Spinnaker C

3.1.0.79

Generated by Doxygen 1.9.1

Chapter 1

Getting Started

The Spinnaker application programming interface (API) is used to interface with FLIR's USB3 Vision and GigE Vision cameras.

- Benefits of Spinnaker
- Software Licensing Information
- Software Maintenance Policy
- FlyCapture2 Feature Comparison with Spinnaker
- Programmer's Guide
- Working with GenICam GenTL Devices
- Myricom

Chapter 2

Programmer's Guide

Chapter 3

Benefits of Spinnaker

Please see (`http://softwareservices.flir.com/Spinnaker/latest/index.html`) for the latest version of this document

Chapter 4

FlyCapture2 Feature Comparison with Spinnaker

Please see (`http://softwareservices.flir.com/Spinnaker/latest/page3.html`) for the latest version of this document

Chapter 5

Working with GenICam GenTL Devices

5.1 GenTL Overview

FLIR GenTL Producer is a software driver that implements the GenICam™ GenTL 1.5 standard (`https↩ ://www.emva.org/`). It allows users to enumerate, communicate and stream from FLIR GigE Vision and USB3 Vision devices in a generic way independent from the underlying transport technology. This allows third-party software such as MATLAB (`https://www.mathworks.com`) and other software libraries to work with FLIR devices in a transport layer agnostic way. These applications are referred to as "GenTL Consumers," which directly use one or more GenTL Producers.

NOTE: Consumer applications must be aware of differences in device capabilities and be prepared to handle specific device models differently.

5.2 Installation

In order to use a FLIR GenTL producer, it needs to be properly registered and installed on the system. **The FLIR Producer comes packaged with the full Spinnaker SDK installer as of 2.x or newer.**

The GenTL Producer is provided as a platform dependent, dynamic loadable library file with the `.cti` ("Common Transport Interface") extension.

The Spinnaker SDK installer stores the folder paths for 32-bit and 64-bit GenTL Producers (`.cti` files) in environment variables named `GENICAM_GENTL32_PATH` and `GENICAM_GENTL64_PATH`, respectively. If there are multiple GenTL Producers installed on the system, path entries must be separated by **;** on Windows and **:** on UNIX-like systems.

NOTE: A 32bit GenTL consumer application will require a 32-bit GenTL producer and a 64-bit application will require a 64-bit producer library.

5.3 Troubleshooting

5.3.1 Enable FLIR GenTL Logging

FLIR GenTL Logging can be enabled if a configuration file with the name "log4cpp.gentl.property" resides in the path of where the consumer application executes from. For MATLAB, this is where the working directory is set and may default to the "Downloads" folder on Windows.

Sample log4cpp.gentl.property configuration file:

```
# FLIR GenTL Property Configuration file
log4cpp.rootCategory=ERROR, rootAppender
log4cpp.category.GenTLCategory=ERROR, GenTLCategory

log4cpp.appender.rootAppender=ConsoleAppender
log4cpp.appender.rootAppender.layout=PatternLayout
log4cpp.appender.rootAppender.layout.ConversionPattern=[%p] %d [%t] %m%n

log4cpp.appender.GenTLCategory=RollingFileAppender
log4cpp.appender.GenTLCategory.fileName=$(ALLUSERSPROFILE)\Spinnaker\Logs\GenTL.log
log4cpp.appender.GenTLCategory.append=true
log4cpp.appender.GenTLCategory.maxFileSize=1000000
log4cpp.appender.GenTLCategory.maxBackupIndex=5
log4cpp.appender.GenTLCategory.layout=PatternLayout
log4cpp.appender.GenTLCategory.layout.ConversionPattern=[%p] %d [%t] %m%n
```

5.3.2 USB3 Device Image Tearing

Image tearing could occur with certain USB3 host controllers when streaming with a GenTL producer. To work around the issue, make sure the size of each buffer announced to the FLIR GenTL producer is 1024 bytes aligned. The size of each buffer should be (bufferSize + 1024 - 1) / 1024) $*$ 1024 where 1024 is the USB3 packet transfer size.

For more information about image tearing causes and solutions, please refer to: https://www.flir.↩
com/support-center/iis/machine-vision/application-note/image-tearing-causes-and-solution

Chapter 6

Software Licensing Information

Table 6.1 License table

Component	License
Spinnaker	Copyright (c) 2001-2020 FLIR Systems, Inc. All Rights Reserved. This software is the confidential and proprietary information of FLIR Integrated Imaging Solutions, Inc. ("↩ Confidential Information"). You shall not disclose such Confidential Information and shall use it only in accordance with the terms of the license agreement you entered into with FLIR Integrated Imaging Solutions, Inc. (FLIR). FLIR MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. FLIR SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES.
GenICam	GenICam License
AdapterList	The Code Project Open License (CPOL)
Make ListView.ScrollIntoView Scroll the Item into the Center of the ListView	WP:CC_BY-SA License
Work with Bitmaps Faster in C#	The Code Project Open License (CPOL) 1.02
FreeImage	FreeImage public license
Boost	Boost Software License
Libusb	LGPLv2.1 License
Libraw1394	LGPLv2.0 License
FFMPEG	LGPv2.1 License
log4Net	Apache license 2.0
log4Cpp	LGPL License

The licenses mentioned above can also be found in the Spinnaker installed license folder.

Chapter 7

Software Maintenance Policy

7.1 GenTL Overview

This document outlines the FLIR maintenance policy for Spinnaker Software Development Kit (SDK). FLIR regularly provides SDK updates that may contain support for new or updated features, enhancements, updated drivers, updated examples, bug fixes or documentation updates. Updates may also address changes with introducing and/or deprecating language runtimes, operating systems and dependencies.

We recommend users to stay up-to-date with SDK releases to keep up with the latest features, bug fixes and performance improvements. Continued use of an unsupported SDK version is not recommended and is done at the user's discretion.

Spinnaker SDK releases are published through our website and can be found here: `https://www.flir.ca/products/spinnaker-sdk/`

7.2 Platform Support Policy

7.2.1 Windows Support

FLIR will continue to maintain, fix and build the last two major versions of Spinnaker SDK against the latest available version of Windows x86/x64. The latest three versions of Visual Studio compiler toolchain are supported on Windows. Only the latest compiler toolchain on the latest available version of Windows are being actively tested.

7.2.2 Linux Desktop Support

FLIR will continue to maintain, fix and build the last two major versions of Spinnaker SDK against the latest two LTS versions of Ubuntu x86/x64. Only the latest x64 LTS version of Ubuntu is being actively tested.

7.2.3 Linux Embedded Support

FLIR will continue to maintain, fix and build the last two major versions of Spinnaker SDK against the latest supported LTS version of Ubuntu ARMHF/ARM64 for a specific board. Only the latest LTS Ubuntu version on an ARM64 board is being actively tested. Contact sales if you need support for a specific embedded board.

7.2.4 MacOS Support

FLIR will continue to maintain, fix and build the last two major versions of Spinnaker SDK against MacOS Mojave (10.14). Contact sales if you need newer MacOS support.

7.3 Versioning Policy

Spinnaker SDK releases use a modified semantic versioning scheme and is indicated by four sets of numbers separated by periods:

MAJOR.MINOR.0.PATCH

- MAJOR: Version change that can include incompatible API changes
- MINOR: Version change that adds functionality in a backwards-compatible manner
- PATCH: Version change with backwards-compatible fixes

Reference: `https://www.flir.com/support-center/iis/machine-vision/knowledge-base/flir-mac`

Chapter 8

Module Index

8.1 Modules

Here is a list of all modules:

Chapter 9

Data Structure Index

9.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 10

File Index

10.1 File List

Here is a list of all files with brief descriptions:

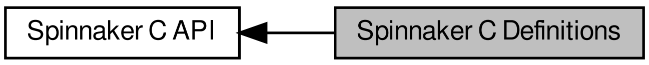
Chapter 11

Module Documentation

11.1 Spinnaker C Definitions

Definitions for Spinnaker C.

Collaboration diagram for Spinnaker C Definitions:



Definitions for Spinnaker C.

Definitions for Spinnaker C API.

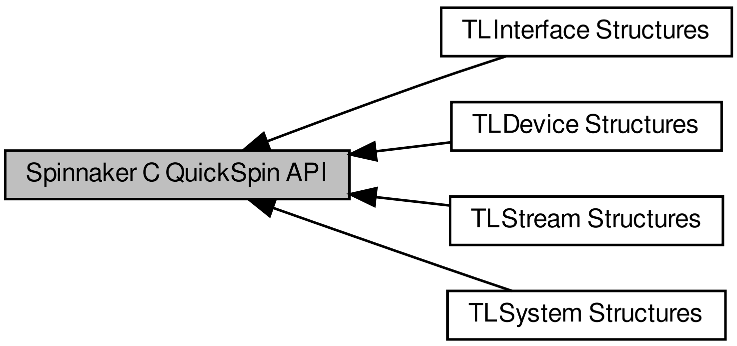
Holds enumerations, typedefs and structures that are used across the Spinnaker C API wrapper.

11.2 Camera Enumerations

11.3 Chunk Data Structures

11.4 Spinnaker C QuickSpin API

Collaboration diagram for Spinnaker C QuickSpin API:



Modules

- TLDevice Structures
- TLInterface Structures
- TLStream Structures
- TLSystem Structures

11.4.1 Detailed Description

11.5 QuickSpin Access

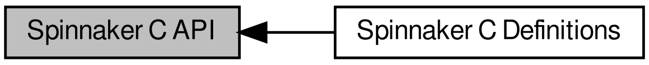
The functions in this section initialize the various QuickSpin structs for the C API.

The functions in this section initialize the various QuickSpin structs for the C API.

11.6 Spinnaker C API

SpinnakerPlatform C Include.

Collaboration diagram for Spinnaker C API:



Modules

- Spinnaker C Definitions

  Definitions for Spinnaker C.

11.6.1 Detailed Description

SpinnakerPlatform C Include.

Spinnaker C Definition Includes Spinnaker GenICam C Wrapper Includes Spinnaker QuickSpin C Includes

Spinnaker C Definition Includes

11.7 Error Handling

The functions in this section provide access to additional information related to error returns.

The functions in this section provide access to additional information related to error returns.

11.8 System Access

The functions in this section provide access to information, objects, and functionality of the system object.

The functions in this section provide access to information, objects, and functionality of the system object.

This includes the system object, interface and camera lists, and interface and logging events.

11.9 InterfaceList Access

The functions in this section provide access to information, objects, and functionality of interface lists.

The functions in this section provide access to information, objects, and functionality of interface lists.

This includes updating, size and interface retrieval, and clearance.

11.10 CameraList Access

The functions in this section provide access to information, objects, and functionality of camera lists.

The functions in this section provide access to information, objects, and functionality of camera lists.

This includes updating, size and camera retrieval, and clearance.

11.11 ImageList Access

The functions in this section provide access to information, objects, and functionality of image lists.

The functions in this section provide access to information, objects, and functionality of image lists.

This includes updating, size and image retrieval, and clearance.

11.12 Interface Access

The functions in this section provide access to information, objects, and functionality of interfaces.

The functions in this section provide access to information, objects, and functionality of interfaces.

This includes camera list and nodemap retrieval, event handler registration, and interface release.

11.13 Camera Access

The functions in this section provide access to information, objects, and functionality of cameras.

The functions in this section provide access to information, objects, and functionality of cameras.

This includes nodemap retrieval, acquisition and init commands, event handler registration, and camera property retrieval.

11.14 Image Access

The functions in this section provide access to information and functionality of images.

The functions in this section provide access to information and functionality of images.

This includes creation, destruction, and saving as well as a wealth of information including things like width, height, stride, and timestamp.

11.15 Image Processor Access

The functions in this section provide access to information and functionality of image processor.

The functions in this section provide access to information and functionality of image processor.

This includes image processor creation, deletion, image conversion, image decompression and image post processing methods.

All supported input image pixel formats can be converted to supported output image pixel formats. If the input pixel format is a compressed format, the decompression will occur before converting to the output pixel format.

List of supported input image pixel formats:

- PixelFormat_Mono8
- PixelFormat_Mono16
- PixelFormat_BayerGR8
- PixelFormat_BayerRG8
- PixelFormat_BayerGB8
- PixelFormat_BayerBG8
- PixelFormat_BayerGR16
- PixelFormat_BayerRG16
- PixelFormat_BayerGB16
- PixelFormat_BayerBG16
- PixelFormat_Mono12Packed
- PixelFormat_BayerGR12Packed
- PixelFormat_BayerRG12Packed
- PixelFormat_BayerGB12Packed
- PixelFormat_BayerBG12Packed
- PixelFormat_YUV411Packed
- PixelFormat_YUV422Packed
- PixelFormat_YUV444Packed
- PixelFormat_Mono12p

- PixelFormat_BayerGR12p
- PixelFormat_BayerRG12p
- PixelFormat_BayerGB12p
- PixelFormat_BayerBG12p
- PixelFormat_YCbCr8
- PixelFormat_YCbCr422_8
- PixelFormat_YCbCr411_8
- PixelFormat_BGR8
- PixelFormat_BGRa8
- PixelFormat_Mono10Packed
- PixelFormat_BayerGR10Packed
- PixelFormat_BayerRG10Packed
- PixelFormat_BayerGB10Packed
- PixelFormat_BayerBG10Packed
- PixelFormat_Mono10p
- PixelFormat_BayerGR10p
- PixelFormat_BayerRG10p
- PixelFormat_BayerGB10p
- PixelFormat_BayerBG10p
- PixelFormat_Mono10
- PixelFormat_Mono12
- PixelFormat_Mono14
- PixelFormat_BayerBG10
- PixelFormat_BayerBG12
- PixelFormat_BayerGB10
- PixelFormat_BayerGB12
- PixelFormat_BayerGR10
- PixelFormat_BayerGR12
- PixelFormat_BayerRG10
- PixelFormat_BayerRG12
- PixelFormat_RGBa8
- PixelFormat_RGB8
- PixelFormat_BGR16
- PixelFormat_R12
- PixelFormat_G12
- PixelFormat_B12

- PixelFormat_YUV8_UYV
- PixelFormat_YUV411_8_UYYVYY
- PixelFormat_YUV422_8
- PixelFormat_Polarized8
- PixelFormat_Polarized10p
- PixelFormat_Polarized12p
- PixelFormat_Polarized16
- PixelFormat_BayerRGPolarized8
- PixelFormat_BayerRGPolarized10p
- PixelFormat_BayerRGPolarized12p
- PixelFormat_BayerRGPolarized16
- PixelFormat_LLCMono8
- PixelFormat_LLCBayerRG8
- PixelFormat_JPEGMono8
- PixelFormat_JPEGColor8
- PixelFormat_Raw16
- PixelFormat_Raw8
- PixelFormat_R12_Jpeg
- PixelFormat_GR12_Jpeg
- PixelFormat_GB12_Jpeg
- PixelFormat_B12_Jpeg

List of supported output image pixel formats

- PixelFormat_Mono8
- PixelFormat_Mono16
- PixelFormat_BayerBG8
- PixelFormat_BayerGB8
- PixelFormat_BayerRG8
- PixelFormat_BayerGR8
- PixelFormat_BayerBG16
- PixelFormat_BayerGB16
- PixelFormat_BayerRG16
- PixelFormat_BayerGR16
- PixelFormat_BGR8
- PixelFormat_BGRa8

- PixelFormat_RGB8
- PixelFormat_RGBa8
- PixelFormat_BGR16
- PixelFormat_RGB16
- PixelFormat_R12
- PixelFormat_G12
- PixelFormat_B12

11.16 Event Access

The functions in this section allow for the creation and destruction of events.

The functions in this section allow for the creation and destruction of events.

11.17 ImageStatistics Access

The functions in this section provide access to information and functionality related to image statistics.

The functions in this section provide access to information and functionality related to image statistics.

This includes context creation and destruction, the enabling and disabling of channels, and value retrieval.

11.18 Logging Event Data Access

The functions in this section allow for the retrieval of logging event data.

The functions in this section allow for the retrieval of logging event data.

11.19 Device Event Data Access

The functions in this section allow for the retrieval of device event data.

The functions in this section allow for the retrieval of device event data.

11.20 Chunk data access

The functions in this section provide access to chunk data stored on images.

The functions in this section provide access to chunk data stored on images.

11.21 Spinnaker C Handles

Spinnaker C handle definitions.

Spinnaker C handle definitions.

11.22 Spinnaker C Function Signatures

Spinnaker C function signature definitions.

Spinnaker C function signature definitions.

11.23 Spinnaker C Enumerations

Spinnaker C enumumeration definitions.

Spinnaker C enumumeration definitions.

11.24 Spinnaker C Structures

Spinnaker C structure definitions.

Spinnaker C structure definitions.

11.25 Spinnaker C GenICam API

11.26 Node Map Access

The functions in this section provide access to information, objects, and functionality related to nodemaps.

The functions in this section provide access to information, objects, and functionality related to nodemaps.

This includes nodes, node counts, and polling.

11.27 Node Access

The functions in this section provide access to information and objects retrieved from nodes.

The functions in this section provide access to information and objects retrieved from nodes.

This includes node properties and callback registration.

11.28 IValue Access

The functions in this section provide access to nodes as value nodes.

The functions in this section provide access to nodes as value nodes.

As value nodes are not an actual node type, the functions are named as regular nodes. Functions include reading from and writing to any node with a string.

11.29 String Access

The functions in this section provide access to string nodes using character pointers and arrays.

The functions in this section provide access to string nodes using character pointers and arrays.

This includes getters and setters of values and value lengths.

11.30 IInteger Access

The functions in this section provide access to integer nodes using the int64_t data type.

The functions in this section provide access to integer nodes using the int64_t data type.

This includes value getters and setters, min, max, and increment functions, and node representation.

11.31 IFloat Access

The functions in this section provide access to float nodes using double as the data type.

The functions in this section provide access to float nodes using double as the data type.

This includes value getters and setters, min and max functions, and node representation.

11.32 IEnumeration Access

The functions in this section provide access to enum nodes.

The functions in this section provide access to enum nodes.

This includes retrieving the number of entries, an entry by index or name, retrieving the current entry node, or setting the node using an integer.

11.33 IEnumEntry Access

The functions in this section provide access to entry nodes This includes retrieving the integer value or the symbolic of an entry.

The functions in this section provide access to entry nodes This includes retrieving the integer value or the symbolic of an entry.

11.34 IBoolean Access

The functions in this section provide access to boolean nodes using the bool8_t data type, values represented with 'True' and 'False'.

The functions in this section provide access to boolean nodes using the bool8_t data type, values represented with 'True' and 'False'.

This includes value getters and setters.

11.35 ICommand Access

The functions in this section all provide access to information and objects retrieved from nodes.

The functions in this section all provide access to information and objects retrieved from nodes.

This includes node properties and callbacks.

11.36 ICategory Access

The functions in this section all provide access to information and objects retrieved from nodes.

The functions in this section all provide access to information and objects retrieved from nodes.

This includes node properties and callbacks.

11.37 IRegister Access

The functions in this section provide access to register nodes.

The functions in this section provide access to register nodes.

This includes access to the node, its address and length, and reference.

11.38 Spinnaker C GenICam Handles

Handle definitions for Spinnaker C GenICam API.

Handle definitions for Spinnaker C GenICam API.

11.39 Spinnaker C GenICam Enumerations

Enumeration definitions for Spinnaker C GenICam API.

Enumeration definitions for Spinnaker C GenICam API.

