICE READ INDEX [3] GenericDevice out [2] Index [0] error out	Accessor VI for this class property.			
Write Index	 GenericDevice in [11] Index [10] error in (no error) [8] DEVICE WRITE INDEX [3] GenericDevice out [0] error out	Accessor VI for this class property.			
Read productId	 GenericDevice in [11] error in (no error) [8] DEVICE READ PRODUCT ID [3] GenericDevice out [2] productId [0] error out	Accesor to the product ID.s			
Write productId	 GenericDevice in [11] productId [10] error in (no error) [8] DEVICE WRITE PRODUCT ID [3] GenericDevice out [0] error out	Accessor to the product ID.			
Read SerialNumber	 GenericDevice in [11] error in (no error) [8] DEVICE READ SERIAL NUMBER [3] GenericDevice out [2] serialNumber [0] error out	Accessor VI for this class property.			
Write SerialNumber	 GenericDevice in [11] serialNumber [10] error in (no error) [8] DEVICE WRITE SERIAL NUMBER [3] GenericDevice out [0] error out	Accessor VI for this class property.			

Scope: 🔑 → Protected | 🔑 → Community

Reentrancy: 🄿 → Preallocated reentrancy | 🅂 → Shared reentrancy

Inlining: ⇉ → Inlined

3.4. Monochromator.lvclass

Responsibility: This class contains all functionality needed for Horiba's Monochromators.

Version: 1.0.0.1

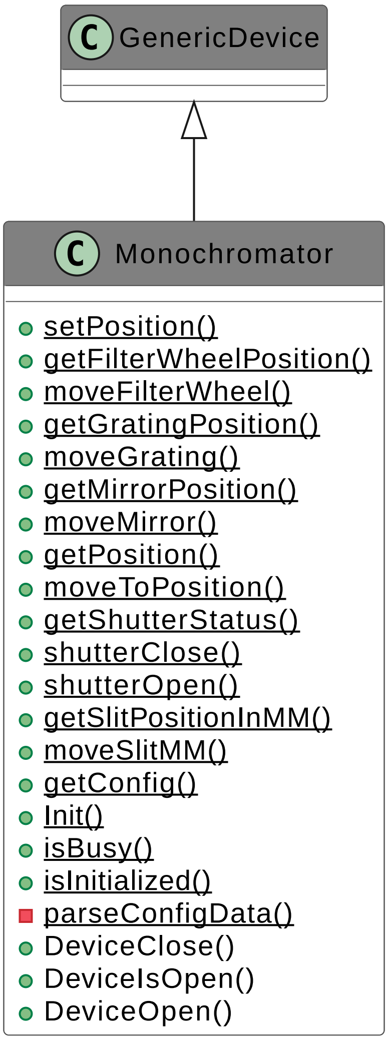
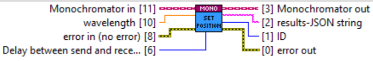
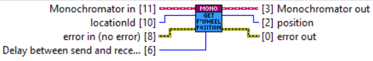
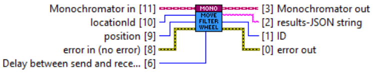
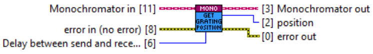
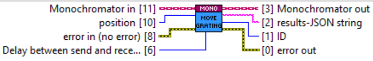
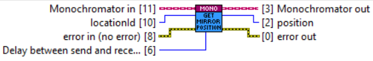
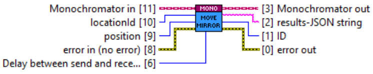
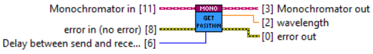
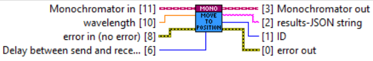
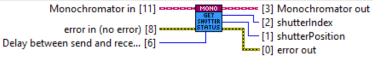
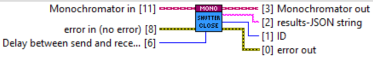
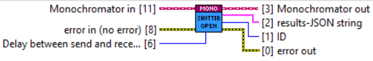
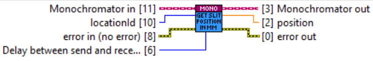
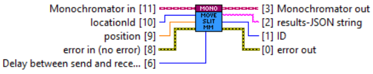
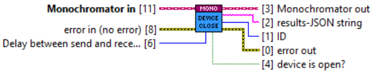
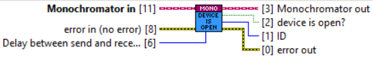
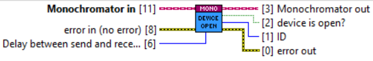
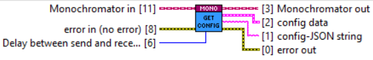


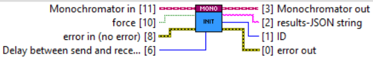
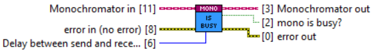
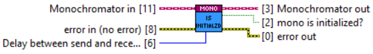
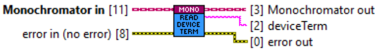
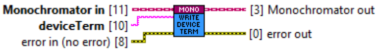
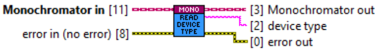
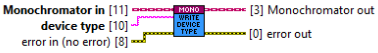
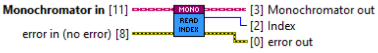
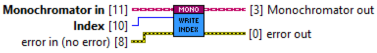
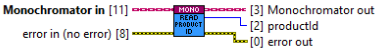
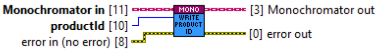
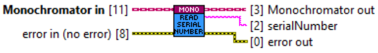
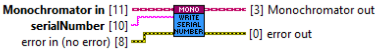
Table 12. Functions (non private scope only)

Name	Connector pane	Description	S.	R.	I.
setPosition	 Monochromator in [11] wavelength [10] error in (no error) [8] Delay between send and rece... [6] MONO SET POSITION [3] Monochromator out [2] results-JSON string [1] ID [0] error out	!!! Attention: this VI can potentially uncalibrate your Mono !!! !!! Only use after reading the description !!! !!! Use moveToPosition to move the mono to a different wavelength !!! This command sets the wavelength value of the current grating position of the monochromator. This could potentially un-calibrate the monochromator and report an incorrect wavelength compared to the actual output wavelength. wavelength Float. Set the wavelength of the mono at the current position.			
getFilterWheelPosition	 Monochromator in [11] locationId [10] error in (no error) [8] Delay between send and rece... [6] MONO [3] Monochromator out [2] position [0] error out	Returns the current filter wheel position. parameter description index Integer. Used to identify which mono to control. See mono_list command locationId Integer. Specifies the filter wheel location. 0 = Filter wheel 1 (Internal) 1 = Filter wheel 2 (External)			
moveFilterWheel	 Monochromator in [11] locationId [10] position [9] error in (no error) [8] Delay between send and rece... [6] MONO MOVE FILTER WHEEL [3] Monochromator out [2] results-JSON string [1] ID [0] error out	Move the filter wheel to a position. locationId Integer. Specifies which filter wheel to move. 0 = Filter wheel 1 (Internal) 1 = Filter wheel 2 (External) position Integer. Position to move the filter wheel.			
getGratingPosition	 Monochromator in [11] error in (no error) [8] Delay between send and rece... [6] MONO GET GRATING POSITION [3] Monochromator out [2] position [0] error out	Returns the current grating turret position. Note: Prior to the initialization of the grating turret, this value may not reflect the actual position of the turret. To read the current position of the grating turret, please run mono_init prior to running this command.			