11.40 SpinVideo Recording Access

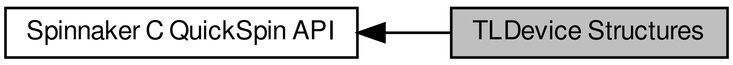
The functions in this section provide access to video recording capabilities, which include opening, building, and closing video files.

The functions in this section provide access to video recording capabilities, which include opening, building, and closing video files.

11.41 Transport Layer Enumerations

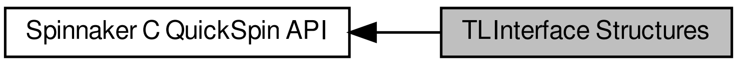
11.42 TLDevice Structures

Collaboration diagram for TLDevice Structures:



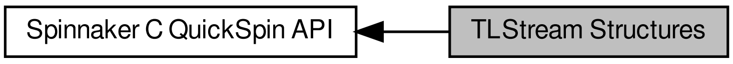
11.43 TLInterface Structures

Collaboration diagram for TLInterface Structures:



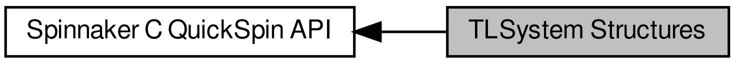
11.44 TLStream Structures

Collaboration diagram for TLStream Structures:



11.45 TLSystem Structures

Collaboration diagram for TLSystem Structures:



Chapter 12

Data Structure Documentation

12.1 actionCommandResult Struct Reference

Action Command Result.

Data Fields

- unsigned int DeviceAddress
- spinActionCommandStatus Status

12.1.1 Detailed Description

Action Command Result.

12.1.2 Field Documentation

12.1.2.1 DeviceAddress

```
unsigned int DeviceAddress
```

12.1.2.2 Status

```
spinActionCommandStatus Status
```

The documentation for this struct was generated from the following file:

- include/spinc/SpinnakerDefsC.h

12.2 quickSpin Struct Reference

Data Fields

- quickSpinIntegerNode LUTIndex
- quickSpinBooleanNode LUTEnable
- quickSpinIntegerNode LUTValue
- quickSpinEnumerationNode LUTSelector
- quickSpinFloatNode ExposureTime
- quickSpinCommandNode AcquisitionStop
- quickSpinFloatNode AcquisitionResultingFrameRate
- quickSpinFloatNode AcquisitionLineRate
- quickSpinCommandNode AcquisitionStart
- quickSpinCommandNode TriggerSoftware
- quickSpinEnumerationNode ExposureMode
- quickSpinEnumerationNode AcquisitionMode
- quickSpinIntegerNode AcquisitionFrameCount
- quickSpinEnumerationNode TriggerSource
- quickSpinEnumerationNode TriggerActivation
- quickSpinEnumerationNode SensorShutterMode
- quickSpinFloatNode TriggerDelay
- quickSpinEnumerationNode TriggerMode
- quickSpinFloatNode AcquisitionFrameRate
- quickSpinEnumerationNode TriggerOverlap
- quickSpinEnumerationNode TriggerSelector
- quickSpinBooleanNode AcquisitionFrameRateEnable
- quickSpinEnumerationNode ExposureAuto
- quickSpinIntegerNode AcquisitionBurstFrameCount
- quickSpinIntegerNode EventTest
- quickSpinIntegerNode EventTestTimestamp
- quickSpinIntegerNode EventExposureEndFrameID
- quickSpinIntegerNode EventExposureEnd
- quickSpinIntegerNode EventExposureEndTimestamp
- quickSpinIntegerNode EventError
- quickSpinIntegerNode EventErrorTimestamp
- quickSpinIntegerNode EventErrorCode
- quickSpinIntegerNode EventErrorFrameID
- quickSpinEnumerationNode EventSelector
- quickSpinBooleanNode EventSerialReceiveOverflow
- quickSpinIntegerNode EventSerialPortReceive
- quickSpinIntegerNode EventSerialPortReceiveTimestamp
- quickSpinStringNode EventSerialData
- quickSpinIntegerNode EventSerialDataLength
- quickSpinEnumerationNode EventNotification
- quickSpinIntegerNode LogicBlockLUTRowIndex
- quickSpinEnumerationNode LogicBlockSelector
- quickSpinEnumerationNode LogicBlockLUTInputActivation
- quickSpinEnumerationNode LogicBlockLUTInputSelector
- quickSpinEnumerationNode LogicBlockLUTInputSource
- quickSpinBooleanNode LogicBlockLUTOutputValue
- quickSpinIntegerNode LogicBlockLUTOutputValueAll
- quickSpinEnumerationNode LogicBlockLUTSelector
- quickSpinFloatNode ColorTransformationValue
- quickSpinBooleanNode ColorTransformationEnable

- quickSpinEnumerationNode ColorTransformationSelector
- quickSpinEnumerationNode RgbTransformLightSource
- quickSpinFloatNode Saturation
- quickSpinBooleanNode SaturationEnable
- quickSpinEnumerationNode ColorTransformationValueSelector
- quickSpinIntegerNode TimestampLatchValue
- quickSpinCommandNode TimestampReset
- quickSpinStringNode DeviceUserID
- quickSpinFloatNode DeviceTemperature
- quickSpinIntegerNode MaxDeviceResetTime
- quickSpinIntegerNode DeviceTLVersionMinor
- quickSpinStringNode DeviceSerialNumber
- quickSpinStringNode DeviceVendorName
- quickSpinEnumerationNode DeviceRegistersEndianness
- quickSpinStringNode DeviceManufacturerInfo
- quickSpinIntegerNode DeviceLinkSpeed
- quickSpinIntegerNode LinkUptime
- quickSpinIntegerNode DeviceEventChannelCount
- quickSpinCommandNode TimestampLatch
- quickSpinEnumerationNode DeviceScanType
- quickSpinCommandNode DeviceReset
- quickSpinEnumerationNode DeviceCharacterSet
- quickSpinIntegerNode DeviceLinkThroughputLimit
- quickSpinStringNode DeviceFirmwareVersion
- quickSpinIntegerNode DeviceStreamChannelCount
- quickSpinEnumerationNode DeviceTLType
- quickSpinStringNode DeviceVersion
- quickSpinEnumerationNode DevicePowerSupplySelector
- quickSpinStringNode SensorDescription
- quickSpinStringNode DeviceModelName
- quickSpinIntegerNode DeviceTLVersionMajor
- quickSpinEnumerationNode DeviceTemperatureSelector
- quickSpinIntegerNode EnumerationCount
- quickSpinFloatNode PowerSupplyCurrent
- quickSpinStringNode DeviceID
- quickSpinIntegerNode DeviceUptime
- quickSpinIntegerNode DeviceLinkCurrentThroughput
- quickSpinIntegerNode DeviceMaxThroughput
- quickSpinCommandNode FactoryReset
- quickSpinFloatNode PowerSupplyVoltage
- quickSpinEnumerationNode DeviceIndicatorMode
- quickSpinFloatNode DeviceLinkBandwidthReserve
- quickSpinIntegerNode AasRoiOffsetY
- quickSpinIntegerNode AasRoiOffsetX
- quickSpinEnumerationNode AutoExposureControlPriority
- quickSpinFloatNode BalanceWhiteAutoLowerLimit
- quickSpinFloatNode BalanceWhiteAutoDamping
- quickSpinIntegerNode AasRoiHeight
- quickSpinFloatNode AutoExposureGreyValueUpperLimit
- quickSpinFloatNode AutoExposureTargetGreyValue
- quickSpinFloatNode AutoExposureGainLowerLimit
- quickSpinFloatNode AutoExposureGreyValueLowerLimit
- quickSpinEnumerationNode AutoExposureMeteringMode
- quickSpinFloatNode AutoExposureExposureTimeUpperLimit
- quickSpinFloatNode AutoExposureGainUpperLimit

- quickSpinFloatNode AutoExposureControlLoopDamping
- quickSpinFloatNode AutoExposureEVCompensation
- quickSpinFloatNode AutoExposureExposureTimeLowerLimit
- quickSpinEnumerationNode BalanceWhiteAutoProfile
- quickSpinEnumerationNode AutoAlgorithmSelector
- quickSpinEnumerationNode AutoExposureTargetGreyValueAuto
- quickSpinBooleanNode AasRoiEnable
- quickSpinEnumerationNode AutoExposureLightingMode
- quickSpinIntegerNode AasRoiWidth
- quickSpinFloatNode BalanceWhiteAutoUpperLimit
- quickSpinIntegerNode LinkErrorCount
- quickSpinBooleanNode GevCurrentIPConfigurationDHCP
- quickSpinIntegerNode GevInterfaceSelector
- quickSpinIntegerNode GevSCPD
- quickSpinIntegerNode GevTimestampTickFrequency
- quickSpinIntegerNode GevSCPSPacketSize
- quickSpinIntegerNode GevCurrentDefaultGateway
- quickSpinBooleanNode GevSCCFGUnconditionalStreaming
- quickSpinIntegerNode GevMCTT
- quickSpinBooleanNode GevSCPSDoNotFragment
- quickSpinIntegerNode GevCurrentSubnetMask
- quickSpinIntegerNode GevStreamChannelSelector
- quickSpinIntegerNode GevCurrentIPAddress
- quickSpinIntegerNode GevMCSP
- quickSpinIntegerNode GevGVCPPendingTimeout
- quickSpinEnumerationNode GevIEEE1588Status
- quickSpinStringNode GevFirstURL
- quickSpinIntegerNode GevMACAddress
- quickSpinIntegerNode GevPersistentSubnetMask
- quickSpinIntegerNode GevMCPHostPort
- quickSpinIntegerNode GevSCPHostPort
- quickSpinBooleanNode GevGVCPPendingAck
- quickSpinIntegerNode GevSCPInterfaceIndex
- quickSpinBooleanNode GevSupportedOption
- quickSpinEnumerationNode GevIEEE1588Mode
- quickSpinBooleanNode GevCurrentIPConfigurationLLA
- quickSpinIntegerNode GevSCSP
- quickSpinBooleanNode GevIEEE1588
- quickSpinBooleanNode GevSCCFGExtendedChunkData
- quickSpinIntegerNode GevPersistentIPAddress
- quickSpinBooleanNode GevCurrentIPConfigurationPersistentIP
- quickSpinEnumerationNode GevIEEE1588ClockAccuracy
- quickSpinIntegerNode GevHeartbeatTimeout
- quickSpinIntegerNode GevPersistentDefaultGateway
- quickSpinEnumerationNode GevCCP
- quickSpinIntegerNode GevMCDA
- quickSpinIntegerNode GevSCDA
- quickSpinIntegerNode GevSCPDirection
- quickSpinBooleanNode GevSCPSFireTestPacket
- quickSpinStringNode GevSecondURL
- quickSpinEnumerationNode GevSupportedOptionSelector
- quickSpinBooleanNode GevGVCPHeartbeatDisable
- quickSpinIntegerNode GevMCRC
- quickSpinBooleanNode GevSCPSBigEndian
- quickSpinIntegerNode GevNumberOfInterfaces

- quickSpinIntegerNode TLParamsLocked
- quickSpinIntegerNode PayloadSize
- quickSpinIntegerNode PacketResendRequestCount
- quickSpinBooleanNode SharpeningEnable
- quickSpinEnumerationNode BlackLevelSelector
- quickSpinBooleanNode GammaEnable
- quickSpinBooleanNode SharpeningAuto
- quickSpinBooleanNode BlackLevelClampingEnable
- quickSpinFloatNode BalanceRatio
- quickSpinEnumerationNode BalanceWhiteAuto
- quickSpinFloatNode SharpeningThreshold
- quickSpinEnumerationNode GainAuto
- quickSpinFloatNode Sharpening
- quickSpinFloatNode Gain
- quickSpinEnumerationNode BalanceRatioSelector
- quickSpinEnumerationNode GainSelector
- quickSpinFloatNode BlackLevel
- quickSpinIntegerNode BlackLevelRaw
- quickSpinFloatNode Gamma
- quickSpinIntegerNode DefectTableIndex
- quickSpinCommandNode DefectTableFactoryRestore
- quickSpinIntegerNode DefectTableCoordinateY
- quickSpinCommandNode DefectTableSave
- quickSpinEnumerationNode DefectCorrectionMode
- quickSpinIntegerNode DefectTableCoordinateX
- quickSpinIntegerNode DefectTablePixelCount
- quickSpinBooleanNode DefectCorrectStaticEnable
- quickSpinCommandNode DefectTableApply
- quickSpinBooleanNode UserSetFeatureEnable
- quickSpinCommandNode UserSetSave
- quickSpinEnumerationNode UserSetSelector
- quickSpinCommandNode UserSetLoad
- quickSpinEnumerationNode UserSetDefault
- quickSpinEnumerationNode SerialPortBaudRate
- quickSpinIntegerNode SerialPortDataBits
- quickSpinEnumerationNode SerialPortParity
- quickSpinIntegerNode SerialTransmitQueueMaxCharacterCount
- quickSpinIntegerNode SerialReceiveQueueCurrentCharacterCount
- quickSpinEnumerationNode SerialPortSelector
- quickSpinEnumerationNode SerialPortStopBits
- quickSpinCommandNode SerialReceiveQueueClear
- quickSpinIntegerNode SerialReceiveFramingErrorCount
- quickSpinIntegerNode SerialTransmitQueueCurrentCharacterCount
- quickSpinIntegerNode SerialReceiveParityErrorCount
- quickSpinEnumerationNode SerialPortSource
- quickSpinIntegerNode SerialReceiveQueueMaxCharacterCount
- quickSpinIntegerNode SequencerSetStart
- quickSpinEnumerationNode SequencerMode
- quickSpinEnumerationNode SequencerConfigurationValid
- quickSpinEnumerationNode SequencerSetValid
- quickSpinIntegerNode SequencerSetSelector
- quickSpinEnumerationNode SequencerTriggerActivation
- quickSpinEnumerationNode SequencerConfigurationMode
- quickSpinCommandNode SequencerSetSave
- quickSpinEnumerationNode SequencerTriggerSource

- quickSpinIntegerNode SequencerSetActive
- quickSpinIntegerNode SequencerSetNext
- quickSpinCommandNode SequencerSetLoad
- quickSpinIntegerNode SequencerPathSelector
- quickSpinBooleanNode SequencerFeatureEnable
- quickSpinIntegerNode TransferBlockCount
- quickSpinCommandNode TransferStart
- quickSpinIntegerNode TransferQueueMaxBlockCount
- quickSpinIntegerNode TransferQueueCurrentBlockCount
- quickSpinEnumerationNode TransferQueueMode
- quickSpinEnumerationNode TransferOperationMode
- quickSpinCommandNode TransferStop
- quickSpinIntegerNode TransferQueueOverflowCount
- quickSpinEnumerationNode TransferControlMode
- quickSpinFloatNode ChunkBlackLevel
- quickSpinIntegerNode ChunkFrameID
- quickSpinStringNode ChunkSerialData
- quickSpinFloatNode ChunkExposureTime
- quickSpinIntegerNode ChunkCompressionMode
- quickSpinFloatNode ChunkCompressionRatio
- quickSpinBooleanNode ChunkSerialReceiveOverflow
- quickSpinIntegerNode ChunkTimestamp
- quickSpinBooleanNode ChunkModeActive
- quickSpinIntegerNode ChunkExposureEndLineStatusAll
- quickSpinEnumerationNode ChunkGainSelector
- quickSpinEnumerationNode ChunkSelector
- quickSpinEnumerationNode ChunkBlackLevelSelector
- quickSpinIntegerNode ChunkWidth
- quickSpinIntegerNode ChunkImage
- quickSpinIntegerNode ChunkHeight
- quickSpinEnumerationNode ChunkPixelFormat
- quickSpinFloatNode ChunkGain
- quickSpinIntegerNode ChunkSequencerSetActive
- quickSpinIntegerNode ChunkCRC
- quickSpinIntegerNode ChunkOffsetX
- quickSpinIntegerNode ChunkOffsetY
- quickSpinBooleanNode ChunkEnable
- quickSpinIntegerNode ChunkSerialDataLength
- quickSpinIntegerNode FileAccessOffset
- quickSpinIntegerNode FileAccessLength
- quickSpinEnumerationNode FileOperationStatus
- quickSpinCommandNode FileOperationExecute
- quickSpinEnumerationNode FileOpenMode
- quickSpinIntegerNode FileOperationResult
- quickSpinEnumerationNode FileOperationSelector
- quickSpinEnumerationNode FileSelector
- quickSpinIntegerNode FileSize
- quickSpinEnumerationNode BinningSelector
- quickSpinIntegerNode PixelDynamicRangeMin
- quickSpinIntegerNode PixelDynamicRangeMax
- quickSpinIntegerNode OffsetY
- quickSpinIntegerNode BinningHorizontal
- quickSpinIntegerNode Width
- quickSpinEnumerationNode TestPatternGeneratorSelector
- quickSpinFloatNode CompressionRatio

- quickSpinEnumerationNode CompressionSaturationPriority
- quickSpinBooleanNode ReverseX
- quickSpinBooleanNode ReverseY
- quickSpinEnumerationNode TestPattern
- quickSpinEnumerationNode PixelColorFilter
- quickSpinIntegerNode WidthMax
- quickSpinEnumerationNode AdcBitDepth
- quickSpinIntegerNode BinningVertical
- quickSpinEnumerationNode DecimationHorizontalMode
- quickSpinEnumerationNode BinningVerticalMode
- quickSpinIntegerNode OffsetX
- quickSpinIntegerNode HeightMax
- quickSpinIntegerNode DecimationHorizontal
- quickSpinEnumerationNode PixelSize
- quickSpinIntegerNode SensorHeight
- quickSpinEnumerationNode DecimationSelector
- quickSpinBooleanNode IspEnable
- quickSpinBooleanNode AdaptiveCompressionEnable
- quickSpinEnumerationNode ImageCompressionMode
- quickSpinIntegerNode DecimationVertical
- quickSpinIntegerNode Height
- quickSpinEnumerationNode BinningHorizontalMode
- quickSpinEnumerationNode PixelFormat
- quickSpinIntegerNode SensorWidth
- quickSpinEnumerationNode DecimationVerticalMode
- quickSpinCommandNode TestEventGenerate
- quickSpinCommandNode TriggerEventTest
- quickSpinIntegerNode GuiXmlManifestAddress
- quickSpinIntegerNode Test0001
- quickSpinBooleanNode V3_3Enable
- quickSpinEnumerationNode LineMode
- quickSpinEnumerationNode LineSource
- quickSpinEnumerationNode LineInputFilterSelector
- quickSpinBooleanNode UserOutputValue
- quickSpinIntegerNode UserOutputValueAll
- quickSpinEnumerationNode UserOutputSelector
- quickSpinBooleanNode LineStatus
- quickSpinEnumerationNode LineFormat
- quickSpinIntegerNode LineStatusAll
- quickSpinEnumerationNode LineSelector
- quickSpinEnumerationNode ExposureActiveMode
- quickSpinBooleanNode LineInverter
- quickSpinFloatNode LineFilterWidth
- quickSpinEnumerationNode CounterTriggerActivation
- quickSpinIntegerNode CounterValue
- quickSpinEnumerationNode CounterSelector
- quickSpinIntegerNode CounterValueAtReset
- quickSpinEnumerationNode CounterStatus
- quickSpinEnumerationNode CounterTriggerSource
- quickSpinIntegerNode CounterDelay
- quickSpinEnumerationNode CounterResetSource
- quickSpinEnumerationNode CounterEventSource
- quickSpinEnumerationNode CounterEventActivation
- quickSpinIntegerNode CounterDuration
- quickSpinEnumerationNode CounterResetActivation

- quickSpinEnumerationNode DeviceType
- quickSpinStringNode DeviceFamilyName
- quickSpinIntegerNode DeviceSFNCVersionMajor
- quickSpinIntegerNode DeviceSFNCVersionMinor
- quickSpinIntegerNode DeviceSFNCVersionSubMinor
- quickSpinIntegerNode DeviceManifestEntrySelector
- quickSpinIntegerNode DeviceManifestXMLMajorVersion
- quickSpinIntegerNode DeviceManifestXMLMinorVersion
- quickSpinIntegerNode DeviceManifestXMLSubMinorVersion
- quickSpinIntegerNode DeviceManifestSchemaMajorVersion
- quickSpinIntegerNode DeviceManifestSchemaMinorVersion
- quickSpinStringNode DeviceManifestPrimaryURL
- quickSpinStringNode DeviceManifestSecondaryURL
- quickSpinIntegerNode DeviceTLVersionSubMinor
- quickSpinIntegerNode DeviceGenCPVersionMajor
- quickSpinIntegerNode DeviceGenCPVersionMinor
- quickSpinIntegerNode DeviceConnectionSelector
- quickSpinIntegerNode DeviceConnectionSpeed
- quickSpinEnumerationNode DeviceConnectionStatus
- quickSpinIntegerNode DeviceLinkSelector
- quickSpinEnumerationNode DeviceLinkThroughputLimitMode
- quickSpinIntegerNode DeviceLinkConnectionCount
- quickSpinEnumerationNode DeviceLinkHeartbeatMode
- quickSpinFloatNode DeviceLinkHeartbeatTimeout
- quickSpinFloatNode DeviceLinkCommandTimeout
- quickSpinIntegerNode DeviceStreamChannelSelector
- quickSpinEnumerationNode DeviceStreamChannelType
- quickSpinIntegerNode DeviceStreamChannelLink
- quickSpinEnumerationNode DeviceStreamChannelEndianness
- quickSpinIntegerNode DeviceStreamChannelPacketSize
- quickSpinCommandNode DeviceFeaturePersistenceStart
- quickSpinCommandNode DeviceFeaturePersistenceEnd
- quickSpinCommandNode DeviceRegistersStreamingStart
- quickSpinCommandNode DeviceRegistersStreamingEnd
- quickSpinCommandNode DeviceRegistersCheck
- quickSpinBooleanNode DeviceRegistersValid
- quickSpinEnumerationNode DeviceClockSelector
- quickSpinFloatNode DeviceClockFrequency
- quickSpinEnumerationNode DeviceSerialPortSelector
- quickSpinEnumerationNode DeviceSerialPortBaudRate
- quickSpinIntegerNode Timestamp
- quickSpinEnumerationNode SensorTaps
- quickSpinEnumerationNode SensorDigitizationTaps
- quickSpinEnumerationNode RegionSelector
- quickSpinEnumerationNode RegionMode
- quickSpinEnumerationNode RegionDestination
- quickSpinEnumerationNode ImageComponentSelector
- quickSpinBooleanNode ImageComponentEnable
- quickSpinIntegerNode LinePitch
- quickSpinEnumerationNode PixelFormatInfoSelector
- quickSpinIntegerNode PixelFormatInfoID
- quickSpinEnumerationNode Deinterlacing
- quickSpinEnumerationNode ImageCompressionRateOption
- quickSpinIntegerNode ImageCompressionQuality
- quickSpinFloatNode ImageCompressionBitrate

- quickSpinEnumerationNode ImageCompressionJPEGFormatOption
- quickSpinCommandNode AcquisitionAbort
- quickSpinCommandNode AcquisitionArm
- quickSpinEnumerationNode AcquisitionStatusSelector
- quickSpinBooleanNode AcquisitionStatus
- quickSpinIntegerNode TriggerDivider
- quickSpinIntegerNode TriggerMultiplier
- quickSpinEnumerationNode ExposureTimeMode
- quickSpinEnumerationNode ExposureTimeSelector
- quickSpinEnumerationNode GainAutoBalance
- quickSpinEnumerationNode BlackLevelAuto
- quickSpinEnumerationNode BlackLevelAutoBalance
- quickSpinEnumerationNode WhiteClipSelector
- quickSpinFloatNode WhiteClip
- quickSpinRegisterNode LUTValueAll
- quickSpinIntegerNode UserOutputValueAllMask
- quickSpinCommandNode CounterReset
- quickSpinEnumerationNode TimerSelector
- quickSpinFloatNode TimerDuration
- quickSpinFloatNode TimerDelay
- quickSpinCommandNode TimerReset
- quickSpinFloatNode TimerValue
- quickSpinEnumerationNode TimerStatus
- quickSpinEnumerationNode TimerTriggerSource
- quickSpinEnumerationNode TimerTriggerActivation
- quickSpinEnumerationNode EncoderSelector
- quickSpinEnumerationNode EncoderSourceA
- quickSpinEnumerationNode EncoderSourceB
- quickSpinEnumerationNode EncoderMode
- quickSpinIntegerNode EncoderDivider
- quickSpinEnumerationNode EncoderOutputMode
- quickSpinEnumerationNode EncoderStatus
- quickSpinFloatNode EncoderTimeout
- quickSpinEnumerationNode EncoderResetSource
- quickSpinEnumerationNode EncoderResetActivation
- quickSpinCommandNode EncoderReset
- quickSpinIntegerNode EncoderValue
- quickSpinIntegerNode EncoderValueAtReset
- quickSpinEnumerationNode SoftwareSignalSelector
- quickSpinCommandNode SoftwareSignalPulse
- quickSpinEnumerationNode ActionUnconditionalMode
- quickSpinIntegerNode ActionDeviceKey
- quickSpinIntegerNode ActionQueueSize
- quickSpinIntegerNode ActionSelector
- quickSpinIntegerNode ActionGroupMask
- quickSpinIntegerNode ActionGroupKey
- quickSpinIntegerNode EventAcquisitionTrigger
- quickSpinIntegerNode EventAcquisitionTriggerTimestamp
- quickSpinIntegerNode EventAcquisitionTriggerFrameID
- quickSpinIntegerNode EventAcquisitionStart
- quickSpinIntegerNode EventAcquisitionStartTimestamp
- quickSpinIntegerNode EventAcquisitionStartFrameID
- quickSpinIntegerNode EventAcquisitionEnd
- quickSpinIntegerNode EventAcquisitionEndTimestamp
- quickSpinIntegerNode EventAcquisitionEndFrameID

- quickSpinIntegerNode EventAcquisitionTransferStart
- quickSpinIntegerNode EventAcquisitionTransferStartTimestamp
- quickSpinIntegerNode EventAcquisitionTransferStartFrameID
- quickSpinIntegerNode EventAcquisitionTransferEnd
- quickSpinIntegerNode EventAcquisitionTransferEndTimestamp
- quickSpinIntegerNode EventAcquisitionTransferEndFrameID
- quickSpinIntegerNode EventAcquisitionError
- quickSpinIntegerNode EventAcquisitionErrorTimestamp
- quickSpinIntegerNode EventAcquisitionErrorFrameID
- quickSpinIntegerNode EventFrameTrigger
- quickSpinIntegerNode EventFrameTriggerTimestamp
- quickSpinIntegerNode EventFrameTriggerFrameID
- quickSpinIntegerNode EventFrameStart
- quickSpinIntegerNode EventFrameStartTimestamp
- quickSpinIntegerNode EventFrameStartFrameID
- quickSpinIntegerNode EventFrameEnd
- quickSpinIntegerNode EventFrameEndTimestamp
- quickSpinIntegerNode EventFrameEndFrameID
- quickSpinIntegerNode EventFrameBurstStart
- quickSpinIntegerNode EventFrameBurstStartTimestamp
- quickSpinIntegerNode EventFrameBurstStartFrameID
- quickSpinIntegerNode EventFrameBurstEnd
- quickSpinIntegerNode EventFrameBurstEndTimestamp
- quickSpinIntegerNode EventFrameBurstEndFrameID
- quickSpinIntegerNode EventFrameTransferStart
- quickSpinIntegerNode EventFrameTransferStartTimestamp
- quickSpinIntegerNode EventFrameTransferStartFrameID
- quickSpinIntegerNode EventFrameTransferEnd
- quickSpinIntegerNode EventFrameTransferEndTimestamp
- quickSpinIntegerNode EventFrameTransferEndFrameID
- quickSpinIntegerNode EventExposureStart
- quickSpinIntegerNode EventExposureStartTimestamp
- quickSpinIntegerNode EventExposureStartFrameID
- quickSpinIntegerNode EventStream0TransferStart
- quickSpinIntegerNode EventStream0TransferStartTimestamp
- quickSpinIntegerNode EventStream0TransferStartFrameID
- quickSpinIntegerNode EventStream0TransferEnd
- quickSpinIntegerNode EventStream0TransferEndTimestamp
- quickSpinIntegerNode EventStream0TransferEndFrameID
- quickSpinIntegerNode EventStream0TransferPause
- quickSpinIntegerNode EventStream0TransferPauseTimestamp
- quickSpinIntegerNode EventStream0TransferPauseFrameID
- quickSpinIntegerNode EventStream0TransferResume
- quickSpinIntegerNode EventStream0TransferResumeTimestamp
- quickSpinIntegerNode EventStream0TransferResumeFrameID
- quickSpinIntegerNode EventStream0TransferBlockStart
- quickSpinIntegerNode EventStream0TransferBlockStartTimestamp
- quickSpinIntegerNode EventStream0TransferBlockStartFrameID
- quickSpinIntegerNode EventStream0TransferBlockEnd
- quickSpinIntegerNode EventStream0TransferBlockEndTimestamp
- quickSpinIntegerNode EventStream0TransferBlockEndFrameID
- quickSpinIntegerNode EventStream0TransferBlockTrigger
- quickSpinIntegerNode EventStream0TransferBlockTriggerTimestamp
- quickSpinIntegerNode EventStream0TransferBlockTriggerFrameID
- quickSpinIntegerNode EventStream0TransferBurstStart

- quickSpinIntegerNode EventStream0TransferBurstStartTimestamp
- quickSpinIntegerNode EventStream0TransferBurstStartFrameID
- quickSpinIntegerNode EventStream0TransferBurstEnd
- quickSpinIntegerNode EventStream0TransferBurstEndTimestamp
- quickSpinIntegerNode EventStream0TransferBurstEndFrameID
- quickSpinIntegerNode EventStream0TransferOverflow
- quickSpinIntegerNode EventStream0TransferOverflowTimestamp
- quickSpinIntegerNode EventStream0TransferOverflowFrameID
- quickSpinIntegerNode EventSequencerSetChange
- quickSpinIntegerNode EventSequencerSetChangeTimestamp
- quickSpinIntegerNode EventSequencerSetChangeFrameID
- quickSpinIntegerNode EventCounter0Start
- quickSpinIntegerNode EventCounter0StartTimestamp
- quickSpinIntegerNode EventCounter0StartFrameID
- quickSpinIntegerNode EventCounter1Start
- quickSpinIntegerNode EventCounter1StartTimestamp
- quickSpinIntegerNode EventCounter1StartFrameID
- quickSpinIntegerNode EventCounter0End
- quickSpinIntegerNode EventCounter0EndTimestamp
- quickSpinIntegerNode EventCounter0EndFrameID
- quickSpinIntegerNode EventCounter1End
- quickSpinIntegerNode EventCounter1EndTimestamp
- quickSpinIntegerNode EventCounter1EndFrameID
- quickSpinIntegerNode EventTimer0Start
- quickSpinIntegerNode EventTimer0StartTimestamp
- quickSpinIntegerNode EventTimer0StartFrameID
- quickSpinIntegerNode EventTimer1Start
- quickSpinIntegerNode EventTimer1StartTimestamp
- quickSpinIntegerNode EventTimer1StartFrameID
- quickSpinIntegerNode EventTimer0End
- quickSpinIntegerNode EventTimer0EndTimestamp
- quickSpinIntegerNode EventTimer0EndFrameID
- quickSpinIntegerNode EventTimer1End
- quickSpinIntegerNode EventTimer1EndTimestamp
- quickSpinIntegerNode EventTimer1EndFrameID
- quickSpinIntegerNode EventEncoder0Stopped
- quickSpinIntegerNode EventEncoder0StoppedTimestamp
- quickSpinIntegerNode EventEncoder0StoppedFrameID
- quickSpinIntegerNode EventEncoder1Stopped
- quickSpinIntegerNode EventEncoder1StoppedTimestamp
- quickSpinIntegerNode EventEncoder1StoppedFrameID
- quickSpinIntegerNode EventEncoder0Restarted
- quickSpinIntegerNode EventEncoder0RestartedTimestamp
- quickSpinIntegerNode EventEncoder0RestartedFrameID
- quickSpinIntegerNode EventEncoder1Restarted
- quickSpinIntegerNode EventEncoder1RestartedTimestamp
- quickSpinIntegerNode EventEncoder1RestartedFrameID
- quickSpinIntegerNode EventLine0RisingEdge
- quickSpinIntegerNode EventLine0RisingEdgeTimestamp
- quickSpinIntegerNode EventLine0RisingEdgeFrameID
- quickSpinIntegerNode EventLine1RisingEdge
- quickSpinIntegerNode EventLine1RisingEdgeTimestamp
- quickSpinIntegerNode EventLine1RisingEdgeFrameID
- quickSpinIntegerNode EventLine0FallingEdge
- quickSpinIntegerNode EventLine0FallingEdgeTimestamp

- quickSpinIntegerNode EventLine0FallingEdgeFrameID
- quickSpinIntegerNode EventLine1FallingEdge
- quickSpinIntegerNode EventLine1FallingEdgeTimestamp
- quickSpinIntegerNode EventLine1FallingEdgeFrameID
- quickSpinIntegerNode EventLine0AnyEdge
- quickSpinIntegerNode EventLine0AnyEdgeTimestamp
- quickSpinIntegerNode EventLine0AnyEdgeFrameID
- quickSpinIntegerNode EventLine1AnyEdge
- quickSpinIntegerNode EventLine1AnyEdgeTimestamp
- quickSpinIntegerNode EventLine1AnyEdgeFrameID
- quickSpinIntegerNode EventLinkTrigger0
- quickSpinIntegerNode EventLinkTrigger0Timestamp
- quickSpinIntegerNode EventLinkTrigger0FrameID
- quickSpinIntegerNode EventLinkTrigger1
- quickSpinIntegerNode EventLinkTrigger1Timestamp
- quickSpinIntegerNode EventLinkTrigger1FrameID
- quickSpinIntegerNode EventActionLate
- quickSpinIntegerNode EventActionLateTimestamp
- quickSpinIntegerNode EventActionLateFrameID
- quickSpinIntegerNode EventLinkSpeedChange
- quickSpinIntegerNode EventLinkSpeedChangeTimestamp
- quickSpinIntegerNode EventLinkSpeedChangeFrameID
- quickSpinRegisterNode FileAccessBuffer
- quickSpinIntegerNode SourceCount
- quickSpinEnumerationNode SourceSelector
- quickSpinEnumerationNode TransferSelector
- quickSpinIntegerNode TransferBurstCount
- quickSpinCommandNode TransferAbort
- quickSpinCommandNode TransferPause
- quickSpinCommandNode TransferResume
- quickSpinEnumerationNode TransferTriggerSelector
- quickSpinEnumerationNode TransferTriggerMode
- quickSpinEnumerationNode TransferTriggerSource
- quickSpinEnumerationNode TransferTriggerActivation
- quickSpinEnumerationNode TransferStatusSelector
- quickSpinBooleanNode TransferStatus
- quickSpinEnumerationNode TransferComponentSelector
- quickSpinIntegerNode TransferStreamChannel
- quickSpinEnumerationNode Scan3dDistanceUnit
- quickSpinEnumerationNode Scan3dCoordinateSystem
- quickSpinEnumerationNode Scan3dOutputMode
- quickSpinEnumerationNode Scan3dCoordinateSystemReference
- quickSpinEnumerationNode Scan3dCoordinateSelector
- quickSpinFloatNode Scan3dCoordinateScale
- quickSpinFloatNode Scan3dCoordinateOffset
- quickSpinBooleanNode Scan3dInvalidDataFlag
- quickSpinFloatNode Scan3dInvalidDataValue
- quickSpinFloatNode Scan3dAxisMin
- quickSpinFloatNode Scan3dAxisMax
- quickSpinEnumerationNode Scan3dCoordinateTransformSelector
- quickSpinFloatNode Scan3dTransformValue
- quickSpinEnumerationNode Scan3dCoordinateReferenceSelector
- quickSpinFloatNode Scan3dCoordinateReferenceValue
- quickSpinIntegerNode ChunkPartSelector
- quickSpinEnumerationNode ChunkImageComponent

- quickSpinIntegerNode ChunkPixelDynamicRangeMin
- quickSpinIntegerNode ChunkPixelDynamicRangeMax
- quickSpinIntegerNode ChunkTimestampLatchValue
- quickSpinIntegerNode ChunkLineStatusAll
- quickSpinEnumerationNode ChunkCounterSelector
- quickSpinIntegerNode ChunkCounterValue
- quickSpinEnumerationNode ChunkTimerSelector
- quickSpinFloatNode ChunkTimerValue
- quickSpinEnumerationNode ChunkEncoderSelector
- quickSpinIntegerNode ChunkScanLineSelector
- quickSpinIntegerNode ChunkEncoderValue
- quickSpinEnumerationNode ChunkEncoderStatus
- quickSpinEnumerationNode ChunkExposureTimeSelector
- quickSpinIntegerNode ChunkLinePitch
- quickSpinEnumerationNode ChunkSourceID
- quickSpinEnumerationNode ChunkRegionID
- quickSpinIntegerNode ChunkTransferBlockID
- quickSpinEnumerationNode ChunkTransferStreamID
- quickSpinIntegerNode ChunkTransferQueueCurrentBlockCount
- quickSpinIntegerNode ChunkStreamChannelID
- quickSpinEnumerationNode ChunkScan3dDistanceUnit
- quickSpinEnumerationNode ChunkScan3dOutputMode
- quickSpinEnumerationNode ChunkScan3dCoordinateSystem
- quickSpinEnumerationNode ChunkScan3dCoordinateSystemReference
- quickSpinEnumerationNode ChunkScan3dCoordinateSelector
- quickSpinFloatNode ChunkScan3dCoordinateScale
- quickSpinFloatNode ChunkScan3dCoordinateOffset
- quickSpinBooleanNode ChunkScan3dInvalidDataFlag
- quickSpinFloatNode ChunkScan3dInvalidDataValue
- quickSpinFloatNode ChunkScan3dAxisMin
- quickSpinFloatNode ChunkScan3dAxisMax
- quickSpinEnumerationNode ChunkScan3dCoordinateTransformSelector
- quickSpinFloatNode ChunkScan3dTransformValue
- quickSpinEnumerationNode ChunkScan3dCoordinateReferenceSelector
- quickSpinFloatNode ChunkScan3dCoordinateReferenceValue
- quickSpinIntegerNode TestPendingAck
- quickSpinEnumerationNode DeviceTapGeometry
- quickSpinEnumerationNode GevPhysicalLinkConfiguration
- quickSpinEnumerationNode GevCurrentPhysicalLinkConfiguration
- quickSpinIntegerNode GevActiveLinkCount
- quickSpinBooleanNode GevPAUSEFrameReception
- quickSpinBooleanNode GevPAUSEFrameTransmission
- quickSpinEnumerationNode GevIPConfigurationStatus
- quickSpinIntegerNode GevDiscoveryAckDelay
- quickSpinEnumerationNode GevGVCPExtendedStatusCodesSelector
- quickSpinBooleanNode GevGVCPExtendedStatusCodes
- quickSpinIntegerNode GevPrimaryApplicationSwitchoverKey
- quickSpinEnumerationNode GevGVSPExtendedIDMode
- quickSpinIntegerNode GevPrimaryApplicationSocket
- quickSpinIntegerNode GevPrimaryApplicationIPAddress
- quickSpinBooleanNode GevSCCFGPacketResendDestination
- quickSpinBooleanNode GevSCCFGAllInTransmission
- quickSpinIntegerNode GevSCZoneCount
- quickSpinIntegerNode GevSCZoneDirectionAll
- quickSpinBooleanNode GevSCZoneConfigurationLock