Name	Connector pane	Description	S.	R.	I.
moveGrating	 Monochromator in [11] position [10] error in (no error) [8] Delay between send and rece... [6] MONO MOVE GRATING [3] Monochromator out [2] results-JSON string [1] ID [0] error out	Moves the grating turret to the specified position. Note: The turret sensor does not re-read the position each time it is moved, therefore the position may not be accurate prior to initialization. See note for mono_getGratingPosition.			
getMirrorPosition	 Monochromator in [11] locationId [10] error in (no error) [8] Delay between send and rece... [6] MONO GET MIRROR POSITION [3] Monochromator out [2] position [0] error out	Returns the position of the specified mirror. parameter description index Integer. Used to identify which mono to control. See mono_list command locationId Integer. Identifies which mirror to get the position from. 0 = Mirror 1 (Entrance) 1 = Mirror 2 (Exit)			
moveMirror	 Monochromator in [11] locationId [10] position [9] error in (no error) [8] Delay between send and rece... [6] MONO MOVE MIRROR [3] Monochromator out [2] results-JSON string [1] ID [0] error out	Moves the specified mirror to a position. locationId Integer. Identifies which mirror to move (zero-based). 0 = Mirror 1 (Entrance) 1 = Mirror 2 (Exit) position Integer. Position to move to. 0 = Axial 1 = Lateral			
getPosition	 Monochromator in [11] error in (no error) [8] Delay between send and rece... [6] MONO GET POSITION [3] Monochromator out [2] wavelength [0] error out	Returns the wavelength value, in nm, of the monochromator's current position.			
moveToPosition	 Monochromator in [11] wavelength [10] error in (no error) [8] Delay between send and rece... [6] MONO MOVE TO POSITION [3] Monochromator out [2] results-JSON string [1] ID [0] error out	This command starts the monochromator moving to the requested wavelength in nm. This is an asynchronous command. Use the mono_isBusy command to know when the move has completed.			
getShutterStatus	 Monochromator in [11] error in (no error) [8] Delay between send and rece... [6] MONO GET SHUTTER STATUS [3] Monochromator out [2] shutterIndex [1] shutterPosition [0] error out	Returns the status of the currently selected shutter. Note: To view the status of the shutter solenoid the device must be configured for internal shutter mode. locationId Integer. Identifies the currently selected shutter. 0 = Shutter 1 (Front shutter) 1 = Shutter 2 (Side shutter) position Integer. Shutter position status. 0 = Closed 1 = Open			

Name	Connector pane	Description	S.	R.	I.
shutterClose	 Monochromator in [11] error in (no error) [8] Delay between send and rece... [6] MONO SHUTTER CLOSE [3] Monochromator out [2] results-JSON string [1] ID [0] error out	Deactivates the currently selected shutter solenoid. Note: The device must be configured for internal shutter mode. The shutter solenoid will not respond in External (Bypass) mode.			
shutterOpen	 Monochromator in [11] error in (no error) [8] Delay between send and rece... [6] MONO SHUTTER OPEN [3] Monochromator out [2] results-JSON string [1] ID [0] error out	Activates the currently selected shutter solenoid. Note: The device must be configured for internal shutter mode. The shutter solenoid will not respond in External (Bypass) mode.			
getSlitPositionInMM	 Monochromator in [11] locationId [10] error in (no error) [8] Delay between send and rece... [6] MONO GET SLIT POSITION IN MM [3] Monochromator out [2] position [0] error out	Returns the position of the specified slit in millimeters. The location id of each configured slit can be found under the ports section of the mono configuration. See mono_getConfig for additional information. For example: "ports": [{ "locationId": 1, "slitType": 1 }, { "locationId": 2, "slitType": 1 }, { "locationId": 4, "slitType": 1 }] Note: The "locationId" parameter found in the mono configuration is 1-based. However, the mono_getSlitPositionInMM command uses a 0-based "locationId".			

Name	Connector pane	Description	S.	R.	I.
moveSlitMM	 Monochromator in [11] locationId [10] position [9] error in (no error) [8] Delay between send and rece... [6] MONO MOVE SLIT MM [3] Monochromator out [2] results-JSON string [1] ID [0] error out	Moves the specified slit to the position in millimeters. The location id of each configured slit can be found under the ports section of the mono configuration. See mono_getConfig for additional information. For example: "ports": [{ "locationId": 1, "slitType": 1 }, { "locationId": 2, "slitType": 1 }, { "locationId": 4, "slitType": 1 }] Note: The "locationId" parameter found in the mono configuration is 1-based. However, the mono_moveSlitMM command uses a 0-based "locationId". locationId Integer. Slit location (zero-based) position Float. Position in millimeters			
DeviceClose	 Monochromator in [11] error in (no error) [8] Delay between send and rece... [6] MONO DEVICE CLOSE [3] Monochromator out [2] results-JSON string [1] ID [0] error out [4] device is open?	Closes communications with the monochromator indicated by the index.			
DeviceIsOpen	 Monochromator in [11] error in (no error) [8] Delay between send and rece... [6] MONO DEVICE IS OPEN [3] Monochromator out [2] device is open? [1] ID [0] error out	Returns true if selected monochromator is open.			
DeviceOpen	 Monochromator in [11] error in (no error) [8] Delay between send and rece... [6] MONO DEVICE OPEN [3] Monochromator out [2] device is open? [1] ID [0] error out	Opens communications with the monochromator indicated by the index command parameter.			
getConfig	 Monochromator in [11] error in (no error) [8] Delay between send and rece... [6] MONO GET CONFIG [3] Monochromator out [2] config data [1] config-JSON string [0] error out	This command returns the monochromator configuration. Port Descriptions: locationId Integer. Used to identify the slit location. 1 = Front entrance (axial) 2 = Side entrance (lateral) 3 = Front exit (axial) 4 = Side exit (lateral) slitType Integer. Used to identify the slit size. 1 = 2mm slit 2 = 7mm slit			

Name	Connector pane	Description	S.	R.	I.
Init	 Monochromator in [11] force [10] error in (no error) [8] Delay between send and rece... [6] MONO INIT [3] Monochromator out [2] results-JSON string [1] ID [0] error out	Starts the monochromator initialization process (homing...). This is a "long-running" asynchronous command. Use the mono_isBusy command to know when initialization has completed. force Boolean. Force starts the initialization process.			
isBusy	 Monochromator in [11] error in (no error) [8] Delay between send and rece... [6] MONO IS BUSY [3] Monochromator out [2] mono is busy? [0] error out	Returns true if selected monochromator is busy.			
isInitialized	 Monochromator in [11] error in (no error) [8] Delay between send and rece... [6] MONO IS INITIALZD [3] Monochromator out [2] mono is initialized? [0] error out	This command returns true when the mono is initialized. Otherwise it returns false. Note: This command may also return false when the mono is busy with another command.			
Read DeviceTerm	 Monochromator in [11] error in (no error) [8] MONO READ DEVICE TERM [3] Monochromator out [2] deviceTerm [0] error out	Accessor VI for this class property.			
Write DeviceTerm	 Monochromator in [11] deviceTerm [10] error in (no error) [8] MONO WRITE DEVICE TERM [3] Monochromator out [0] error out	Accessor VI for this class property.			
Read DeviceType	 Monochromator in [11] error in (no error) [8] MONO READ DEVICE TYPE [3] Monochromator out [2] device type [0] error out	Accessor VI for this class property.			
Write DeviceType	 Monochromator in [11] device type [10] error in (no error) [8] MONO WRITE DEVICE TYPE [3] Monochromator out [0] error out	Accessor VI for this class property.			
Read Index	 Monochromator in [11] error in (no error) [8] MONO READ INDEX [3] Monochromator out [2] Index [0] error out	Accessor VI for this class property.			
Write Index	 Monochromator in [11] Index [10] error in (no error) [8] MONO WRITE INDEX [3] Monochromator out [0] error out	Accessor VI for this class property.			
Read productId	 Monochromator in [11] error in (no error) [8] MONO READ PRODUCT ID [3] Monochromator out [2] productId [0] error out	Accessor VI for this class property.			
Write productId	 Monochromator in [11] productId [10] error in (no error) [8] MONO WRITE PRODUCT ID [3] Monochromator out [0] error out	Accessor VI for this class property.			
Read serialNumber	 Monochromator in [11] error in (no error) [8] MONO READ SERIAL NUMBER [3] Monochromator out [2] serialNumber [0] error out	Accessor VI for this class property.			
Write serialNumber	 Monochromator in [11] serialNumber [10] error in (no error) [8] MONO WRITE SERIAL NUMBER [3] Monochromator out [0] error out	Accessor VI for this class property.			

Scope: → Protected | → Community

Reentrancy: → Preallocated reentrancy | → Shared reentrancy

Inlining: → Inlined

3.5. CCD.lvclass

Responsibility: This class contains all functionality needed for Horiba's Multi-Channel-detectors.

Version: 1.0.0.3

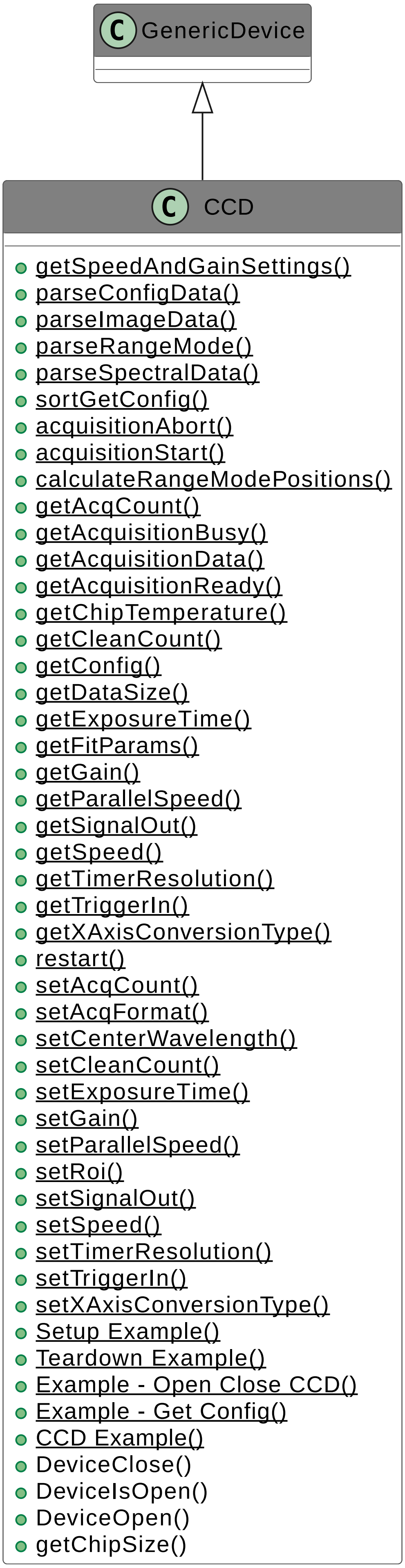
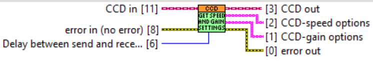
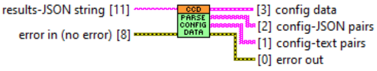
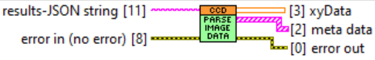
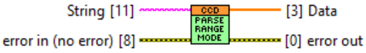
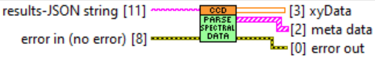
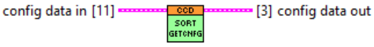
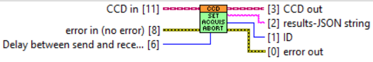
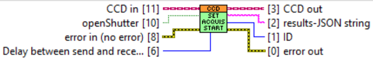
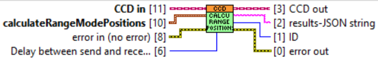
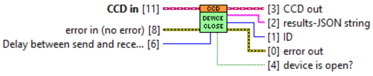
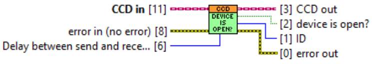
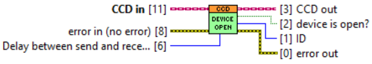
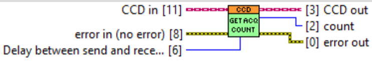
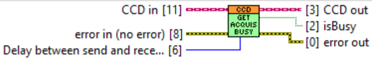
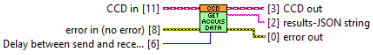
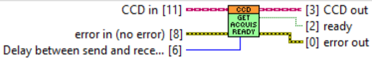
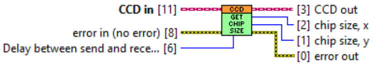
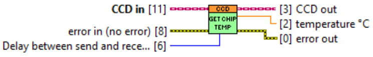
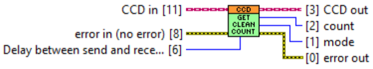
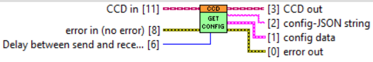