- quickSpinIntegerNode aPAUSEMACCtrlFramesTransmitted
- quickSpinIntegerNode aPAUSEMACCtrlFramesReceived
- quickSpinEnumerationNode ClConfiguration
- quickSpinEnumerationNode ClTimeSlotsCount
- quickSpinEnumerationNode CxpLinkConfigurationStatus
- quickSpinEnumerationNode CxpLinkConfigurationPreferred
- quickSpinEnumerationNode CxpLinkConfiguration
- quickSpinIntegerNode CxpConnectionSelector
- quickSpinEnumerationNode CxpConnectionTestMode
- quickSpinIntegerNode CxpConnectionTestErrorCount
- quickSpinIntegerNode CxpConnectionTestPacketCount
- quickSpinCommandNode CxpPoCxpAuto
- quickSpinCommandNode CxpPoCxpTurnOff
- quickSpinCommandNode CxpPoCxpTripReset
- quickSpinEnumerationNode CxpPoCxpStatus
- quickSpinIntegerNode ChunkInferenceFrameId
- quickSpinIntegerNode ChunkInferenceResult
- quickSpinFloatNode ChunkInferenceConfidence
- quickSpinRegisterNode ChunkInferenceBoundingBoxResult

12.2.1 Field Documentation

12.2.1.1 AasRoiEnable

```
quickSpinBooleanNode AasRoiEnable
```

12.2.1.2 AasRoiHeight

```
quickSpinIntegerNode AasRoiHeight
```

12.2.1.3 AasRoiOffsetX

```
quickSpinIntegerNode AasRoiOffsetX
```

12.2.1.4 AasRoiOffsetY

```
quickSpinIntegerNode AasRoiOffsetY
```

12.2.1.5 AasRoiWidth

```
quickSpinIntegerNode AasRoiWidth
```

12.2.1.6 AcquisitionAbort

```
quickSpinCommandNode AcquisitionAbort
```

12.2.1.7 AcquisitionArm

```
quickSpinCommandNode AcquisitionArm
```

12.2.1.8 AcquisitionBurstFrameCount

```
quickSpinIntegerNode AcquisitionBurstFrameCount
```

12.2.1.9 AcquisitionFrameCount

```
quickSpinIntegerNode AcquisitionFrameCount
```

12.2.1.10 AcquisitionFrameRate

```
quickSpinFloatNode AcquisitionFrameRate
```

12.2.1.11 AcquisitionFrameRateEnable

```
quickSpinBooleanNode AcquisitionFrameRateEnable
```

12.2.1.12 AcquisitionLineRate

```
quickSpinFloatNode AcquisitionLineRate
```

12.2.1.13 AcquisitionMode

`quickSpinEnumerationNode AcquisitionMode`

12.2.1.14 AcquisitionResultingFrameRate

`quickSpinFloatNode AcquisitionResultingFrameRate`

12.2.1.15 AcquisitionStart

`quickSpinCommandNode AcquisitionStart`

12.2.1.16 AcquisitionStatus

`quickSpinBooleanNode AcquisitionStatus`

12.2.1.17 AcquisitionStatusSelector

`quickSpinEnumerationNode AcquisitionStatusSelector`

12.2.1.18 AcquisitionStop

`quickSpinCommandNode AcquisitionStop`

12.2.1.19 ActionDeviceKey

`quickSpinIntegerNode ActionDeviceKey`

12.2.1.20 ActionGroupKey

`quickSpinIntegerNode ActionGroupKey`

12.2.1.21 ActionGroupMask

`quickSpinIntegerNode ActionGroupMask`

12.2.1.22 ActionQueueSize

`quickSpinIntegerNode ActionQueueSize`

12.2.1.23 ActionSelector

`quickSpinIntegerNode ActionSelector`

12.2.1.24 ActionUnconditionalMode

`quickSpinEnumerationNode ActionUnconditionalMode`

12.2.1.25 AdaptiveCompressionEnable

`quickSpinBooleanNode AdaptiveCompressionEnable`

12.2.1.26 AdcBitDepth

`quickSpinEnumerationNode AdcBitDepth`

12.2.1.27 aPAUSEMACCtrlFramesReceived

`quickSpinIntegerNode aPAUSEMACCtrlFramesReceived`

12.2.1.28 aPAUSEMACCtrlFramesTransmitted

`quickSpinIntegerNode aPAUSEMACCtrlFramesTransmitted`

12.2.1.29 AutoAlgorithmSelector

```
quickSpinEnumerationNode AutoAlgorithmSelector
```

12.2.1.30 AutoExposureControlLoopDamping

```
quickSpinFloatNode AutoExposureControlLoopDamping
```

12.2.1.31 AutoExposureControlPriority

```
quickSpinEnumerationNode AutoExposureControlPriority
```

12.2.1.32 AutoExposureEVCompensation

```
quickSpinFloatNode AutoExposureEVCompensation
```

12.2.1.33 AutoExposureExposureTimeLowerLimit

```
quickSpinFloatNode AutoExposureExposureTimeLowerLimit
```

12.2.1.34 AutoExposureExposureTimeUpperLimit

```
quickSpinFloatNode AutoExposureExposureTimeUpperLimit
```

12.2.1.35 AutoExposureGainLowerLimit

```
quickSpinFloatNode AutoExposureGainLowerLimit
```

12.2.1.36 AutoExposureGainUpperLimit

```
quickSpinFloatNode AutoExposureGainUpperLimit
```

12.2.1.37 AutoExposureGreyValueLowerLimit

```
quickSpinFloatNode AutoExposureGreyValueLowerLimit
```

12.2.1.38 AutoExposureGreyValueUpperLimit

```
quickSpinFloatNode AutoExposureGreyValueUpperLimit
```

12.2.1.39 AutoExposureLightingMode

```
quickSpinEnumerationNode AutoExposureLightingMode
```

12.2.1.40 AutoExposureMeteringMode

```
quickSpinEnumerationNode AutoExposureMeteringMode
```

12.2.1.41 AutoExposureTargetGreyValue

```
quickSpinFloatNode AutoExposureTargetGreyValue
```

12.2.1.42 AutoExposureTargetGreyValueAuto

```
quickSpinEnumerationNode AutoExposureTargetGreyValueAuto
```

12.2.1.43 BalanceRatio

```
quickSpinFloatNode BalanceRatio
```

12.2.1.44 BalanceRatioSelector

```
quickSpinEnumerationNode BalanceRatioSelector
```

12.2.1.45 BalanceWhiteAuto

```
quickSpinEnumerationNode BalanceWhiteAuto
```

12.2.1.46 BalanceWhiteAutoDamping

```
quickSpinFloatNode BalanceWhiteAutoDamping
```

12.2.1.47 BalanceWhiteAutoLowerLimit

```
quickSpinFloatNode BalanceWhiteAutoLowerLimit
```

12.2.1.48 BalanceWhiteAutoProfile

```
quickSpinEnumerationNode BalanceWhiteAutoProfile
```

12.2.1.49 BalanceWhiteAutoUpperLimit

```
quickSpinFloatNode BalanceWhiteAutoUpperLimit
```

12.2.1.50 BinningHorizontal

```
quickSpinIntegerNode BinningHorizontal
```

12.2.1.51 BinningHorizontalMode

```
quickSpinEnumerationNode BinningHorizontalMode
```

12.2.1.52 BinningSelector

```
quickSpinEnumerationNode BinningSelector
```

12.2.1.53 BinningVertical

`quickSpinIntegerNode BinningVertical`

12.2.1.54 BinningVerticalMode

`quickSpinEnumerationNode BinningVerticalMode`

12.2.1.55 BlackLevel

`quickSpinFloatNode BlackLevel`

12.2.1.56 BlackLevelAuto

`quickSpinEnumerationNode BlackLevelAuto`

12.2.1.57 BlackLevelAutoBalance

`quickSpinEnumerationNode BlackLevelAutoBalance`

12.2.1.58 BlackLevelClampingEnable

`quickSpinBooleanNode BlackLevelClampingEnable`

12.2.1.59 BlackLevelRaw

`quickSpinIntegerNode BlackLevelRaw`

12.2.1.60 BlackLevelSelector

`quickSpinEnumerationNode BlackLevelSelector`

12.2.1.61 ChunkBlackLevel

`quickSpinFloatNode ChunkBlackLevel`

12.2.1.62 ChunkBlackLevelSelector

`quickSpinEnumerationNode ChunkBlackLevelSelector`

12.2.1.63 ChunkCompressionMode

`quickSpinIntegerNode ChunkCompressionMode`

12.2.1.64 ChunkCompressionRatio

`quickSpinFloatNode ChunkCompressionRatio`

12.2.1.65 ChunkCounterSelector

`quickSpinEnumerationNode ChunkCounterSelector`

12.2.1.66 ChunkCounterValue

`quickSpinIntegerNode ChunkCounterValue`

12.2.1.67 ChunkCRC

`quickSpinIntegerNode ChunkCRC`

12.2.1.68 ChunkEnable

`quickSpinBooleanNode ChunkEnable`

12.2.1.69 ChunkEncoderSelector

```
quickSpinEnumerationNode ChunkEncoderSelector
```

12.2.1.70 ChunkEncoderStatus

```
quickSpinEnumerationNode ChunkEncoderStatus
```

12.2.1.71 ChunkEncoderValue

```
quickSpinIntegerNode ChunkEncoderValue
```

12.2.1.72 ChunkExposureEndLineStatusAll

```
quickSpinIntegerNode ChunkExposureEndLineStatusAll
```

12.2.1.73 ChunkExposureTime

```
quickSpinFloatNode ChunkExposureTime
```

12.2.1.74 ChunkExposureTimeSelector

```
quickSpinEnumerationNode ChunkExposureTimeSelector
```

12.2.1.75 ChunkFrameID

```
quickSpinIntegerNode ChunkFrameID
```

12.2.1.76 ChunkGain

```
quickSpinFloatNode ChunkGain
```

12.2.1.77 ChunkGainSelector

`quickSpinEnumerationNode ChunkGainSelector`

12.2.1.78 ChunkHeight

`quickSpinIntegerNode ChunkHeight`

12.2.1.79 ChunkImage

`quickSpinIntegerNode ChunkImage`

12.2.1.80 ChunkImageComponent

`quickSpinEnumerationNode ChunkImageComponent`

12.2.1.81 ChunkInferenceBoundingBoxResult

`quickSpinRegisterNode ChunkInferenceBoundingBoxResult`

12.2.1.82 ChunkInferenceConfidence

`quickSpinFloatNode ChunkInferenceConfidence`

12.2.1.83 ChunkInferenceFrameId

`quickSpinIntegerNode ChunkInferenceFrameId`

12.2.1.84 ChunkInferenceResult

`quickSpinIntegerNode ChunkInferenceResult`

12.2.1.85 ChunkLinePitch

```
quickSpinIntegerNode ChunkLinePitch
```

12.2.1.86 ChunkLineStatusAll

```
quickSpinIntegerNode ChunkLineStatusAll
```

12.2.1.87 ChunkModeActive

```
quickSpinBooleanNode ChunkModeActive
```

12.2.1.88 ChunkOffsetX

```
quickSpinIntegerNode ChunkOffsetX
```

12.2.1.89 ChunkOffsetY

```
quickSpinIntegerNode ChunkOffsetY
```

12.2.1.90 ChunkPartSelector

```
quickSpinIntegerNode ChunkPartSelector
```

12.2.1.91 ChunkPixelDynamicRangeMax

```
quickSpinIntegerNode ChunkPixelDynamicRangeMax
```

12.2.1.92 ChunkPixelDynamicRangeMin

```
quickSpinIntegerNode ChunkPixelDynamicRangeMin
```

12.2.1.93 ChunkPixelFormat

`quickSpinEnumerationNode ChunkPixelFormat`

12.2.1.94 ChunkRegionID

`quickSpinEnumerationNode ChunkRegionID`

12.2.1.95 ChunkScan3dAxisMax

`quickSpinFloatNode ChunkScan3dAxisMax`

12.2.1.96 ChunkScan3dAxisMin

`quickSpinFloatNode ChunkScan3dAxisMin`

12.2.1.97 ChunkScan3dCoordinateOffset

`quickSpinFloatNode ChunkScan3dCoordinateOffset`

12.2.1.98 ChunkScan3dCoordinateReferenceSelector

`quickSpinEnumerationNode ChunkScan3dCoordinateReferenceSelector`

12.2.1.99 ChunkScan3dCoordinateReferenceValue

`quickSpinFloatNode ChunkScan3dCoordinateReferenceValue`

12.2.1.100 ChunkScan3dCoordinateScale

`quickSpinFloatNode ChunkScan3dCoordinateScale`

12.2.1.101 ChunkScan3dCoordinateSelector

```
quickSpinEnumerationNode ChunkScan3dCoordinateSelector
```

12.2.1.102 ChunkScan3dCoordinateSystem

```
quickSpinEnumerationNode ChunkScan3dCoordinateSystem
```

12.2.1.103 ChunkScan3dCoordinateSystemReference

```
quickSpinEnumerationNode ChunkScan3dCoordinateSystemReference
```

12.2.1.104 ChunkScan3dCoordinateTransformSelector

```
quickSpinEnumerationNode ChunkScan3dCoordinateTransformSelector
```