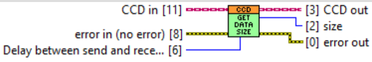
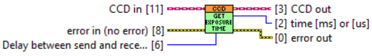
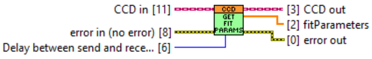
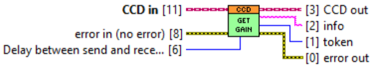
Table 13. Functions (non private scope only)

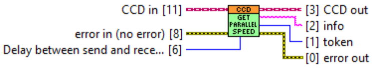
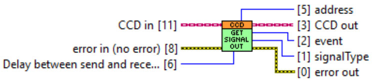
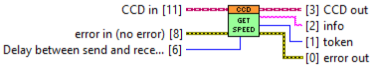
Name	Connector pane	Description	S.	R.	I.
getSpeedAndGainSettings	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] [3] CCD out [2] CCD-speed options [1] CCD-gain options [0] error out	Wrapper around the getSpeed and getGain VIs for convenience.			
parseConfigData	 results-JSON string [11] error in (no error) [8] [3] config data [2] config-JSON pairs [1] config-text pairs [0] error out	Helper VI to parse the configuration string to a LV cluster.			
parseImageData	 results-JSON string [11] error in (no error) [8] [3] xyData [2] meta data [0] error out	Helper VI to parse the ccd data to a LV 2D image array.			
parseRangeMode	 String [11] error in (no error) [8] [3] Data [0] error out	No description found (add content in vi description)			
parseSpectralData	 results-JSON string [11] error in (no error) [8] [3] xyData [2] meta data [0] error out	Helper VI to parse the ccd data to a LV 1D image array and 1d Spectral information arry.			
sortGetConfig		Helper VI to sort the config data.			
acquisitionAbort	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] [3] CCD out [2] results-JSON string [1] ID [0] error out	Stops the current acquisition.			
acquisitionStart	 CCD in [11] openShutter [10] error in (no error) [8] Delay between send and rece... [6] [3] CCD out [2] results-JSON string [1] ID [0] error out	tarts an acquisition that has been set up according to the previously defined acquisition parameters. Note: To specify the acquisition parameters please see ccd_setROI and ccd_setXAxisConversionType. If there are no acquisition parameters specified at the time of acquisition it may result in no data being generated.			
calculateRangeModePositions	 CCD in [11] calculateRangeModePositions [10] error in (no error) [8] Delay between send and rece... [6] [3] CCD out [2] results-JSON string [1] ID [0] error out	tarts an acquisition that has been set up according to the previously defined acquisition parameters. Note: To specify the acquisition parameters please see ccd_setROI and ccd_setXAxisConversionType. If there are no acquisition parameters specified at the time of acquisition it may result in no data being generated.			
DeviceClose	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] [3] CCD out [2] results-JSON string [1] ID [0] error out [4] device is open?	Closes the connection to the CCD device.			
DeviceIsOpen	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] [3] CCD out [2] device is open? [1] ID [0] error out	Checks if connection to the device is open or not.			
DeviceOpen	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] [3] CCD out [2] device is open? [1] ID [0] error out	Opens the connection to the CCD device.			

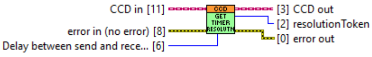
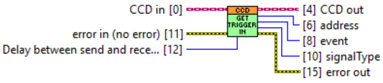
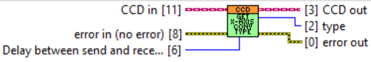
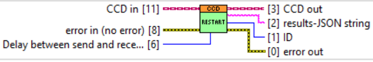
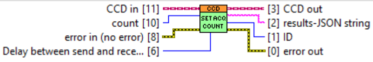
Name	Connector pane	Description	S.	R.	I.
getAcqCount	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET ACQ COUNT [3] CCD out [2] count [0] error out	Gets the number of acquisition measurements to be perform sequentially by the hardware. Return Results: results description count Integer. The number of acquisition measurements to be performed.			
getAcquisitionBusy	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET ACQUIS BUSY [3] CCD out [2] isBusy [0] error out	No description found (add content in vi description)			
getAcquisitionData	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET ACQUIS DATA [3] CCD out [2] results-JSON string [0] error out	The acquisition description string consists of the following information: acqIndex: Acquisition number roiIndex: Region of Interest number xOrigin: ROI's X Origin yOrigin: ROI's Y Origin xSize: ROI's X Size ySize: ROI's Y Size xBinning: ROI's X Bin yBinning: ROI's Y Bin Timestamp: This is a timestamp that relates to the time when the all the programmed acquisitions have completed. The data from all programmed acquisitions are retrieved from the CCD after all acquisitions have completed, therefore the same timestamp is used for all acquisitions. Command Parameters: Return Results: results description acquisition String. Acquisition data. Example command: Example response: { "command": "ccd_getAcquisitionData", "errors": [], "id": 1234, "results": { "acquisition": [{ "acqIndex": 1, "roi": [{ "roiIndex": 1, "xBinning": 1, "xOrigin": 1, "xSize": 8, "xyData": [[885.6389770507812, 976], [w 885.2899780273438, 975], [884.9409790039062, 979], [884.593017578125, 976], }			
getAcquisitionReady	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET ACQUIS READY [3] CCD out [2] ready [0] error out	No description found (add content in vi description)			

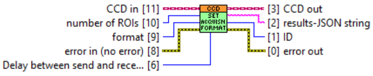
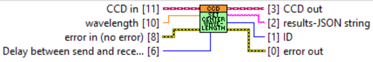
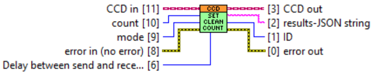
Name	Connector pane	Description	S.	R.	I.
getChipSize	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET CHIP SIZE [3] CCD out [2] chip size, x [1] chip size, y [0] error out	Returns the chip sensor’s pixel width and height size. Return Results: results description x Integer. Chip sensor's x size in pixels (width) y Integer. Chip sensor's y size in pixels (height)			
getChipTemperature	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET CHIP TEMP [3] CCD out [2] temperature °C [0] error out	Returns the temperature of the chip sensor in degrees C. Return Results: temperature Float. Chip sensor temperature in degrees C.			
getCleanCount	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET CLEAN COUNT [3] CCD out [2] count [1] mode [0] error out	Gets the number of cleans to be performed prior to measurement. results description count Integer. Number of cleans. mode Integer. Specifies how the cleans will be performed. 0 = Never 1 = First Only 2 = Between Only 3 = Each			

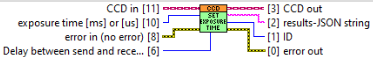
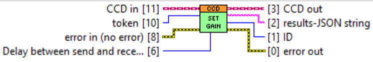
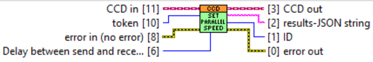
Name	Connector pane	Description	S.	R.	I.
getConfig	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET CONFIG [3] CCD out [2] config-JSON string [1] config data [0] error out	Returns the CCD device configuration. results description configuration JSON. CCD device configuration. xample response: { "command": "ccd_getConfig", "errors": [], "id": 1234, "results": { "configuration": { "chipHSpacing": "140", "chipHeight": "70", "chipName": "S10420", "chipSerialNumber": "FAH23 098", "chipVSpacing": "140", "chipWidth": "2048", "deviceType": "HORIBA Scientific Syncerity", "fitParameters": [0, 1, 0, 0, 0], "gains": [{ "info": "Best Dynamic Range", "token": 1 }, { "info": "High Sensitivity", "token": 2 }, { "info": "High Light", "token": 0 }], "hardwareAvgAvailable": false, "lineScan": false, "parallelSpeeds": [{ "info": "9.6 µSec", "token": 1 }, { "info": "4.9 µSec", "token": 2 }, { "info": "19 µSec", "token": 0 }], "productId": "13", "serialNumber": "Camera SN: 5128", "signals": [{ "events": [{ "name": "Ready For Trigger", "token": 1, "types": [{ "name": "TTL Active Low", "token": 1 }, { "name": "TTL Active High", "token": 0 }] }, { "name": "Not Readout", "token": 2, "types": [{ "name": "TTL Active Low", "token": 1 }, { "name": "TTL Active High", "token": 0 }] }, { "name": "Shutter Open", "token": 3, "types": [{ "name": "TTL Active Low", "token": 1 }, { "name": "TTL Active High", "token": 0 }] }, { "name": "Start Experiment", "token": 0, "types": [{ "name": "TTL Active Low", "token": 1 }, { "name": "TTL Active High", "token": 0 }] }], "name": "Signal Output", "token": 0 }], "speeds": [{ "info": "500 kHz ", "token": 1 }, { "info": "500 kHz Ultra", "token": 2 }, { "info": "500 kHz Wrap", "token": 127 }, { "info": " 45 kHz ", "token": 0 }], "supportedFeatures": { "cf_3PositionSlit": false, "cf_CMOSOffsetCorrection": false, "cf_Cleaning": true, "cf_DSP": false, "cf_DSPBin2X": false, "cf_DelayAfterTrigger": false, "cf_Delays":			

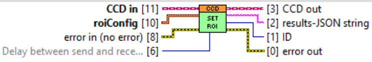
Name	Connector pane	Description	S.	R.	I.
getDataSize	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET DATA SIZE [3] CCD out [2] size [0] error out	Gets the number of pixels to be returned based on the current settings. results description size Integer. Byte data size for all ROIs and acquisitions.			
getExposureTime	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET EXPOSURE TIME [3] CCD out [2] time [ms] or [us] [0] error out	Gets the exposure time (expressed in Timer Resolution units). Note: To check the current Timer Resolution value see ccd_getTimerResolution. Alternatively the Timer Resolution value can be set using ccd_setTimerResolution. Example: If Exposure Time is set to 50, and the Timer Resolution value is 1000, the CCD exposure time (integration time) = 50 milliseconds. If Exposure Time is set to 50, and the Timer Resolution value is 1, the CCD exposure time (integration time) = 50 microseconds.			
getFitParams	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET FIT PARAMS [3] CCD out [2] fitParameters [0] error out	Gets the FIT parameters contained in the CCD configuration for the conversion of pixel to wavelength if done via the settings contained in the CCD.			
getGain	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET GAIN [3] CCD out [2] info [1] token [0] error out	Gets the current gain token and the associated description information for the gain token. Gain tokens and their descriptions are part of the CCD configuration information. See ccd_getConfig command. For example: "gains": [{ "info": "Best Dynamic Range", "token": 1 }, { "info": "High Sensitivity", "token": 2 }, { "info": "High Light", "token": 0 }]			

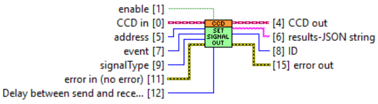
Name	Connector pane	Description	S.	R.	I.
getParallelSpeed	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET PARALLEL SPEED [3] CCD out [2] info [1] token [0] error out	Gets the current parallel speed token and token description. Parallel speed tokens and their descriptions are contained in the CCD configuration information. See ccd_getConfig command. Note: The Parallel Speed value may also be referred to as the Vertical Shift Rate. These terms are interchangeable. For example: "parallelSpeeds": [{ "info": "9.6 µSec", "token": 1 }, { "info": "4.9 µSec", "token": 2 }, { "info": "19 µSec", "token": 0 }],			
getSignalOut	 [5] address CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET SIGNAL OUT [3] CCD out [2] event [1] signalType [0] error out	ccd_getSignalOut This command is used to get the current setting of the signal output. The address, event, and signalType parameters are used to define the signal based on the supported options of that particular CCD. The supported signal options are retrieved using the ccd_getConfig command, and begin with the "Signals" string contained in the configuration. For example: "signals": [{ "events": [{ "name": "Shutter Open", "token": 3, "types": [{ "name": "TTL Active Low", "token": 1 }, { "name": "TTL Active High", "token": 0 }] }, { "name": "Start Experiment", "token": 0, "types": [{ "name": "TTL Active Low", "token": 1 }, { "name": "TTL Active High", "token": 0 }] }], "name": "Signal Output", "token": 0 }]			
getSpeed	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET SPEED [3] CCD out [2] info [1] token [0] error out	ccd_getSpeed Gets the current speed token and the associated description information for the speed token. Speed tokens and their descriptions are part of the CCD configuration information. See ccd_getConfig command. For example: "speeds": [{ "info": "500 kHz ", "token": 1 }, { "info": "500 kHz Ultra", "token": 2 }, { "info": "500 kHz Wrap", "token": 127 }, { "info": " 45 kHz ", "token": 0 }]			