12.2.1.105 ChunkScan3dDistanceUnit

```
quickSpinEnumerationNode ChunkScan3dDistanceUnit
```

12.2.1.106 ChunkScan3dInvalidDataFlag

```
quickSpinBooleanNode ChunkScan3dInvalidDataFlag
```

12.2.1.107 ChunkScan3dInvalidDataValue

```
quickSpinFloatNode ChunkScan3dInvalidDataValue
```

12.2.1.108 ChunkScan3dOutputMode

```
quickSpinEnumerationNode ChunkScan3dOutputMode
```

12.2.1.109 ChunkScan3dTransformValue

```
quickSpinFloatNode ChunkScan3dTransformValue
```

12.2.1.110 ChunkScanLineSelector

```
quickSpinIntegerNode ChunkScanLineSelector
```

12.2.1.111 ChunkSelector

```
quickSpinEnumerationNode ChunkSelector
```

12.2.1.112 ChunkSequencerSetActive

```
quickSpinIntegerNode ChunkSequencerSetActive
```

12.2.1.113 ChunkSerialData

```
quickSpinStringNode ChunkSerialData
```

12.2.1.114 ChunkSerialDataLength

```
quickSpinIntegerNode ChunkSerialDataLength
```

12.2.1.115 ChunkSerialReceiveOverflow

```
quickSpinBooleanNode ChunkSerialReceiveOverflow
```

12.2.1.116 ChunkSourceID

```
quickSpinEnumerationNode ChunkSourceID
```

12.2.1.117 ChunkStreamChannelID

```
quickSpinIntegerNode ChunkStreamChannelID
```

12.2.1.118 ChunkTimerSelector

```
quickSpinEnumerationNode ChunkTimerSelector
```

12.2.1.119 ChunkTimerValue

```
quickSpinFloatNode ChunkTimerValue
```

12.2.1.120 ChunkTimestamp

```
quickSpinIntegerNode ChunkTimestamp
```

12.2.1.121 ChunkTimestampLatchValue

```
quickSpinIntegerNode ChunkTimestampLatchValue
```

12.2.1.122 ChunkTransferBlockID

```
quickSpinIntegerNode ChunkTransferBlockID
```

12.2.1.123 ChunkTransferQueueCurrentBlockCount

```
quickSpinIntegerNode ChunkTransferQueueCurrentBlockCount
```

12.2.1.124 ChunkTransferStreamID

```
quickSpinEnumerationNode ChunkTransferStreamID
```

12.2.1.125 ChunkWidth

```
quickSpinIntegerNode ChunkWidth
```

12.2.1.126 ClConfiguration

```
quickSpinEnumerationNode ClConfiguration
```

12.2.1.127 ClTimeSlotsCount

```
quickSpinEnumerationNode ClTimeSlotsCount
```

12.2.1.128 ColorTransformationEnable

```
quickSpinBooleanNode ColorTransformationEnable
```

12.2.1.129 ColorTransformationSelector

```
quickSpinEnumerationNode ColorTransformationSelector
```

12.2.1.130 ColorTransformationValue

```
quickSpinFloatNode ColorTransformationValue
```

12.2.1.131 ColorTransformationValueSelector

```
quickSpinEnumerationNode ColorTransformationValueSelector
```

12.2.1.132 CompressionRatio

```
quickSpinFloatNode CompressionRatio
```

12.2.1.133 CompressionSaturationPriority

```
quickSpinEnumerationNode CompressionSaturationPriority
```

12.2.1.134 CounterDelay

```
quickSpinIntegerNode CounterDelay
```

12.2.1.135 CounterDuration

```
quickSpinIntegerNode CounterDuration
```

12.2.1.136 CounterEventActivation

```
quickSpinEnumerationNode CounterEventActivation
```

12.2.1.137 CounterEventSource

```
quickSpinEnumerationNode CounterEventSource
```

12.2.1.138 CounterReset

```
quickSpinCommandNode CounterReset
```

12.2.1.139 CounterResetActivation

```
quickSpinEnumerationNode CounterResetActivation
```

12.2.1.140 CounterResetSource

```
quickSpinEnumerationNode CounterResetSource
```

12.2.1.141 CounterSelector

`quickSpinEnumerationNode CounterSelector`

12.2.1.142 CounterStatus

`quickSpinEnumerationNode CounterStatus`

12.2.1.143 CounterTriggerActivation

`quickSpinEnumerationNode CounterTriggerActivation`

12.2.1.144 CounterTriggerSource

`quickSpinEnumerationNode CounterTriggerSource`

12.2.1.145 CounterValue

`quickSpinIntegerNode CounterValue`

12.2.1.146 CounterValueAtReset

`quickSpinIntegerNode CounterValueAtReset`

12.2.1.147 CxpConnectionSelector

`quickSpinIntegerNode CxpConnectionSelector`

12.2.1.148 CxpConnectionTestErrorCount

`quickSpinIntegerNode CxpConnectionTestErrorCount`

12.2.1.149 CxpConnectionTestMode

```
quickSpinEnumerationNode CxpConnectionTestMode
```

12.2.1.150 CxpConnectionTestPacketCount

```
quickSpinIntegerNode CxpConnectionTestPacketCount
```

12.2.1.151 CxpLinkConfiguration

```
quickSpinEnumerationNode CxpLinkConfiguration
```

12.2.1.152 CxpLinkConfigurationPreferred

```
quickSpinEnumerationNode CxpLinkConfigurationPreferred
```

12.2.1.153 CxpLinkConfigurationStatus

```
quickSpinEnumerationNode CxpLinkConfigurationStatus
```

12.2.1.154 CxpPoCxpAuto

```
quickSpinCommandNode CxpPoCxpAuto
```

12.2.1.155 CxpPoCxpStatus

```
quickSpinEnumerationNode CxpPoCxpStatus
```

12.2.1.156 CxpPoCxpTripReset

```
quickSpinCommandNode CxpPoCxpTripReset
```

12.2.1.157 CxpPoCxpTurnOff

```
quickSpinCommandNode CxpPoCxpTurnOff
```

12.2.1.158 DecimationHorizontal

```
quickSpinIntegerNode DecimationHorizontal
```

12.2.1.159 DecimationHorizontalMode

```
quickSpinEnumerationNode DecimationHorizontalMode
```

12.2.1.160 DecimationSelector

```
quickSpinEnumerationNode DecimationSelector
```

12.2.1.161 DecimationVertical

```
quickSpinIntegerNode DecimationVertical
```

12.2.1.162 DecimationVerticalMode

```
quickSpinEnumerationNode DecimationVerticalMode
```

12.2.1.163 DefectCorrectionMode

```
quickSpinEnumerationNode DefectCorrectionMode
```

12.2.1.164 DefectCorrectStaticEnable

```
quickSpinBooleanNode DefectCorrectStaticEnable
```

12.2.1.165 DefectTableApply

`quickSpinCommandNode DefectTableApply`

12.2.1.166 DefectTableCoordinateX

`quickSpinIntegerNode DefectTableCoordinateX`

12.2.1.167 DefectTableCoordinateY

`quickSpinIntegerNode DefectTableCoordinateY`

12.2.1.168 DefectTableFactoryRestore

`quickSpinCommandNode DefectTableFactoryRestore`

12.2.1.169 DefectTableIndex

`quickSpinIntegerNode DefectTableIndex`

12.2.1.170 DefectTablePixelCount

`quickSpinIntegerNode DefectTablePixelCount`

12.2.1.171 DefectTableSave

`quickSpinCommandNode DefectTableSave`

12.2.1.172 Deinterlacing

`quickSpinEnumerationNode Deinterlacing`

12.2.1.173 DeviceCharacterSet

```
quickSpinEnumerationNode DeviceCharacterSet
```

12.2.1.174 DeviceClockFrequency

```
quickSpinFloatNode DeviceClockFrequency
```

12.2.1.175 DeviceClockSelector

```
quickSpinEnumerationNode DeviceClockSelector
```

12.2.1.176 DeviceConnectionSelector

```
quickSpinIntegerNode DeviceConnectionSelector
```

12.2.1.177 DeviceConnectionSpeed

```
quickSpinIntegerNode DeviceConnectionSpeed
```

12.2.1.178 DeviceConnectionStatus

```
quickSpinEnumerationNode DeviceConnectionStatus
```

12.2.1.179 DeviceEventChannelCount

```
quickSpinIntegerNode DeviceEventChannelCount
```

12.2.1.180 DeviceFamilyName

```
quickSpinStringNode DeviceFamilyName
```

12.2.1.181 DeviceFeaturePersistenceEnd

```
quickSpinCommandNode DeviceFeaturePersistenceEnd
```

12.2.1.182 DeviceFeaturePersistenceStart

```
quickSpinCommandNode DeviceFeaturePersistenceStart
```

12.2.1.183 DeviceFirmwareVersion

```
quickSpinStringNode DeviceFirmwareVersion
```

12.2.1.184 DeviceGenCPVersionMajor

```
quickSpinIntegerNode DeviceGenCPVersionMajor
```

12.2.1.185 DeviceGenCPVersionMinor

```
quickSpinIntegerNode DeviceGenCPVersionMinor
```

12.2.1.186 DeviceID

```
quickSpinStringNode DeviceID
```

12.2.1.187 DeviceIndicatorMode

```
quickSpinEnumerationNode DeviceIndicatorMode
```

12.2.1.188 DeviceLinkBandwidthReserve

```
quickSpinFloatNode DeviceLinkBandwidthReserve
```

12.2.1.189 DeviceLinkCommandTimeout

```
quickSpinFloatNode DeviceLinkCommandTimeout
```

12.2.1.190 DeviceLinkConnectionCount

```
quickSpinIntegerNode DeviceLinkConnectionCount
```

12.2.1.191 DeviceLinkCurrentThroughput

```
quickSpinIntegerNode DeviceLinkCurrentThroughput
```

12.2.1.192 DeviceLinkHeartbeatMode

```
quickSpinEnumerationNode DeviceLinkHeartbeatMode
```

12.2.1.193 DeviceLinkHeartbeatTimeout

```
quickSpinFloatNode DeviceLinkHeartbeatTimeout
```

12.2.1.194 DeviceLinkSelector

```
quickSpinIntegerNode DeviceLinkSelector
```

12.2.1.195 DeviceLinkSpeed

```
quickSpinIntegerNode DeviceLinkSpeed
```

12.2.1.196 DeviceLinkThroughputLimit

```
quickSpinIntegerNode DeviceLinkThroughputLimit
```

12.2.1.197 DeviceLinkThroughputLimitMode

```
quickSpinEnumerationNode DeviceLinkThroughputLimitMode
```

12.2.1.198 DeviceManifestEntrySelector

```
quickSpinIntegerNode DeviceManifestEntrySelector
```

12.2.1.199 DeviceManifestPrimaryURL

```
quickSpinStringNode DeviceManifestPrimaryURL
```

12.2.1.200 DeviceManifestSchemaMajorVersion

```
quickSpinIntegerNode DeviceManifestSchemaMajorVersion
```

12.2.1.201 DeviceManifestSchemaMinorVersion

```
quickSpinIntegerNode DeviceManifestSchemaMinorVersion
```

12.2.1.202 DeviceManifestSecondaryURL

```
quickSpinStringNode DeviceManifestSecondaryURL
```

12.2.1.203 DeviceManifestXMLMajorVersion

```
quickSpinIntegerNode DeviceManifestXMLMajorVersion
```

12.2.1.204 DeviceManifestXMLMinorVersion

```
quickSpinIntegerNode DeviceManifestXMLMinorVersion
```

12.2.1.205 DeviceManifestXMLSubMinorVersion

`quickSpinIntegerNode DeviceManifestXMLSubMinorVersion`

12.2.1.206 DeviceManufacturerInfo

`quickSpinStringNode DeviceManufacturerInfo`

12.2.1.207 DeviceMaxThroughput

`quickSpinIntegerNode DeviceMaxThroughput`

12.2.1.208 DeviceModelName

`quickSpinStringNode DeviceModelName`

12.2.1.209 DevicePowerSupplySelector

`quickSpinEnumerationNode DevicePowerSupplySelector`

12.2.1.210 DeviceRegistersCheck

`quickSpinCommandNode DeviceRegistersCheck`

12.2.1.211 DeviceRegistersEndianness

`quickSpinEnumerationNode DeviceRegistersEndianness`

12.2.1.212 DeviceRegistersStreamingEnd

`quickSpinCommandNode DeviceRegistersStreamingEnd`

12.2.1.213 DeviceRegistersStreamingStart

```
quickSpinCommandNode DeviceRegistersStreamingStart
```

12.2.1.214 DeviceRegistersValid

```
quickSpinBooleanNode DeviceRegistersValid
```

12.2.1.215 DeviceReset

```
quickSpinCommandNode DeviceReset
```

12.2.1.216 DeviceScanType

```
quickSpinEnumerationNode DeviceScanType
```

12.2.1.217 DeviceSerialNumber

```
quickSpinStringNode DeviceSerialNumber
```

12.2.1.218 DeviceSerialPortBaudRate

```
quickSpinEnumerationNode DeviceSerialPortBaudRate
```

12.2.1.219 DeviceSerialPortSelector

```
quickSpinEnumerationNode DeviceSerialPortSelector
```

12.2.1.220 DeviceSFNCVersionMajor

```
quickSpinIntegerNode DeviceSFNCVersionMajor
```

12.2.1.221 DeviceSFNCVersionMinor

```
quickSpinIntegerNode DeviceSFNCVersionMinor
```

12.2.1.222 DeviceSFNCVersionSubMinor

```
quickSpinIntegerNode DeviceSFNCVersionSubMinor
```

12.2.1.223 DeviceStreamChannelCount

```
quickSpinIntegerNode DeviceStreamChannelCount
```

12.2.1.224 DeviceStreamChannelEndianness

```
quickSpinEnumerationNode DeviceStreamChannelEndianness
```

12.2.1.225 DeviceStreamChannelLink

```
quickSpinIntegerNode DeviceStreamChannelLink
```

12.2.1.226 DeviceStreamChannelPacketSize

```
quickSpinIntegerNode DeviceStreamChannelPacketSize
```

12.2.1.227 DeviceStreamChannelSelector

```
quickSpinIntegerNode DeviceStreamChannelSelector
```

12.2.1.228 DeviceStreamChannelType

```
quickSpinEnumerationNode DeviceStreamChannelType
```

12.2.1.229 DeviceTapGeometry

```
quickSpinEnumerationNode DeviceTapGeometry
```

12.2.1.230 DeviceTemperature

```
quickSpinFloatNode DeviceTemperature
```

12.2.1.231 DeviceTemperatureSelector

```
quickSpinEnumerationNode DeviceTemperatureSelector
```

12.2.1.232 DeviceTLType

```
quickSpinEnumerationNode DeviceTLType
```

12.2.1.233 DeviceTLVersionMajor

```
quickSpinIntegerNode DeviceTLVersionMajor
```

12.2.1.234 DeviceTLVersionMinor

```
quickSpinIntegerNode DeviceTLVersionMinor
```

12.2.1.235 DeviceTLVersionSubMinor

```
quickSpinIntegerNode DeviceTLVersionSubMinor
```

12.2.1.236 DeviceType

```
quickSpinEnumerationNode DeviceType
```

12.2.1.237 DeviceUptime

`quickSpinIntegerNode DeviceUptime`

12.2.1.238 DeviceUserID

`quickSpinStringNode DeviceUserID`

12.2.1.239 DeviceVendorName

`quickSpinStringNode DeviceVendorName`

12.2.1.240 DeviceVersion

`quickSpinStringNode DeviceVersion`

12.2.1.241 EncoderDivider

`quickSpinIntegerNode EncoderDivider`

12.2.1.242 EncoderMode

`quickSpinEnumerationNode EncoderMode`

12.2.1.243 EncoderOutputMode

`quickSpinEnumerationNode EncoderOutputMode`

12.2.1.244 EncoderReset

`quickSpinCommandNode EncoderReset`

12.2.1.245 EncoderResetActivation

```
quickSpinEnumerationNode EncoderResetActivation
```

12.2.1.246 EncoderResetSource

```
quickSpinEnumerationNode EncoderResetSource
```

12.2.1.247 EncoderSelector

```
quickSpinEnumerationNode EncoderSelector
```

12.2.1.248 EncoderSourceA

```
quickSpinEnumerationNode EncoderSourceA
```

12.2.1.249 EncoderSourceB

```
quickSpinEnumerationNode EncoderSourceB
```

12.2.1.250 EncoderStatus

```
quickSpinEnumerationNode EncoderStatus
```

12.2.1.251 EncoderTimeout

```
quickSpinFloatNode EncoderTimeout
```

12.2.1.252 EncoderValue

```
quickSpinIntegerNode EncoderValue
```

12.2.1.253 EncoderValueAtReset

```
quickSpinIntegerNode EncoderValueAtReset
```

12.2.1.254 EnumerationCount

```
quickSpinIntegerNode EnumerationCount
```

12.2.1.255 EventAcquisitionEnd

```
quickSpinIntegerNode EventAcquisitionEnd
```

12.2.1.256 EventAcquisitionEndFrameID

```
quickSpinIntegerNode EventAcquisitionEndFrameID
```

12.2.1.257 EventAcquisitionEndTimestamp

```
quickSpinIntegerNode EventAcquisitionEndTimestamp
```

12.2.1.258 EventAcquisitionError

```
quickSpinIntegerNode EventAcquisitionError
```

12.2.1.259 EventAcquisitionErrorFrameID

```
quickSpinIntegerNode EventAcquisitionErrorFrameID
```

12.2.1.260 EventAcquisitionErrorTimestamp

```
quickSpinIntegerNode EventAcquisitionErrorTimestamp
```

12.2.1.261 EventAcquisitionStart

```
quickSpinIntegerNode EventAcquisitionStart
```

12.2.1.262 EventAcquisitionStartFrameID

```
quickSpinIntegerNode EventAcquisitionStartFrameID
```

12.2.1.263 EventAcquisitionStartTimestamp

```
quickSpinIntegerNode EventAcquisitionStartTimestamp
```

12.2.1.264 EventAcquisitionTransferEnd

```
quickSpinIntegerNode EventAcquisitionTransferEnd
```

12.2.1.265 EventAcquisitionTransferEndFrameID

```
quickSpinIntegerNode EventAcquisitionTransferEndFrameID
```

12.2.1.266 EventAcquisitionTransferEndTimestamp

```
quickSpinIntegerNode EventAcquisitionTransferEndTimestamp
```

12.2.1.267 EventAcquisitionTransferStart

```
quickSpinIntegerNode EventAcquisitionTransferStart
```

12.2.1.268 EventAcquisitionTransferStartFrameID

```
quickSpinIntegerNode EventAcquisitionTransferStartFrameID
```

12.2.1.269 EventAcquisitionTransferStartTimestamp

```
quickSpinIntegerNode EventAcquisitionTransferStartTimestamp
```

12.2.1.270 EventAcquisitionTrigger

```
quickSpinIntegerNode EventAcquisitionTrigger
```

12.2.1.271 EventAcquisitionTriggerFrameID

```
quickSpinIntegerNode EventAcquisitionTriggerFrameID
```

12.2.1.272 EventAcquisitionTriggerTimestamp

```
quickSpinIntegerNode EventAcquisitionTriggerTimestamp
```

12.2.1.273 EventActionLate

```
quickSpinIntegerNode EventActionLate
```

12.2.1.274 EventActionLateFrameID

```
quickSpinIntegerNode EventActionLateFrameID
```

12.2.1.275 EventActionLateTimestamp

```
quickSpinIntegerNode EventActionLateTimestamp
```

12.2.1.276 EventCounter0End

```
quickSpinIntegerNode EventCounter0End
```

12.2.1.277 EventCounter0EndFrameID

```
quickSpinIntegerNode EventCounter0EndFrameID
```

12.2.1.278 EventCounter0EndTimestamp

```
quickSpinIntegerNode EventCounter0EndTimestamp
```

12.2.1.279 EventCounter0Start

```
quickSpinIntegerNode EventCounter0Start
```

12.2.1.280 EventCounter0StartFrameID

```
quickSpinIntegerNode EventCounter0StartFrameID
```

12.2.1.281 EventCounter0StartTimestamp

```
quickSpinIntegerNode EventCounter0StartTimestamp
```

12.2.1.282 EventCounter1End

```
quickSpinIntegerNode EventCounter1End
```

12.2.1.283 EventCounter1EndFrameID

```
quickSpinIntegerNode EventCounter1EndFrameID
```

12.2.1.284 EventCounter1EndTimestamp

```
quickSpinIntegerNode EventCounter1EndTimestamp
```

12.2.1.285 EventCounter1Start

```
quickSpinIntegerNode EventCounter1Start
```

12.2.1.286 EventCounter1StartFrameID

```
quickSpinIntegerNode EventCounter1StartFrameID
```

12.2.1.287 EventCounter1StartTimestamp

```
quickSpinIntegerNode EventCounter1StartTimestamp
```

12.2.1.288 EventEncoder0Restarted

```
quickSpinIntegerNode EventEncoder0Restarted
```

12.2.1.289 EventEncoder0RestartedFrameID

```
quickSpinIntegerNode EventEncoder0RestartedFrameID
```

12.2.1.290 EventEncoder0RestartedTimestamp

```
quickSpinIntegerNode EventEncoder0RestartedTimestamp
```

12.2.1.291 EventEncoder0Stopped

```
quickSpinIntegerNode EventEncoder0Stopped
```

12.2.1.292 EventEncoder0StoppedFrameID

```
quickSpinIntegerNode EventEncoder0StoppedFrameID
```

12.2.1.293 EventEncoder0StoppedTimestamp

```
quickSpinIntegerNode EventEncoder0StoppedTimestamp
```

12.2.1.294 EventEncoder1Restarted

```
quickSpinIntegerNode EventEncoder1Restarted
```

12.2.1.295 EventEncoder1RestartedFrameID

```
quickSpinIntegerNode EventEncoder1RestartedFrameID
```

12.2.1.296 EventEncoder1RestartedTimestamp

```
quickSpinIntegerNode EventEncoder1RestartedTimestamp
```

12.2.1.297 EventEncoder1Stopped

```
quickSpinIntegerNode EventEncoder1Stopped
```

12.2.1.298 EventEncoder1StoppedFrameID

```
quickSpinIntegerNode EventEncoder1StoppedFrameID
```

12.2.1.299 EventEncoder1StoppedTimestamp

```
quickSpinIntegerNode EventEncoder1StoppedTimestamp
```

12.2.1.300 EventError

```
quickSpinIntegerNode EventError
```

12.2.1.301 EventErrorCode

```
quickSpinIntegerNode EventErrorCode
```

12.2.1.302 EventErrorFrameID

```
quickSpinIntegerNode EventErrorFrameID
```

12.2.1.303 EventErrorTimestamp

```
quickSpinIntegerNode EventErrorTimestamp
```

12.2.1.304 EventExposureEnd

```
quickSpinIntegerNode EventExposureEnd
```

12.2.1.305 EventExposureEndFrameID

```
quickSpinIntegerNode EventExposureEndFrameID
```

12.2.1.306 EventExposureEndTimestamp

```
quickSpinIntegerNode EventExposureEndTimestamp
```

12.2.1.307 EventExposureStart

```
quickSpinIntegerNode EventExposureStart
```

12.2.1.308 EventExposureStartFrameID

```
quickSpinIntegerNode EventExposureStartFrameID
```

12.2.1.309 EventExposureStartTimestamp

```
quickSpinIntegerNode EventExposureStartTimestamp
```

12.2.1.310 EventFrameBurstEnd

```
quickSpinIntegerNode EventFrameBurstEnd
```

12.2.1.311 EventFrameBurstEndFrameID

```
quickSpinIntegerNode EventFrameBurstEndFrameID
```

12.2.1.312 EventFrameBurstEndTimestamp

```
quickSpinIntegerNode EventFrameBurstEndTimestamp
```

12.2.1.313 EventFrameBurstStart

```
quickSpinIntegerNode EventFrameBurstStart
```

12.2.1.314 EventFrameBurstStartFrameID

```
quickSpinIntegerNode EventFrameBurstStartFrameID
```

12.2.1.315 EventFrameBurstStartTimestamp

```
quickSpinIntegerNode EventFrameBurstStartTimestamp
```

12.2.1.316 EventFrameEnd

```
quickSpinIntegerNode EventFrameEnd
```

12.2.1.317 EventFrameEndFrameID

```
quickSpinIntegerNode EventFrameEndFrameID
```

12.2.1.318 EventFrameEndTimestamp

```
quickSpinIntegerNode EventFrameEndTimestamp
```

12.2.1.319 EventFrameStart

```
quickSpinIntegerNode EventFrameStart
```

12.2.1.320 EventFrameStartFrameID

```
quickSpinIntegerNode EventFrameStartFrameID
```

12.2.1.321 EventFrameStartTimestamp

```
quickSpinIntegerNode EventFrameStartTimestamp
```

12.2.1.322 EventFrameTransferEnd

```
quickSpinIntegerNode EventFrameTransferEnd
```

12.2.1.323 EventFrameTransferEndFrameID

```
quickSpinIntegerNode EventFrameTransferEndFrameID
```

12.2.1.324 EventFrameTransferEndTimestamp

```
quickSpinIntegerNode EventFrameTransferEndTimestamp
```

12.2.1.325 EventFrameTransferStart

```
quickSpinIntegerNode EventFrameTransferStart
```

12.2.1.326 EventFrameTransferStartFrameID

```
quickSpinIntegerNode EventFrameTransferStartFrameID
```

12.2.1.327 EventFrameTransferStartTimestamp

```
quickSpinIntegerNode EventFrameTransferStartTimestamp
```

12.2.1.328 EventFrameTrigger

```
quickSpinIntegerNode EventFrameTrigger
```

12.2.1.329 EventFrameTriggerFrameID

```
quickSpinIntegerNode EventFrameTriggerFrameID
```

12.2.1.330 EventFrameTriggerTimestamp

```
quickSpinIntegerNode EventFrameTriggerTimestamp
```

12.2.1.331 EventLine0AnyEdge

```
quickSpinIntegerNode EventLine0AnyEdge
```

12.2.1.332 EventLine0AnyEdgeFrameID

```
quickSpinIntegerNode EventLine0AnyEdgeFrameID
```

12.2.1.333 EventLine0AnyEdgeTimestamp

```
quickSpinIntegerNode EventLine0AnyEdgeTimestamp
```

12.2.1.334 EventLine0FallingEdge

```
quickSpinIntegerNode EventLine0FallingEdge
```

12.2.1.335 EventLine0FallingEdgeFrameID

```
quickSpinIntegerNode EventLine0FallingEdgeFrameID
```

12.2.1.336 EventLine0FallingEdgeTimestamp

```
quickSpinIntegerNode EventLine0FallingEdgeTimestamp
```

12.2.1.337 EventLine0RisingEdge

```
quickSpinIntegerNode EventLine0RisingEdge
```

12.2.1.338 EventLine0RisingEdgeFrameID

```
quickSpinIntegerNode EventLine0RisingEdgeFrameID
```

12.2.1.339 EventLine0RisingEdgeTimestamp

```
quickSpinIntegerNode EventLine0RisingEdgeTimestamp
```

12.2.1.340 EventLine1AnyEdge

```
quickSpinIntegerNode EventLine1AnyEdge
```

12.2.1.341 EventLine1AnyEdgeFrameID

```
quickSpinIntegerNode EventLine1AnyEdgeFrameID
```

12.2.1.342 EventLine1AnyEdgeTimestamp

```
quickSpinIntegerNode EventLine1AnyEdgeTimestamp
```

12.2.1.343 EventLine1FallingEdge

```
quickSpinIntegerNode EventLine1FallingEdge
```

12.2.1.344 EventLine1FallingEdgeFrameID

```
quickSpinIntegerNode EventLine1FallingEdgeFrameID
```

12.2.1.345 EventLine1FallingEdgeTimestamp

```
quickSpinIntegerNode EventLine1FallingEdgeTimestamp
```

12.2.1.346 EventLine1RisingEdge

```
quickSpinIntegerNode EventLine1RisingEdge
```

12.2.1.347 EventLine1RisingEdgeFrameID

```
quickSpinIntegerNode EventLine1RisingEdgeFrameID
```

12.2.1.348 EventLine1RisingEdgeTimestamp

```
quickSpinIntegerNode EventLine1RisingEdgeTimestamp
```

12.2.1.349 EventLinkSpeedChange

`quickSpinIntegerNode EventLinkSpeedChange`

12.2.1.350 EventLinkSpeedChangeFrameID

`quickSpinIntegerNode EventLinkSpeedChangeFrameID`

12.2.1.351 EventLinkSpeedChangeTimestamp

`quickSpinIntegerNode EventLinkSpeedChangeTimestamp`

12.2.1.352 EventLinkTrigger0

`quickSpinIntegerNode EventLinkTrigger0`

12.2.1.353 EventLinkTrigger0FrameID

`quickSpinIntegerNode EventLinkTrigger0FrameID`

12.2.1.354 EventLinkTrigger0Timestamp

`quickSpinIntegerNode EventLinkTrigger0Timestamp`

12.2.1.355 EventLinkTrigger1

`quickSpinIntegerNode EventLinkTrigger1`

12.2.1.356 EventLinkTrigger1FrameID

`quickSpinIntegerNode EventLinkTrigger1FrameID`

12.2.1.357 EventLinkTrigger1Timestamp

```
quickSpinIntegerNode EventLinkTrigger1Timestamp
```

12.2.1.358 EventNotification

```
quickSpinEnumerationNode EventNotification
```

12.2.1.359 EventSelector

```
quickSpinEnumerationNode EventSelector
```

12.2.1.360 EventSequencerSetChange

```
quickSpinIntegerNode EventSequencerSetChange
```

12.2.1.361 EventSequencerSetChangeFrameID

```
quickSpinIntegerNode EventSequencerSetChangeFrameID
```

12.2.1.362 EventSequencerSetChangeTimestamp

```
quickSpinIntegerNode EventSequencerSetChangeTimestamp
```

12.2.1.363 EventSerialData

```
quickSpinStringNode EventSerialData
```

12.2.1.364 EventSerialDataLength

```
quickSpinIntegerNode EventSerialDataLength
```

12.2.1.365 EventSerialPortReceive

```
quickSpinIntegerNode EventSerialPortReceive
```

12.2.1.366 EventSerialPortReceiveTimestamp

```
quickSpinIntegerNode EventSerialPortReceiveTimestamp
```

12.2.1.367 EventSerialReceiveOverflow

```
quickSpinBooleanNode EventSerialReceiveOverflow
```

12.2.1.368 EventStream0TransferBlockEnd

```
quickSpinIntegerNode EventStream0TransferBlockEnd
```

12.2.1.369 EventStream0TransferBlockEndFrameID

```
quickSpinIntegerNode EventStream0TransferBlockEndFrameID
```

12.2.1.370 EventStream0TransferBlockEndTimestamp

```
quickSpinIntegerNode EventStream0TransferBlockEndTimestamp
```

12.2.1.371 EventStream0TransferBlockStart

```
quickSpinIntegerNode EventStream0TransferBlockStart
```

12.2.1.372 EventStream0TransferBlockStartFrameID

```
quickSpinIntegerNode EventStream0TransferBlockStartFrameID
```

12.2.1.373 EventStream0TransferBlockStartTimestamp

```
quickSpinIntegerNode EventStream0TransferBlockStartTimestamp
```

12.2.1.374 EventStream0TransferBlockTrigger

```
quickSpinIntegerNode EventStream0TransferBlockTrigger
```

12.2.1.375 EventStream0TransferBlockTriggerFrameID

```
quickSpinIntegerNode EventStream0TransferBlockTriggerFrameID
```

12.2.1.376 EventStream0TransferBlockTriggerTimestamp

```
quickSpinIntegerNode EventStream0TransferBlockTriggerTimestamp
```