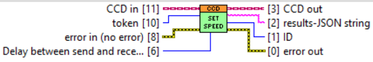
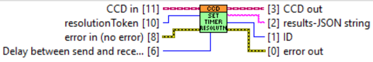
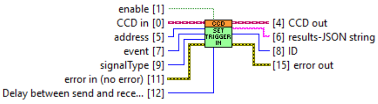
Name	Connector pane	Description	S.	R.	I.
getTimerResolution	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET TIMER RESOLUTN [3] CCD out [2] resolutionToken [0] error out	Gets the current timer resolution token. results description resolutionToken Integer. Timer resolution token. 0 - Timer resolution is set to 1000 microseconds 1 - Timer resolution is set to 1 microsecond			
getTriggerIn	 CCD in [0] error in (no error) [11] Delay between send and rece... [12] CCD GET TRIGGER IN [4] CCD out [6] address [8] event [10] signalType [15] error out	This command is used to get the current setting of the input trigger. The address, event, and signalType parameters are used to define the input trigger based on the supported options of that particular CCD. The supported trigger options are retrieved using the ccd_getConfig command, and begin with the "Triggers" string contained in the configuration. For example: "triggers": [{ "events": [{ "name": "Each - For Each Acq", "token": 1, "types": [{ "name": "TTL Rising Edge", "token": 1 }, { "name": "TTL Falling Edge", "token": 0 }] }, { "name": "Once - Start All", "token": 0, "types": [{ "name": "TTL Rising Edge", "token": 1 }, { "name": "TTL Falling Edge", "token": 0 }] }], "name": "Trigger Input", "token": 0 }]			
getXAxisConversionType	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD GET X-AXIS CONV TYPE [3] CCD out [2] type [0] error out	Gets the X axis pixel conversion type to be used when retrieving the acquisition data with the ccd_getAcquisitionData command. results description type Integer. The X-axis pixel conversion type to be used. 0 = None (default) 1 = CCD FIT parameters contained in the CCD firmware 2 = Mono Wavelength parameters contained in the icl_settings.ini file			
restart	 CCD in [11] error in (no error) [8] Delay between send and rece... [6] CCD RESTART [3] CCD out [2] results-JSON string [1] ID [0] error out	Performs a restart on the CCD.			
setAcqCount	 CCD in [11] count [10] error in (no error) [8] Delay between send and rece... [6] CCD SET ACQ COUNT [3] CCD out [2] results-JSON string [1] ID [0] error out	Sets the number of acquisition measurements to be performed sequentially by the hardware. A count > 1 is commonly referred to as "MultiAcq".			

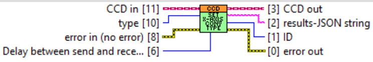
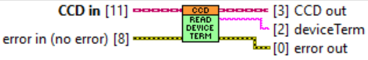
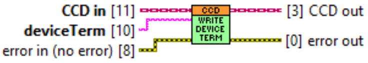
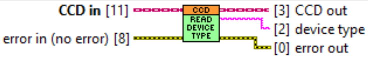
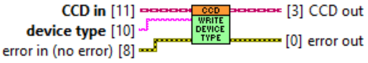
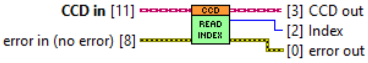
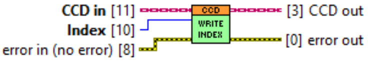
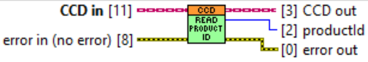
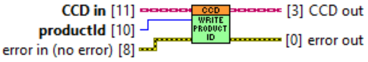
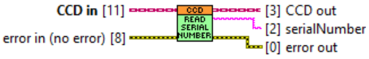
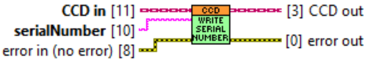
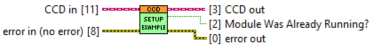
Name	Connector pane	Description	S.	R.	I.
setAcqFormat	 CCD in [11] number of ROIs [10] format [9] error in (no error) [8] Delay between send and rece... [6] [3] CCD out [2] results-JSON string [1] ID [0] error out	Sets the acquisition format and the number of ROIs (Regions of Interest) or areas. This command will remove all previously defined ROIs. After using this command, the ccd_setRoi command should be used to define each ROI. parameter description numberOfRois Integer. Number of ROIs (Regions of Interest / areas) format Integer. The acquisition format. 0 = Spectra 1 = Image 2 = Crop* 3 = Fast Kinetics* * Note: The Crop (2) and Fast Kinetics (3) acquisition formats are not supported by every CCD.			
setCenterWavelength	 CCD in [11] wavelength [10] error in (no error) [8] Delay between send and rece... [6] [3] CCD out [2] results-JSON string [1] ID [0] error out	This command sets the center wavelength value and other parameters to be used in the pixel to wavelength conversion. Note: This command should be called before ccd_setXAxisConversionType and ccd_setAcquisitionStart and is only useful uf the xAxisConversion type is set to Fitparams.			
setCleanCount	 CCD in [11] count [10] mode [9] error in (no error) [8] Delay between send and rece... [6] [3] CCD out [2] results-JSON string [1] ID [0] error out	Sets the number of cleans to be performed according to the specified mode setting. parameter description index Integer. Used to identify which CCD to target. See ccd_list command count Integer. Number of cleans. mode Integer. Specifies how the cleans will be performed. 0 = Never 1 = First Only 2 = Between Only 3 = Each			

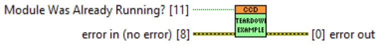
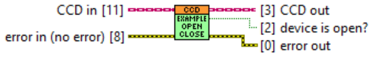
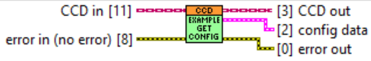
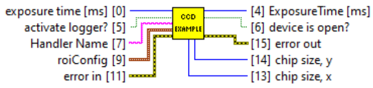
Name	Connector pane	Description	S.	R.	I.
setExposureTime	 CCD in [11] exposure time [ms] or [us] [10] error in (no error) [8] Delay between send and rece... [6] CCD SET EXPOSURE TIME [3] CCD out [2] results-JSON string [1] ID [0] error out	Sets the exposure time (expressed in Timer Resolution units). Note: To check the current Timer Resolution value see ccd_getTimerResolution. Alternatively the Timer Resolution value can be set using ccd_setTimerResolution. Example: If Exposure Time is set to 50, and the Timer Resolution value is 1000, the CCD exposure time (integration time) = 50 milliseconds. If Exposure Time is set to 50, and the Timer Resolution value is 1, the CCD exposure time (integration time) = 50 microseconds.			
setGain	 CCD in [11] token [10] error in (no error) [8] Delay between send and rece... [6] CCD SET GAIN [3] CCD out [2] results-JSON string [1] ID [0] error out	Sets the CCD gain token. A list of supported gain tokens can be found in the CCD configuration. See ccd_getConfig command. For example: "gains": [{ "info": "Best Dynamic Range", "token": 1 }, { "info": "High Sensitivity", "token": 2 }, { "info": "High Light", "token": 0 }]			
setParallelSpeed	 CCD in [11] token [10] error in (no error) [8] Delay between send and rece... [6] CCD SET PARALLEL SPEED [3] CCD out [2] results-JSON string [1] ID [0] error out	Sets the CCD parallel speed token. A list of supported parallel speed tokens can be found in the CCD configuration. See ccd_getConfig command. Note: The Parallel Speed value may also be referred to as the Vertical Shift Rate. These terms are interchangeable. For example: "parallelSpeeds": [{ "info": "9.6 µSec", "token": 1 }, { "info": "4.9 µSec", "token": 2 }, { "info": "19 µSec", "token": 0 }],			