12.2.1.377 EventStream0TransferBurstEnd

```
quickSpinIntegerNode EventStream0TransferBurstEnd
```

12.2.1.378 EventStream0TransferBurstEndFrameID

```
quickSpinIntegerNode EventStream0TransferBurstEndFrameID
```

12.2.1.379 EventStream0TransferBurstEndTimestamp

```
quickSpinIntegerNode EventStream0TransferBurstEndTimestamp
```

12.2.1.380 EventStream0TransferBurstStart

```
quickSpinIntegerNode EventStream0TransferBurstStart
```

12.2.1.381 EventStream0TransferBurstStartFrameID

```
quickSpinIntegerNode EventStream0TransferBurstStartFrameID
```

12.2.1.382 EventStream0TransferBurstStartTimestamp

```
quickSpinIntegerNode EventStream0TransferBurstStartTimestamp
```

12.2.1.383 EventStream0TransferEnd

```
quickSpinIntegerNode EventStream0TransferEnd
```

12.2.1.384 EventStream0TransferEndFrameID

```
quickSpinIntegerNode EventStream0TransferEndFrameID
```

12.2.1.385 EventStream0TransferEndTimestamp

```
quickSpinIntegerNode EventStream0TransferEndTimestamp
```

12.2.1.386 EventStream0TransferOverflow

```
quickSpinIntegerNode EventStream0TransferOverflow
```

12.2.1.387 EventStream0TransferOverflowFrameID

```
quickSpinIntegerNode EventStream0TransferOverflowFrameID
```

12.2.1.388 EventStream0TransferOverflowTimestamp

```
quickSpinIntegerNode EventStream0TransferOverflowTimestamp
```

12.2.1.389 EventStream0TransferPause

```
quickSpinIntegerNode EventStream0TransferPause
```

12.2.1.390 EventStream0TransferPauseFrameID

```
quickSpinIntegerNode EventStream0TransferPauseFrameID
```

12.2.1.391 EventStream0TransferPauseTimestamp

```
quickSpinIntegerNode EventStream0TransferPauseTimestamp
```

12.2.1.392 EventStream0TransferResume

```
quickSpinIntegerNode EventStream0TransferResume
```

12.2.1.393 EventStream0TransferResumeFrameID

```
quickSpinIntegerNode EventStream0TransferResumeFrameID
```

12.2.1.394 EventStream0TransferResumeTimestamp

```
quickSpinIntegerNode EventStream0TransferResumeTimestamp
```

12.2.1.395 EventStream0TransferStart

```
quickSpinIntegerNode EventStream0TransferStart
```

12.2.1.396 EventStream0TransferStartFrameID

```
quickSpinIntegerNode EventStream0TransferStartFrameID
```

12.2.1.397 EventStream0TransferStartTimestamp

```
quickSpinIntegerNode EventStream0TransferStartTimestamp
```

12.2.1.398 EventTest

```
quickSpinIntegerNode EventTest
```

12.2.1.399 EventTestTimestamp

```
quickSpinIntegerNode EventTestTimestamp
```

12.2.1.400 EventTimer0End

```
quickSpinIntegerNode EventTimer0End
```

12.2.1.401 EventTimer0EndFrameID

```
quickSpinIntegerNode EventTimer0EndFrameID
```

12.2.1.402 EventTimer0EndTimestamp

```
quickSpinIntegerNode EventTimer0EndTimestamp
```

12.2.1.403 EventTimer0Start

```
quickSpinIntegerNode EventTimer0Start
```

12.2.1.404 EventTimer0StartFrameID

```
quickSpinIntegerNode EventTimer0StartFrameID
```

12.2.1.405 EventTimer0StartTimestamp

```
quickSpinIntegerNode EventTimer0StartTimestamp
```

12.2.1.406 EventTimer1End

```
quickSpinIntegerNode EventTimer1End
```

12.2.1.407 EventTimer1EndFrameID

```
quickSpinIntegerNode EventTimer1EndFrameID
```

12.2.1.408 EventTimer1EndTimestamp

```
quickSpinIntegerNode EventTimer1EndTimestamp
```

12.2.1.409 EventTimer1Start

```
quickSpinIntegerNode EventTimer1Start
```

12.2.1.410 EventTimer1StartFrameID

```
quickSpinIntegerNode EventTimer1StartFrameID
```

12.2.1.411 EventTimer1StartTimestamp

```
quickSpinIntegerNode EventTimer1StartTimestamp
```

12.2.1.412 ExposureActiveMode

```
quickSpinEnumerationNode ExposureActiveMode
```

12.2.1.413 ExposureAuto

```
quickSpinEnumerationNode ExposureAuto
```

12.2.1.414 ExposureMode

```
quickSpinEnumerationNode ExposureMode
```

12.2.1.415 ExposureTime

```
quickSpinFloatNode ExposureTime
```

12.2.1.416 ExposureTimeMode

```
quickSpinEnumerationNode ExposureTimeMode
```

12.2.1.417 ExposureTimeSelector

```
quickSpinEnumerationNode ExposureTimeSelector
```

12.2.1.418 FactoryReset

```
quickSpinCommandNode FactoryReset
```

12.2.1.419 FileAccessBuffer

```
quickSpinRegisterNode FileAccessBuffer
```

12.2.1.420 FileAccessLength

```
quickSpinIntegerNode FileAccessLength
```

12.2.1.421 FileAccessOffset

```
quickSpinIntegerNode FileAccessOffset
```

12.2.1.422 FileOpenMode

```
quickSpinEnumerationNode FileOpenMode
```

12.2.1.423 FileOperationExecute

```
quickSpinCommandNode FileOperationExecute
```

12.2.1.424 FileOperationResult

```
quickSpinIntegerNode FileOperationResult
```

12.2.1.425 FileOperationSelector

```
quickSpinEnumerationNode FileOperationSelector
```

12.2.1.426 FileOperationStatus

```
quickSpinEnumerationNode FileOperationStatus
```

12.2.1.427 FileSelector

```
quickSpinEnumerationNode FileSelector
```

12.2.1.428 FileSize

```
quickSpinIntegerNode FileSize
```

12.2.1.429 Gain

```
quickSpinFloatNode Gain
```

12.2.1.430 GainAuto

```
quickSpinEnumerationNode GainAuto
```

12.2.1.431 GainAutoBalance

```
quickSpinEnumerationNode GainAutoBalance
```

12.2.1.432 GainSelector

```
quickSpinEnumerationNode GainSelector
```

12.2.1.433 Gamma

```
quickSpinFloatNode Gamma
```

12.2.1.434 GammaEnable

```
quickSpinBooleanNode GammaEnable
```

12.2.1.435 GevActiveLinkCount

```
quickSpinIntegerNode GevActiveLinkCount
```

12.2.1.436 GevCCP

```
quickSpinEnumerationNode GevCCP
```

12.2.1.437 GevCurrentDefaultGateway

```
quickSpinIntegerNode GevCurrentDefaultGateway
```

12.2.1.438 GevCurrentIPAddress

```
quickSpinIntegerNode GevCurrentIPAddress
```

12.2.1.439 GevCurrentIPConfigurationDHCP

```
quickSpinBooleanNode GevCurrentIPConfigurationDHCP
```

12.2.1.440 GevCurrentIPConfigurationLLA

```
quickSpinBooleanNode GevCurrentIPConfigurationLLA
```

12.2.1.441 GevCurrentIPConfigurationPersistentIP

```
quickSpinBooleanNode GevCurrentIPConfigurationPersistentIP
```

12.2.1.442 GevCurrentPhysicalLinkConfiguration

```
quickSpinEnumerationNode GevCurrentPhysicalLinkConfiguration
```

12.2.1.443 GevCurrentSubnetMask

```
quickSpinIntegerNode GevCurrentSubnetMask
```

12.2.1.444 GevDiscoveryAckDelay

```
quickSpinIntegerNode GevDiscoveryAckDelay
```

12.2.1.445 GevFirstURL

`quickSpinStringNode GevFirstURL`

12.2.1.446 GevGVCPExtendedStatusCodes

`quickSpinBooleanNode GevGVCPExtendedStatusCodes`

12.2.1.447 GevGVCPExtendedStatusCodesSelector

`quickSpinEnumerationNode GevGVCPExtendedStatusCodesSelector`

12.2.1.448 GevGVCPHeartbeatDisable

`quickSpinBooleanNode GevGVCPHeartbeatDisable`

12.2.1.449 GevGVCPPendingAck

`quickSpinBooleanNode GevGVCPPendingAck`

12.2.1.450 GevGVCPPendingTimeout

`quickSpinIntegerNode GevGVCPPendingTimeout`

12.2.1.451 GevGVSPExtendedIDMode

`quickSpinEnumerationNode GevGVSPExtendedIDMode`

12.2.1.452 GevHeartbeatTimeout

`quickSpinIntegerNode GevHeartbeatTimeout`

12.2.1.453 GevIEEE1588

```
quickSpinBooleanNode GevIEEE1588
```

12.2.1.454 GevIEEE1588ClockAccuracy

```
quickSpinEnumerationNode GevIEEE1588ClockAccuracy
```

12.2.1.455 GevIEEE1588Mode

```
quickSpinEnumerationNode GevIEEE1588Mode
```

12.2.1.456 GevIEEE1588Status

```
quickSpinEnumerationNode GevIEEE1588Status
```

12.2.1.457 GevInterfaceSelector

```
quickSpinIntegerNode GevInterfaceSelector
```

12.2.1.458 GevIPConfigurationStatus

```
quickSpinEnumerationNode GevIPConfigurationStatus
```

12.2.1.459 GevMACAddress

```
quickSpinIntegerNode GevMACAddress
```

12.2.1.460 GevMCDA

```
quickSpinIntegerNode GevMCDA
```

12.2.1.461 GevMCPHostPort

```
quickSpinIntegerNode GevMCPHostPort
```

12.2.1.462 GevMCRC

```
quickSpinIntegerNode GevMCRC
```

12.2.1.463 GevMCSP

```
quickSpinIntegerNode GevMCSP
```

12.2.1.464 GevMCTT

```
quickSpinIntegerNode GevMCTT
```

12.2.1.465 GevNumberOfInterfaces

```
quickSpinIntegerNode GevNumberOfInterfaces
```

12.2.1.466 GevPAUSEFrameReception

```
quickSpinBooleanNode GevPAUSEFrameReception
```

12.2.1.467 GevPAUSEFrameTransmission

```
quickSpinBooleanNode GevPAUSEFrameTransmission
```

12.2.1.468 GevPersistentDefaultGateway

```
quickSpinIntegerNode GevPersistentDefaultGateway
```

12.2.1.469 GevPersistentIPAddress

```
quickSpinIntegerNode GevPersistentIPAddress
```

12.2.1.470 GevPersistentSubnetMask

```
quickSpinIntegerNode GevPersistentSubnetMask
```

12.2.1.471 GevPhysicalLinkConfiguration

```
quickSpinEnumerationNode GevPhysicalLinkConfiguration
```

12.2.1.472 GevPrimaryApplicationIPAddress

```
quickSpinIntegerNode GevPrimaryApplicationIPAddress
```

12.2.1.473 GevPrimaryApplicationSocket

```
quickSpinIntegerNode GevPrimaryApplicationSocket
```

12.2.1.474 GevPrimaryApplicationSwitchoverKey

```
quickSpinIntegerNode GevPrimaryApplicationSwitchoverKey
```

12.2.1.475 GevSCCFGAllInTransmission

```
quickSpinBooleanNode GevSCCFGAllInTransmission
```

12.2.1.476 GevSCCFGExtendedChunkData

```
quickSpinBooleanNode GevSCCFGExtendedChunkData
```

12.2.1.477 GevSCCFGPacketResendDestination

```
quickSpinBooleanNode GevSCCFGPacketResendDestination
```

12.2.1.478 GevSCCFGUnconditionalStreaming

```
quickSpinBooleanNode GevSCCFGUnconditionalStreaming
```

12.2.1.479 GevSCDA

```
quickSpinIntegerNode GevSCDA
```

12.2.1.480 GevSCPD

```
quickSpinIntegerNode GevSCPD
```

12.2.1.481 GevSCPDirection

```
quickSpinIntegerNode GevSCPDirection
```

12.2.1.482 GevSCPHostPort

```
quickSpinIntegerNode GevSCPHostPort
```

12.2.1.483 GevSCPInterfaceIndex

```
quickSpinIntegerNode GevSCPInterfaceIndex
```

12.2.1.484 GevSCPSBigEndian

```
quickSpinBooleanNode GevSCPSBigEndian
```

12.2.1.485 GevSCPSDoNotFragment

```
quickSpinBooleanNode GevSCPSDoNotFragment
```

12.2.1.486 GevSCPSFireTestPacket

```
quickSpinBooleanNode GevSCPSFireTestPacket
```

12.2.1.487 GevSCPSPacketSize

```
quickSpinIntegerNode GevSCPSPacketSize
```

12.2.1.488 GevSCSP

```
quickSpinIntegerNode GevSCSP
```

12.2.1.489 GevSCZoneConfigurationLock

```
quickSpinBooleanNode GevSCZoneConfigurationLock
```

12.2.1.490 GevSCZoneCount

```
quickSpinIntegerNode GevSCZoneCount
```

12.2.1.491 GevSCZoneDirectionAll

```
quickSpinIntegerNode GevSCZoneDirectionAll
```

12.2.1.492 GevSecondURL

```
quickSpinStringNode GevSecondURL
```

12.2.1.493 GevStreamChannelSelector

```
quickSpinIntegerNode GevStreamChannelSelector
```

12.2.1.494 GevSupportedOption

```
quickSpinBooleanNode GevSupportedOption
```

12.2.1.495 GevSupportedOptionSelector

```
quickSpinEnumerationNode GevSupportedOptionSelector
```

12.2.1.496 GevTimestampTickFrequency

```
quickSpinIntegerNode GevTimestampTickFrequency
```

12.2.1.497 GuiXmlManifestAddress

```
quickSpinIntegerNode GuiXmlManifestAddress
```

12.2.1.498 Height

```
quickSpinIntegerNode Height
```

12.2.1.499 HeightMax

```
quickSpinIntegerNode HeightMax
```

12.2.1.500 ImageComponentEnable

```
quickSpinBooleanNode ImageComponentEnable
```

12.2.1.501 ImageComponentSelector

```
quickSpinEnumerationNode ImageComponentSelector
```

12.2.1.502 ImageCompressionBitrate

```
quickSpinFloatNode ImageCompressionBitrate
```

12.2.1.503 ImageCompressionJPEGFormatOption

```
quickSpinEnumerationNode ImageCompressionJPEGFormatOption
```

12.2.1.504 ImageCompressionMode

```
quickSpinEnumerationNode ImageCompressionMode
```

12.2.1.505 ImageCompressionQuality

```
quickSpinIntegerNode ImageCompressionQuality
```

12.2.1.506 ImageCompressionRateOption

```
quickSpinEnumerationNode ImageCompressionRateOption
```

12.2.1.507 IspEnable

```
quickSpinBooleanNode IspEnable
```

12.2.1.508 LineFilterWidth

```
quickSpinFloatNode LineFilterWidth
```

12.2.1.509 LineFormat

```
quickSpinEnumerationNode LineFormat
```

12.2.1.510 LineInputFilterSelector

```
quickSpinEnumerationNode LineInputFilterSelector
```

12.2.1.511 LineInverter

```
quickSpinBooleanNode LineInverter
```

12.2.1.512 LineMode

```
quickSpinEnumerationNode LineMode
```

12.2.1.513 LinePitch

```
quickSpinIntegerNode LinePitch
```

12.2.1.514 LineSelector

```
quickSpinEnumerationNode LineSelector
```

12.2.1.515 LineSource

```
quickSpinEnumerationNode LineSource
```

12.2.1.516 LineStatus

```
quickSpinBooleanNode LineStatus
```

12.2.1.517 LineStatusAll

```
quickSpinIntegerNode LineStatusAll
```

12.2.1.518 LinkErrorCount

```
quickSpinIntegerNode LinkErrorCount
```

12.2.1.519 LinkUptime

```
quickSpinIntegerNode LinkUptime
```

12.2.1.520 LogicBlockLUTInputActivation

```
quickSpinEnumerationNode LogicBlockLUTInputActivation
```

12.2.1.521 LogicBlockLUTInputSelector

```
quickSpinEnumerationNode LogicBlockLUTInputSelector
```