Name	Connector pane	Description	S.	R.	I.
setRoi	 CCD in [11] roiConfig [10] error in (no error) [8] Delay between send and rece... [6] CCD SET ROI [3] CCD out [2] results-JSON string [1] ID [0] error out	Sets a single (roiIndex) ROI (Region of Interest) or area as defined by the X and Y origin, size, and bin parameters. The number of ROIs may be set using the ccd_setAcqFormat command. For Spectral acquisition format set yBin = ySize. Note: All values must fall within the x and y limits of the chip sensor, see ccd_getChipSize. If the ROI is not valid, the device will not be properly setup for acquisition. Command Parameters: parameter description index Integer. Used to identify which CCD to target. See ccd_list command roiIndex Integer. The region of interest's index (one-based) xOrigin Integer. The starting pixel in the x direction (zero-based) yOrigin Integer. The starting pixel in the y direction (zero-based) xSize Integer. The number of pixels in the x direction (one-based) ySize Integer. The number of pixels in the y direction (one-based) xBin Integer. The number of pixels to "bin" (x pixels summed to 1 value) yBin Integer. The number of pixels to "bin" (y pixels summed to 1 value)			

Name	Connector pane	Description	S.	R.	I.
setSignalOut	 enable [1] CCD in [0] address [5] event [7] signalType [9] error in (no error) [11] Delay between send and rece... [12] CCD SET SIGNAL OUT [4] CCD out [6] results-JSON string [8] ID [15] error out	This command is used to enable or disable the signal output. When enabling the signal output, the address, event, and signalType parameters are used to define the signal based on the supported options of that particular CCD. The supported signal options are retrieved using the ccd_getConfig command, and begin with the "Signals" string contained in the configuration. For example: "signals": [{ "events": [{ "name": "Shutter Open", "token": 3, "types": [{ "name": "TTL Active Low", "token": 1 }, { "name": "TTL Active High", "token": 0 }] }, { "name": "Start Experiment", "token": 0, "types": [{ "name": "TTL Active Low", "token": 1 }, { "name": "TTL Active High", "token": 0 }] }], "name": "Signal Output", "token": 0 }] parameter description index Integer. Used to identify which CCD to target. See ccd_list command enable Boolean. Enables or disables the signal. true = enable false = disable Note: When disabling the signal output, the address, event, and signalType parameters are ignored. address Integer. Token used to specify where the signal is located. (e.g. 0 = Signal Output) Note: Signal name and token can be found in the CCD config, see ccd_getConfig event Integer. Token used to specify when the signal event should occur. (e.g. 3 = Shutter Open) Note: Event name and token can be found in the CCD config, see ccd_getConfig signalType Integer. Token used to specify how the signal will cause the event. (e.g. 0 = TTL Active High) Note: Signal type and token can be found in the CCD config, see ccd_getConfig			

Name	Connector pane	Description	S.	R.	I.
setSpeed	 CCD in [11] token [10] error in (no error) [8] Delay between send and rece... [6] CCD SET SPEED [3] CCD out [2] results-JSON string [1] ID [0] error out	Sets the CCD speed token. A list of supported speed tokens can be found in the CCD configuration. See ccd_getConfig command. For example: "speeds": [{ "info": "500 kHz ", "token": 1 }, { "info": "500 kHz Ultra", "token": 2 }, { "info": "500 kHz Wrap", "token": 127 }, { "info": " 45 kHz ", "token": 0 }]			
setTimerResolution	 CCD in [11] resolutionToken [10] error in (no error) [8] Delay between send and rece... [6] CCD SET TIMER RESOLUTN [3] CCD out [2] results-JSON string [1] ID [0] error out	Sets the current timer resolution token. resolutionToken Integer. Timer resolution token. 0 - Sets the timer resolution to 1000 microseconds 1 - Sets the timer resolution to 1 microsecond*			
setTriggerIn	 enable [1] CCD in [0] address [5] event [7] signalType [9] error in (no error) [11] Delay between send and rece... [12] CCD SET TRIGGER IN [4] CCD out [6] results-JSON string [8] ID [15] error out	This command is used to enable or disable the trigger input. When enabling the trigger input, the address, event, and signalType parameters are used to define the input trigger based on the supported options of that particular CCD. The supported trigger options are retrieved using the ccd_getConfig command, and begin with the "Triggers" string contained in the configuration. For example: "triggers": [{ "events": [{ "name": "Each - For Each Acq", "token": 1, "types": [{ "name": "TTL Rising Edge", "token": 1 }, { "name": "TTL Falling Edge", "token": 0 }] }, { "name": "Once - Start All", "token": 0, "types": [{ "name": "TTL Rising Edge", "token": 1 }, { "name": "TTL Falling Edge", "token": 0 }] }], "name": "Trigger Input", "token": 0 }]			

Name	Connector pane	Description	S.	R.	I.
setXAxisConversionType	 CCD in [11] type [10] error in (no error) [8] Delay between send and rece... [6] [3] CCD out [2] results-JSON string [1] ID [0] error out	Sets the X-axis pixel conversion type to be used when retrieving the acquisition data with the ccd_getAcquisitionData command. Note: To use the parameters contained in the icl_settings.ini file, the ccd_setCenterWavelength command must be called first. type Integer. The X-axis pixel conversion type to be used. 0 = None (default) 1 = CCD FIT parameters contained in the CCD firmware 2 = Mono Wavelength parameters contained in the icl_settings.ini file			
Read DeviceTerm	 CCD in [11] error in (no error) [8] [3] CCD out [2] deviceTerm [0] error out	After initialization of the device this property gives back the device term which is "ccd" for the device type CCD. This is used for prefixing all commands to the ICL.			
Write DeviceTerm	 CCD in [11] deviceTerm [10] error in (no error) [8] [3] CCD out [0] error out	Sets the device term after init.			
Read DeviceType	 CCD in [11] error in (no error) [8] [3] CCD out [2] device type [0] error out	Returns the device type of the device saved in the device firmware.			
Write DeviceType	 CCD in [11] device type [10] error in (no error) [8] [3] CCD out [0] error out	Sets the device type after init			
Read Index	 CCD in [11] error in (no error) [8] [3] CCD out [2] Index [0] error out	Returns the index of the device in the ICL layer.			
Write Index	 CCD in [11] Index [10] error in (no error) [8] [3] CCD out [0] error out	Sets the device id after init			
Read productId	 CCD in [11] error in (no error) [8] [3] CCD out [2] productId [0] error out	Returns the productID of the device saved in the device firmware.			
Write productId	 CCD in [11] productId [10] error in (no error) [8] [3] CCD out [0] error out	Sets the device productID after init			
Read serialNumber	 CCD in [11] error in (no error) [8] [3] CCD out [2] serialNumber [0] error out	Returns the serial of the device saved in the device firmware.			
Write serialNumber	 CCD in [11] serialNumber [10] error in (no error) [8] [3] CCD out [0] error out	Sets the device serial after init			
Setup Example	 CCD in [11] error in (no error) [8] [3] CCD out [2] Module Was Already Running? [0] error out	No description found (add content in vi description)			

Name	Connector pane	Description	S.	R.	I.
Teardown Example	 Module Was Already Running? [11] error in (no error) [8] CCD TEARDOWN EXAMPLE [0] error out	No description found (add content in vi description)			
Example - Open Close CCD	 CCD in [11] error in (no error) [8] CCD EXAMPLE OPEN CLOSE [3] CCD out [2] device is open? [0] error out	No description found (add content in vi description)			
Example - Get Config	 CCD in [11] error in (no error) [8] CCD EXAMPLE GET CONFIG [3] CCD out [2] config data [0] error out	No description found (add content in vi description)			
CCD Example	 exposure time [ms] [0] activate logger? [5] Handler Name [7] roiConfig [9] error in [11] CCD EXAMPLE [4] ExposureTime [ms] [6] device is open? [15] error out [14] chip size, y [13] chip size, x	Example for a CCD spectrum acquisition.			

Scope: 🔑 → Protected | 🔑 → Community

Reentrancy: P → Preallocated reentrancy | S → Shared reentrancy

Inlining: ⇉ → Inlined

3.6. SCD.lvclass

Responsibility: This class contains all functionality needed for Horiba's Single-Channel-detectors.

Version: 1.0.0.1

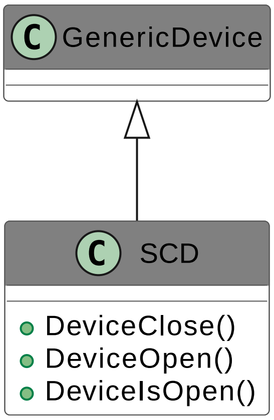
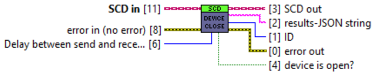
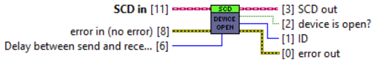
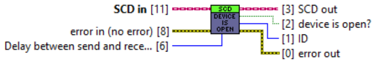
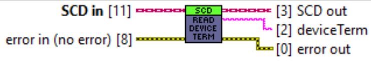
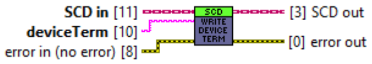
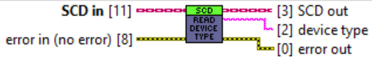
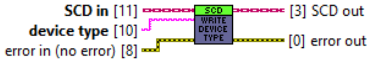
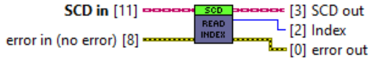
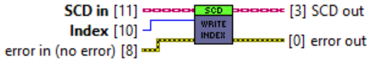
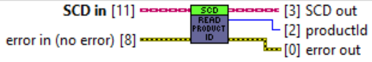
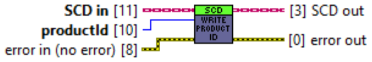
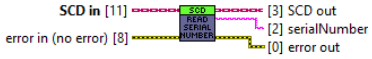
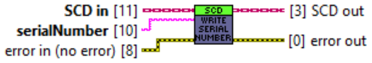


Table 14. Functions (non private scope only)

Name	Connector pane	Description	S.	R.	I.
DeviceClose	 SCD in [11] error in (no error) [8] Delay between send and rece... [6] SCD DEVICE CLOSE [3] SCD out [2] results-JSON string [1] ID [0] error out [4] device is open?	Closes communications with the CCD indicated by the index.			
DeviceOpen	 SCD in [11] error in (no error) [8] Delay between send and rece... [6] SCD DEVICE OPEN [3] SCD out [2] device is open? [1] ID [0] error out	Opens communications with the SpectrAcq3 indicated by the index command parameter.			
DeviceIsOpen	 SCD in [11] error in (no error) [8] Delay between send and rece... [6] SCD DEVICE IS OPEN [3] SCD out [2] device is open? [1] ID [0] error out	Returns true if selected SpectrAcq3 is open.			
Read DeviceTerm	 SCD in [11] error in (no error) [8] SCD READ DEVICE TERM [3] SCD out [2] deviceTerm [0] error out	Accessor VI for this class property.			

Name	Connector pane	Description	S.	R.	I.
Write DeviceTerm	 SCD in [11] deviceTerm [10] error in (no error) [8] SCD WRITE DEVICE TERM [3] SCD out [0] error out	Accessor VI for this class property.			
Read DeviceType	 SCD in [11] error in (no error) [8] SCD READ DEVICE TYPE [3] SCD out [2] device type [0] error out	Accessor VI for this class property.			
Write DeviceType	 SCD in [11] device type [10] error in (no error) [8] SCD WRITE DEVICE TYPE [3] SCD out [0] error out	Accessor VI for this class property.			
Read Index	 SCD in [11] error in (no error) [8] SCD READ INDEX [3] SCD out [2] Index [0] error out	Accessor VI for this class property.			
Write Index	 SCD in [11] Index [10] error in (no error) [8] SCD WRITE INDEX [3] SCD out [0] error out	Accessor VI for this class property.			
Read productId	 SCD in [11] error in (no error) [8] SCD READ PRODUCT ID [3] SCD out [2] productId [0] error out	Accessor VI for this class property.			
Write productId	 SCD in [11] productId [10] error in (no error) [8] SCD WRITE PRODUCT ID [3] SCD out [0] error out	Accessor VI for this class property.			
Read serialNumber	 SCD in [11] error in (no error) [8] SCD READ SERIAL NUMBER [3] SCD out [2] serialNumber [0] error out	Accessor VI for this class property.			
Write serialNumber	 SCD in [11] serialNumber [10] error in (no error) [8] SCD WRITE SERIAL NUMBER [3] SCD out [0] error out	Accessor VI for this class property.			

Scope: 🔑 → Protected | 🔑 → Community

Reentrancy: P → Preallocated reentrancy | S → Shared reentrancy

Inlining: ⇉ → Inlined

Chapter 4. Custom errors

TIP Custom errors are added via vi named `*--error.vi`.

Table 15. Custom errors

Name	Code	Description	Owned by
Module Not Running	0		DeviceManager.lvlib
Module Not Stopped	0		DeviceManager.lvlib
Module Not Synced	0		DeviceManager.lvlib
Request and Wait for Reply Timeout	0		DeviceManager.lvlib

Chapter 5. Legal Information

5.1. Document creation

This document has been generated using the following tools.

5.1.1. Antidoc

Project website: Antidoc

Maintainer website: Wovalab

5.1.2. Asciidoc for LabVIEW™

Project website: Asciidoc toolkit

Maintainer website: Wovalab

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

5.1.3. Graph Builder

Project website: Graph Builder

BSD 3-Clause License

Copyright © 2020, Cyril GAMBINI All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

5.1.4. classy Diagram Viewer

Project website: classy Diagram Viewer

5.2. Product used in the project

The documented project has been developed with the following products.

5.2.1. DQMH®

Find more details on DQMH® Consortium website