12.2.1.522 LogicBlockLUTInputSource

```
quickSpinEnumerationNode LogicBlockLUTInputSource
```

12.2.1.523 LogicBlockLUTOutputValue

```
quickSpinBooleanNode LogicBlockLUTOutputValue
```

12.2.1.524 LogicBlockLUTOutputValueAll

```
quickSpinIntegerNode LogicBlockLUTOutputValueAll
```

12.2.1.525 LogicBlockLUTRowIndex

```
quickSpinIntegerNode LogicBlockLUTRowIndex
```

12.2.1.526 LogicBlockLUTSelector

```
quickSpinEnumerationNode LogicBlockLUTSelector
```

12.2.1.527 LogicBlockSelector

```
quickSpinEnumerationNode LogicBlockSelector
```

12.2.1.528 LUTEnable

```
quickSpinBooleanNode LUTEnable
```

12.2.1.529 LUTIndex

```
quickSpinIntegerNode LUTIndex
```

12.2.1.530 LUTSelector

```
quickSpinEnumerationNode LUTSelector
```

12.2.1.531 LUTValue

```
quickSpinIntegerNode LUTValue
```

12.2.1.532 LUTValueAll

```
quickSpinRegisterNode LUTValueAll
```

12.2.1.533 MaxDeviceResetTime

```
quickSpinIntegerNode MaxDeviceResetTime
```

12.2.1.534 OffsetX

```
quickSpinIntegerNode OffsetX
```

12.2.1.535 OffsetY

```
quickSpinIntegerNode OffsetY
```

12.2.1.536 PacketResendRequestCount

```
quickSpinIntegerNode PacketResendRequestCount
```

12.2.1.537 PayloadSize

```
quickSpinIntegerNode PayloadSize
```

12.2.1.538 PixelColorFilter

```
quickSpinEnumerationNode PixelColorFilter
```

12.2.1.539 PixelDynamicRangeMax

```
quickSpinIntegerNode PixelDynamicRangeMax
```

12.2.1.540 PixelDynamicRangeMin

```
quickSpinIntegerNode PixelDynamicRangeMin
```

12.2.1.541 PixelFormat

`quickSpinEnumerationNode PixelFormat`

12.2.1.542 PixelFormatInfoID

`quickSpinIntegerNode PixelFormatInfoID`

12.2.1.543 PixelFormatInfoSelector

`quickSpinEnumerationNode PixelFormatInfoSelector`

12.2.1.544 PixelSize

`quickSpinEnumerationNode PixelSize`

12.2.1.545 PowerSupplyCurrent

`quickSpinFloatNode PowerSupplyCurrent`

12.2.1.546 PowerSupplyVoltage

`quickSpinFloatNode PowerSupplyVoltage`

12.2.1.547 RegionDestination

`quickSpinEnumerationNode RegionDestination`

12.2.1.548 RegionMode

`quickSpinEnumerationNode RegionMode`

12.2.1.549 RegionSelector

```
quickSpinEnumerationNode RegionSelector
```

12.2.1.550 ReverseX

```
quickSpinBooleanNode ReverseX
```

12.2.1.551 ReverseY

```
quickSpinBooleanNode ReverseY
```

12.2.1.552 RgbTransformLightSource

```
quickSpinEnumerationNode RgbTransformLightSource
```

12.2.1.553 Saturation

```
quickSpinFloatNode Saturation
```

12.2.1.554 SaturationEnable

```
quickSpinBooleanNode SaturationEnable
```

12.2.1.555 Scan3dAxisMax

```
quickSpinFloatNode Scan3dAxisMax
```

12.2.1.556 Scan3dAxisMin

```
quickSpinFloatNode Scan3dAxisMin
```

12.2.1.557 Scan3dCoordinateOffset

```
quickSpinFloatNode Scan3dCoordinateOffset
```

12.2.1.558 Scan3dCoordinateReferenceSelector

```
quickSpinEnumerationNode Scan3dCoordinateReferenceSelector
```

12.2.1.559 Scan3dCoordinateReferenceValue

```
quickSpinFloatNode Scan3dCoordinateReferenceValue
```

12.2.1.560 Scan3dCoordinateScale

```
quickSpinFloatNode Scan3dCoordinateScale
```

12.2.1.561 Scan3dCoordinateSelector

```
quickSpinEnumerationNode Scan3dCoordinateSelector
```

12.2.1.562 Scan3dCoordinateSystem

```
quickSpinEnumerationNode Scan3dCoordinateSystem
```

12.2.1.563 Scan3dCoordinateSystemReference

```
quickSpinEnumerationNode Scan3dCoordinateSystemReference
```

12.2.1.564 Scan3dCoordinateTransformSelector

```
quickSpinEnumerationNode Scan3dCoordinateTransformSelector
```

12.2.1.565 Scan3dDistanceUnit

```
quickSpinEnumerationNode Scan3dDistanceUnit
```

12.2.1.566 Scan3dInvalidDataFlag

```
quickSpinBooleanNode Scan3dInvalidDataFlag
```

12.2.1.567 Scan3dInvalidDataValue

```
quickSpinFloatNode Scan3dInvalidDataValue
```

12.2.1.568 Scan3dOutputMode

```
quickSpinEnumerationNode Scan3dOutputMode
```

12.2.1.569 Scan3dTransformValue

```
quickSpinFloatNode Scan3dTransformValue
```

12.2.1.570 SensorDescription

```
quickSpinStringNode SensorDescription
```

12.2.1.571 SensorDigitizationTaps

```
quickSpinEnumerationNode SensorDigitizationTaps
```

12.2.1.572 SensorHeight

```
quickSpinIntegerNode SensorHeight
```

12.2.1.573 SensorShutterMode

```
quickSpinEnumerationNode SensorShutterMode
```

12.2.1.574 SensorTaps

```
quickSpinEnumerationNode SensorTaps
```

12.2.1.575 SensorWidth

```
quickSpinIntegerNode SensorWidth
```

12.2.1.576 SequencerConfigurationMode

```
quickSpinEnumerationNode SequencerConfigurationMode
```

12.2.1.577 SequencerConfigurationValid

```
quickSpinEnumerationNode SequencerConfigurationValid
```

12.2.1.578 SequencerFeatureEnable

```
quickSpinBooleanNode SequencerFeatureEnable
```

12.2.1.579 SequencerMode

```
quickSpinEnumerationNode SequencerMode
```

12.2.1.580 SequencerPathSelector

```
quickSpinIntegerNode SequencerPathSelector
```

12.2.1.581 SequencerSetActive

```
quickSpinIntegerNode SequencerSetActive
```

12.2.1.582 SequencerSetLoad

```
quickSpinCommandNode SequencerSetLoad
```

12.2.1.583 SequencerSetNext

```
quickSpinIntegerNode SequencerSetNext
```

12.2.1.584 SequencerSetSave

```
quickSpinCommandNode SequencerSetSave
```

12.2.1.585 SequencerSetSelector

```
quickSpinIntegerNode SequencerSetSelector
```

12.2.1.586 SequencerSetStart

```
quickSpinIntegerNode SequencerSetStart
```

12.2.1.587 SequencerSetValid

```
quickSpinEnumerationNode SequencerSetValid
```

12.2.1.588 SequencerTriggerActivation

```
quickSpinEnumerationNode SequencerTriggerActivation
```

12.2.1.589 SequencerTriggerSource

```
quickSpinEnumerationNode SequencerTriggerSource
```

12.2.1.590 SerialPortBaudRate

```
quickSpinEnumerationNode SerialPortBaudRate
```

12.2.1.591 SerialPortDataBits

```
quickSpinIntegerNode SerialPortDataBits
```

12.2.1.592 SerialPortParity

```
quickSpinEnumerationNode SerialPortParity
```

12.2.1.593 SerialPortSelector

```
quickSpinEnumerationNode SerialPortSelector
```

12.2.1.594 SerialPortSource

```
quickSpinEnumerationNode SerialPortSource
```

12.2.1.595 SerialPortStopBits

```
quickSpinEnumerationNode SerialPortStopBits
```

12.2.1.596 SerialReceiveFramingErrorCount

```
quickSpinIntegerNode SerialReceiveFramingErrorCount
```

12.2.1.597 SerialReceiveParityErrorCount

```
quickSpinIntegerNode SerialReceiveParityErrorCount
```

12.2.1.598 SerialReceiveQueueClear

```
quickSpinCommandNode SerialReceiveQueueClear
```

12.2.1.599 SerialReceiveQueueCurrentCharacterCount

```
quickSpinIntegerNode SerialReceiveQueueCurrentCharacterCount
```

12.2.1.600 SerialReceiveQueueMaxCharacterCount

```
quickSpinIntegerNode SerialReceiveQueueMaxCharacterCount
```

12.2.1.601 SerialTransmitQueueCurrentCharacterCount

```
quickSpinIntegerNode SerialTransmitQueueCurrentCharacterCount
```

12.2.1.602 SerialTransmitQueueMaxCharacterCount

```
quickSpinIntegerNode SerialTransmitQueueMaxCharacterCount
```

12.2.1.603 Sharpening

```
quickSpinFloatNode Sharpening
```

12.2.1.604 SharpeningAuto

```
quickSpinBooleanNode SharpeningAuto
```

12.2.1.605 SharpeningEnable

`quickSpinBooleanNode SharpeningEnable`

12.2.1.606 SharpeningThreshold

`quickSpinFloatNode SharpeningThreshold`

12.2.1.607 SoftwareSignalPulse

`quickSpinCommandNode SoftwareSignalPulse`

12.2.1.608 SoftwareSignalSelector

`quickSpinEnumerationNode SoftwareSignalSelector`

12.2.1.609 SourceCount

`quickSpinIntegerNode SourceCount`

12.2.1.610 SourceSelector

`quickSpinEnumerationNode SourceSelector`

12.2.1.611 Test0001

`quickSpinIntegerNode Test0001`

12.2.1.612 TestEventGenerate

`quickSpinCommandNode TestEventGenerate`

12.2.1.613 TestPattern

```
quickSpinEnumerationNode TestPattern
```

12.2.1.614 TestPatternGeneratorSelector

```
quickSpinEnumerationNode TestPatternGeneratorSelector
```

12.2.1.615 TestPendingAck

```
quickSpinIntegerNode TestPendingAck
```

12.2.1.616 TimerDelay

```
quickSpinFloatNode TimerDelay
```

12.2.1.617 TimerDuration

```
quickSpinFloatNode TimerDuration
```

12.2.1.618 TimerReset

```
quickSpinCommandNode TimerReset
```

12.2.1.619 TimerSelector

```
quickSpinEnumerationNode TimerSelector
```

12.2.1.620 TimerStatus

```
quickSpinEnumerationNode TimerStatus
```

12.2.1.621 TimerTriggerActivation

```
quickSpinEnumerationNode TimerTriggerActivation
```

12.2.1.622 TimerTriggerSource

```
quickSpinEnumerationNode TimerTriggerSource
```

12.2.1.623 TimerValue

```
quickSpinFloatNode TimerValue
```

12.2.1.624 Timestamp

```
quickSpinIntegerNode Timestamp
```

12.2.1.625 TimestampLatch

```
quickSpinCommandNode TimestampLatch
```

12.2.1.626 TimestampLatchValue

```
quickSpinIntegerNode TimestampLatchValue
```

12.2.1.627 TimestampReset

```
quickSpinCommandNode TimestampReset
```

12.2.1.628 TLParamsLocked

```
quickSpinIntegerNode TLParamsLocked
```

12.2.1.629 TransferAbort

```
quickSpinCommandNode TransferAbort
```

12.2.1.630 TransferBlockCount

```
quickSpinIntegerNode TransferBlockCount
```

12.2.1.631 TransferBurstCount

```
quickSpinIntegerNode TransferBurstCount
```

12.2.1.632 TransferComponentSelector

```
quickSpinEnumerationNode TransferComponentSelector
```

12.2.1.633 TransferControlMode

```
quickSpinEnumerationNode TransferControlMode
```

12.2.1.634 TransferOperationMode

```
quickSpinEnumerationNode TransferOperationMode
```

12.2.1.635 TransferPause

```
quickSpinCommandNode TransferPause
```

12.2.1.636 TransferQueueCurrentBlockCount

```
quickSpinIntegerNode TransferQueueCurrentBlockCount
```

12.2.1.637 TransferQueueMaxBlockCount

```
quickSpinIntegerNode TransferQueueMaxBlockCount
```

12.2.1.638 TransferQueueMode

```
quickSpinEnumerationNode TransferQueueMode
```

12.2.1.639 TransferQueueOverflowCount

```
quickSpinIntegerNode TransferQueueOverflowCount
```

12.2.1.640 TransferResume

```
quickSpinCommandNode TransferResume
```

12.2.1.641 TransferSelector

```
quickSpinEnumerationNode TransferSelector
```

12.2.1.642 TransferStart

```
quickSpinCommandNode TransferStart
```

12.2.1.643 TransferStatus

```
quickSpinBooleanNode TransferStatus
```

12.2.1.644 TransferStatusSelector

```
quickSpinEnumerationNode TransferStatusSelector
```

12.2.1.645 TransferStop

```
quickSpinCommandNode TransferStop
```

12.2.1.646 TransferStreamChannel

```
quickSpinIntegerNode TransferStreamChannel
```

12.2.1.647 TransferTriggerActivation

```
quickSpinEnumerationNode TransferTriggerActivation
```

12.2.1.648 TransferTriggerMode

```
quickSpinEnumerationNode TransferTriggerMode
```

12.2.1.649 TransferTriggerSelector

```
quickSpinEnumerationNode TransferTriggerSelector
```

12.2.1.650 TransferTriggerSource

```
quickSpinEnumerationNode TransferTriggerSource
```

12.2.1.651 TriggerActivation

```
quickSpinEnumerationNode TriggerActivation
```

12.2.1.652 TriggerDelay

```
quickSpinFloatNode TriggerDelay
```

12.2.1.653 TriggerDivider

```
quickSpinIntegerNode TriggerDivider
```

12.2.1.654 TriggerEventTest

```
quickSpinCommandNode TriggerEventTest
```

12.2.1.655 TriggerMode

```
quickSpinEnumerationNode TriggerMode
```

12.2.1.656 TriggerMultiplier

```
quickSpinIntegerNode TriggerMultiplier
```

12.2.1.657 TriggerOverlap

```
quickSpinEnumerationNode TriggerOverlap
```

12.2.1.658 TriggerSelector

```
quickSpinEnumerationNode TriggerSelector
```

12.2.1.659 TriggerSoftware

```
quickSpinCommandNode TriggerSoftware
```

12.2.1.660 TriggerSource

```
quickSpinEnumerationNode TriggerSource
```

12.2.1.661 UserOutputSelector

```
quickSpinEnumerationNode UserOutputSelector
```

12.2.1.662 UserOutputValue

```
quickSpinBooleanNode UserOutputValue
```

12.2.1.663 UserOutputValueAll

```
quickSpinIntegerNode UserOutputValueAll
```

12.2.1.664 UserOutputValueAllMask

```
quickSpinIntegerNode UserOutputValueAllMask
```

12.2.1.665 UserSetDefault

```
quickSpinEnumerationNode UserSetDefault
```

12.2.1.666 UserSetFeatureEnable

```
quickSpinBooleanNode UserSetFeatureEnable
```

12.2.1.667 UserSetLoad

```
quickSpinCommandNode UserSetLoad
```

12.2.1.668 UserSetSave

```
quickSpinCommandNode UserSetSave
```

12.2.1.669 UserSetSelector

```
quickSpinEnumerationNode UserSetSelector
```

12.2.1.670 V3_3Enable

```
quickSpinBooleanNode V3_3Enable
```

12.2.1.671 WhiteClip

```
quickSpinFloatNode WhiteClip
```

12.2.1.672 WhiteClipSelector

```
quickSpinEnumerationNode WhiteClipSelector
```

12.2.1.673 Width

```
quickSpinIntegerNode Width
```

12.2.1.674 WidthMax

```
quickSpinIntegerNode WidthMax
```

The documentation for this struct was generated from the following file:

- include/spinc/QuickSpinDefsC.h

12.3 quickSpinTLDevice Struct Reference

Data Fields

- quickSpinStringNode DeviceID
- quickSpinStringNode DeviceSerialNumber
- quickSpinStringNode DeviceUserID
- quickSpinStringNode DeviceVendorName
- quickSpinStringNode DeviceModelName
- quickSpinStringNode DeviceVersion
- quickSpinIntegerNode DeviceBootloaderVersion
- quickSpinEnumerationNode DeviceType
- quickSpinStringNode DeviceDisplayName
- quickSpinEnumerationNode DeviceAccessStatus
- quickSpinIntegerNode DeviceLinkSpeed
- quickSpinStringNode DeviceDriverVersion
- quickSpinBooleanNode DeviceIsUpdater
- quickSpinEnumerationNode GenICamXMLLocation
- quickSpinStringNode GenICamXMLPath
- quickSpinEnumerationNode GUIXMLLocation
- quickSpinStringNode GUIXMLPath
- quickSpinEnumerationNode GevCCP
- quickSpinIntegerNode GevDeviceMACAddress
- quickSpinIntegerNode GevDeviceIPAddress
- quickSpinIntegerNode GevDeviceSubnetMask
- quickSpinIntegerNode GevDeviceGateway
- quickSpinIntegerNode GevVersionMajor
- quickSpinIntegerNode GevVersionMinor
- quickSpinBooleanNode GevDeviceModelIsBigEndian
- quickSpinIntegerNode GevDeviceReadAndWriteTimeout
- quickSpinIntegerNode GevDeviceMaximumRetryCount
- quickSpinIntegerNode GevDevicePort
- quickSpinCommandNode GevDeviceDiscoverMaximumPacketSize
- quickSpinIntegerNode GevDeviceMaximumPacketSize
- quickSpinBooleanNode GevDeviceIsWrongSubnet
- quickSpinCommandNode GevDeviceAutoForceIP
- quickSpinCommandNode GevDeviceForceIP
- quickSpinIntegerNode GevDeviceForceIPAddress
- quickSpinIntegerNode GevDeviceForceSubnetMask
- quickSpinIntegerNode GevDeviceForceGateway
- quickSpinBooleanNode DeviceMulticastMonitorMode
- quickSpinEnumerationNode DeviceEndianessMechanism
- quickSpinCommandNode DeviceReset
- quickSpinStringNode DeviceInstanceId
- quickSpinStringNode DeviceLocation
- quickSpinEnumerationNode DeviceCurrentSpeed
- quickSpinBooleanNode DeviceU3VProtocol
- quickSpinStringNode DevicePortId

12.3.1 Field Documentation

12.3.1.1 DeviceAccessStatus

```
quickSpinEnumerationNode DeviceAccessStatus
```

12.3.1.2 DeviceBootloaderVersion

```
quickSpinIntegerNode DeviceBootloaderVersion
```

12.3.1.3 DeviceCurrentSpeed

```
quickSpinEnumerationNode DeviceCurrentSpeed
```

12.3.1.4 DeviceDisplayName

```
quickSpinStringNode DeviceDisplayName
```

12.3.1.5 DeviceDriverVersion

```
quickSpinStringNode DeviceDriverVersion
```

12.3.1.6 DeviceEndianessMechanism

```
quickSpinEnumerationNode DeviceEndianessMechanism
```

12.3.1.7 DeviceID

```
quickSpinStringNode DeviceID
```

12.3.1.8 DeviceInstanceId

```
quickSpinStringNode DeviceInstanceId
```

12.3.1.9 DeviceIsUpdater

```
quickSpinBooleanNode DeviceIsUpdater
```

12.3.1.10 DeviceLinkSpeed

```
quickSpinIntegerNode DeviceLinkSpeed
```

12.3.1.11 DeviceLocation

```
quickSpinStringNode DeviceLocation
```

12.3.1.12 DeviceModelName

```
quickSpinStringNode DeviceModelName
```

12.3.1.13 DeviceMulticastMonitorMode

```
quickSpinBooleanNode DeviceMulticastMonitorMode
```

12.3.1.14 DevicePortId

```
quickSpinStringNode DevicePortId
```

12.3.1.15 DeviceReset

```
quickSpinCommandNode DeviceReset
```

12.3.1.16 DeviceSerialNumber

```
quickSpinStringNode DeviceSerialNumber
```

12.3.1.17 DeviceType

`quickSpinEnumerationNode DeviceType`

12.3.1.18 DeviceU3VProtocol

`quickSpinBooleanNode DeviceU3VProtocol`

12.3.1.19 DeviceUserID

`quickSpinStringNode DeviceUserID`

12.3.1.20 DeviceVendorName

`quickSpinStringNode DeviceVendorName`

12.3.1.21 DeviceVersion

`quickSpinStringNode DeviceVersion`

12.3.1.22 GenICamXMLLocation

`quickSpinEnumerationNode GenICamXMLLocation`

12.3.1.23 GenICamXMLPath

`quickSpinStringNode GenICamXMLPath`

12.3.1.24 GevCCP

`quickSpinEnumerationNode GevCCP`

12.3.1.25 GevDeviceAutoForceIP

```
quickSpinCommandNode GevDeviceAutoForceIP
```

12.3.1.26 GevDeviceDiscoverMaximumPacketSize

```
quickSpinCommandNode GevDeviceDiscoverMaximumPacketSize
```

12.3.1.27 GevDeviceForceGateway

```
quickSpinIntegerNode GevDeviceForceGateway
```

12.3.1.28 GevDeviceForceIP

```
quickSpinCommandNode GevDeviceForceIP
```

12.3.1.29 GevDeviceForceIPAddress

```
quickSpinIntegerNode GevDeviceForceIPAddress
```

12.3.1.30 GevDeviceForceSubnetMask

```
quickSpinIntegerNode GevDeviceForceSubnetMask
```

12.3.1.31 GevDeviceGateway

```
quickSpinIntegerNode GevDeviceGateway
```

12.3.1.32 GevDeviceIPAddress

```
quickSpinIntegerNode GevDeviceIPAddress
```

12.3.1.33 GevDeviceIsWrongSubnet

```
quickSpinBooleanNode GevDeviceIsWrongSubnet
```

12.3.1.34 GevDeviceMACAddress

```
quickSpinIntegerNode GevDeviceMACAddress
```

12.3.1.35 GevDeviceMaximumPacketSize

```
quickSpinIntegerNode GevDeviceMaximumPacketSize
```

12.3.1.36 GevDeviceMaximumRetryCount

```
quickSpinIntegerNode GevDeviceMaximumRetryCount
```

12.3.1.37 GevDeviceModeIsBigEndian

```
quickSpinBooleanNode GevDeviceModeIsBigEndian
```

12.3.1.38 GevDevicePort

```
quickSpinIntegerNode GevDevicePort
```

12.3.1.39 GevDeviceReadAndWriteTimeout

```
quickSpinIntegerNode GevDeviceReadAndWriteTimeout
```

12.3.1.40 GevDeviceSubnetMask

```
quickSpinIntegerNode GevDeviceSubnetMask
```

12.3.1.41 GevVersionMajor

```
quickSpinIntegerNode GevVersionMajor
```

12.3.1.42 GevVersionMinor

```
quickSpinIntegerNode GevVersionMinor
```

12.3.1.43 GUIXMLLocation

```
quickSpinEnumerationNode GUIXMLLocation
```

12.3.1.44 GUIXMLPath

```
quickSpinStringNode GUIXMLPath
```

The documentation for this struct was generated from the following file:

- include/spinc/TransportLayerDeviceC.h

12.4 quickSpinTLInterface Struct Reference

Data Fields

- quickSpinStringNode InterfaceID
- quickSpinStringNode InterfaceDisplayName
- quickSpinEnumerationNode InterfaceType
- quickSpinIntegerNode GevInterfaceGatewaySelector
- quickSpinIntegerNode GevInterfaceGateway
- quickSpinIntegerNode GevInterfaceMACAddress
- quickSpinIntegerNode GevInterfaceSubnetSelector
- quickSpinIntegerNode GevInterfaceSubnetIPAddress
- quickSpinIntegerNode GevInterfaceSubnetMask
- quickSpinIntegerNode GevInterfaceTransmitLinkSpeed
- quickSpinIntegerNode GevInterfaceReceiveLinkSpeed
- quickSpinIntegerNode GevInterfaceMTU
- quickSpinEnumerationNode POEStatus
- quickSpinEnumerationNode FilterDriverStatus
- quickSpinIntegerNode GevActionDeviceKey
- quickSpinIntegerNode GevActionGroupKey
- quickSpinIntegerNode GevActionGroupMask
- quickSpinIntegerNode GevActionTime

- quickSpinCommandNode ActionCommand
- quickSpinStringNode DeviceUnlock
- quickSpinCommandNode DeviceUpdateList
- quickSpinIntegerNode DeviceCount
- quickSpinIntegerNode DeviceSelector
- quickSpinStringNode DeviceID
- quickSpinStringNode DeviceVendorName
- quickSpinStringNode DeviceModelName
- quickSpinStringNode DeviceSerialNumber
- quickSpinEnumerationNode DeviceAccessStatus
- quickSpinIntegerNode GevDeviceIPAddress
- quickSpinIntegerNode GevDeviceSubnetMask
- quickSpinIntegerNode GevDeviceGateway
- quickSpinIntegerNode GevDeviceMACAddress
- quickSpinIntegerNode IncompatibleDeviceCount
- quickSpinIntegerNode IncompatibleDeviceSelector
- quickSpinStringNode IncompatibleDeviceID
- quickSpinStringNode IncompatibleDeviceVendorName
- quickSpinStringNode IncompatibleDeviceModelName
- quickSpinIntegerNode IncompatibleGevDeviceIPAddress
- quickSpinIntegerNode IncompatibleGevDeviceSubnetMask
- quickSpinIntegerNode IncompatibleGevDeviceMACAddress
- quickSpinCommandNode GevDeviceForceIP
- quickSpinIntegerNode GevDeviceForceIPAddress
- quickSpinIntegerNode GevDeviceForceSubnetMask
- quickSpinIntegerNode GevDeviceForceGateway
- quickSpinCommandNode GevDeviceAutoForceIP
- quickSpinStringNode HostAdapterName
- quickSpinStringNode HostAdapterVendor
- quickSpinStringNode HostAdapterDriverVersion

12.4.1 Field Documentation

12.4.1.1 ActionCommand

```
quickSpinCommandNode ActionCommand
```

12.4.1.2 DeviceAccessStatus

```
quickSpinEnumerationNode DeviceAccessStatus
```

12.4.1.3 DeviceCount

```
quickSpinIntegerNode DeviceCount
```

12.4.1.4 DeviceID

```
quickSpinStringNode DeviceID
```

12.4.1.5 DeviceModelName

```
quickSpinStringNode DeviceModelName
```

12.4.1.6 DeviceSelector

```
quickSpinIntegerNode DeviceSelector
```

12.4.1.7 DeviceSerialNumber

```
quickSpinStringNode DeviceSerialNumber
```

12.4.1.8 DeviceUnlock

```
quickSpinStringNode DeviceUnlock
```

12.4.1.9 DeviceUpdateList

```
quickSpinCommandNode DeviceUpdateList
```

12.4.1.10 DeviceVendorName

```
quickSpinStringNode DeviceVendorName
```

12.4.1.11 FilterDriverStatus

```
quickSpinEnumerationNode FilterDriverStatus
```

12.4.1.12 GevActionDeviceKey

```
quickSpinIntegerNode GevActionDeviceKey
```

12.4.1.13 GevActionGroupKey

```
quickSpinIntegerNode GevActionGroupKey
```

12.4.1.14 GevActionGroupMask

```
quickSpinIntegerNode GevActionGroupMask
```

12.4.1.15 GevActionTime

```
quickSpinIntegerNode GevActionTime
```

12.4.1.16 GevDeviceAutoForceIP

```
quickSpinCommandNode GevDeviceAutoForceIP
```

12.4.1.17 GevDeviceForceGateway

```
quickSpinIntegerNode GevDeviceForceGateway
```

12.4.1.18 GevDeviceForceIP

```
quickSpinCommandNode GevDeviceForceIP
```

12.4.1.19 GevDeviceForceIPAddress

```
quickSpinIntegerNode GevDeviceForceIPAddress
```

12.4.1.20 GevDeviceForceSubnetMask

```
quickSpinIntegerNode GevDeviceForceSubnetMask
```

12.4.1.21 GevDeviceGateway

```
quickSpinIntegerNode GevDeviceGateway
```

12.4.1.22 GevDeviceIPAddress

```
quickSpinIntegerNode GevDeviceIPAddress
```

12.4.1.23 GevDeviceMACAddress

```
quickSpinIntegerNode GevDeviceMACAddress
```

12.4.1.24 GevDeviceSubnetMask

```
quickSpinIntegerNode GevDeviceSubnetMask
```

12.4.1.25 GevInterfaceGateway

```
quickSpinIntegerNode GevInterfaceGateway
```

12.4.1.26 GevInterfaceGatewaySelector

```
quickSpinIntegerNode GevInterfaceGatewaySelector
```

12.4.1.27 GevInterfaceMACAddress

```
quickSpinIntegerNode GevInterfaceMACAddress
```

12.4.1.28 GevInterfaceMTU

```
quickSpinIntegerNode GevInterfaceMTU
```

12.4.1.29 GevInterfaceReceiveLinkSpeed

```
quickSpinIntegerNode GevInterfaceReceiveLinkSpeed
```

12.4.1.30 GevInterfaceSubnetIPAddress

```
quickSpinIntegerNode GevInterfaceSubnetIPAddress
```

12.4.1.31 GevInterfaceSubnetMask

```
quickSpinIntegerNode GevInterfaceSubnetMask
```

12.4.1.32 GevInterfaceSubnetSelector

```
quickSpinIntegerNode GevInterfaceSubnetSelector
```

12.4.1.33 GevInterfaceTransmitLinkSpeed

```
quickSpinIntegerNode GevInterfaceTransmitLinkSpeed
```

12.4.1.34 HostAdapterDriverVersion

```
quickSpinStringNode HostAdapterDriverVersion
```

12.4.1.35 HostAdapterName

```
quickSpinStringNode HostAdapterName
```

12.4.1.36 HostAdapterVendor

```
quickSpinStringNode HostAdapterVendor
```

12.4.1.37 IncompatibleDeviceCount

```
quickSpinIntegerNode IncompatibleDeviceCount
```

12.4.1.38 IncompatibleDeviceID

```
quickSpinStringNode IncompatibleDeviceID
```

12.4.1.39 IncompatibleDeviceModelName

```
quickSpinStringNode IncompatibleDeviceModelName
```

12.4.1.40 IncompatibleDeviceSelector

```
quickSpinIntegerNode IncompatibleDeviceSelector
```

12.4.1.41 IncompatibleDeviceVendorName

```
quickSpinStringNode IncompatibleDeviceVendorName
```

12.4.1.42 IncompatibleGevDeviceIPAddress

```
quickSpinIntegerNode IncompatibleGevDeviceIPAddress
```

12.4.1.43 IncompatibleGevDeviceMACAddress

```
quickSpinIntegerNode IncompatibleGevDeviceMACAddress
```

12.4.1.44 IncompatibleGevDeviceSubnetMask

```
quickSpinIntegerNode IncompatibleGevDeviceSubnetMask
```

12.4.1.45 InterfaceDisplayName

```
quickSpinStringNode InterfaceDisplayName
```

12.4.1.46 InterfaceID

```
quickSpinStringNode InterfaceID
```

12.4.1.47 InterfaceType

```
quickSpinEnumerationNode InterfaceType
```

12.4.1.48 POEStatus

```
quickSpinEnumerationNode POEStatus
```

The documentation for this struct was generated from the following file:

- include/spinc/TransportLayerInterfaceC.h

12.5 quickSpinTLStream Struct Reference

Data Fields

- quickSpinStringNode StreamID
- quickSpinEnumerationNode StreamType
- quickSpinEnumerationNode StreamMode
- quickSpinIntegerNode StreamBufferCountManual
- quickSpinIntegerNode StreamBufferCountResult
- quickSpinIntegerNode StreamBufferCountMax
- quickSpinEnumerationNode StreamBufferCountMode
- quickSpinEnumerationNode StreamBufferHandlingMode
- quickSpinIntegerNode StreamAnnounceBufferMinimum
- quickSpinIntegerNode StreamAnnouncedBufferCount
- quickSpinIntegerNode StreamStartedFrameCount
- quickSpinIntegerNode StreamDeliveredFrameCount
- quickSpinIntegerNode StreamReceivedFrameCount
- quickSpinIntegerNode StreamIncompleteFrameCount
- quickSpinIntegerNode StreamLostFrameCount
- quickSpinIntegerNode StreamDroppedFrameCount
- quickSpinIntegerNode StreamInputBufferCount
- quickSpinIntegerNode StreamOutputBufferCount
- quickSpinBooleanNode StreamIsGrabbing
- quickSpinIntegerNode StreamChunkCountMaximum
- quickSpinIntegerNode StreamBufferAlignment
- quickSpinBooleanNode StreamCRCCheckEnable
- quickSpinIntegerNode StreamReceivedPacketCount
- quickSpinIntegerNode StreamMissedPacketCount
- quickSpinBooleanNode StreamPacketResendEnable
- quickSpinIntegerNode StreamPacketResendTimeout
- quickSpinIntegerNode StreamPacketResendMaxRequests
- quickSpinIntegerNode StreamPacketResendRequestCount
- quickSpinIntegerNode StreamPacketResendRequestSuccessCount
- quickSpinIntegerNode StreamPacketResendRequestedPacketCount
- quickSpinIntegerNode StreamPacketResendReceivedPacketCount
- quickSpinIntegerNode StreamBlockTransferSize

12.5.1 Field Documentation

12.5.1.1 StreamAnnounceBufferMinimum

```
quickSpinIntegerNode StreamAnnounceBufferMinimum
```

12.5.1.2 StreamAnnouncedBufferCount

```
quickSpinIntegerNode StreamAnnouncedBufferCount
```

12.5.1.3 StreamBlockTransferSize

```
quickSpinIntegerNode StreamBlockTransferSize
```

12.5.1.4 StreamBufferAlignment

```
quickSpinIntegerNode StreamBufferAlignment
```

12.5.1.5 StreamBufferCountManual

```
quickSpinIntegerNode StreamBufferCountManual
```

12.5.1.6 StreamBufferCountMax

```
quickSpinIntegerNode StreamBufferCountMax
```

12.5.1.7 StreamBufferCountMode

```
quickSpinEnumerationNode StreamBufferCountMode
```

12.5.1.8 StreamBufferCountResult

```
quickSpinIntegerNode StreamBufferCountResult
```

12.5.1.9 StreamBufferHandlingMode

```
quickSpinEnumerationNode StreamBufferHandlingMode
```

12.5.1.10 StreamChunkCountMaximum

```
quickSpinIntegerNode StreamChunkCountMaximum
```

12.5.1.11 StreamCRCCheckEnable

```
quickSpinBooleanNode StreamCRCCheckEnable
```

12.5.1.12 StreamDeliveredFrameCount

```
quickSpinIntegerNode StreamDeliveredFrameCount
```

12.5.1.13 StreamDroppedFrameCount

```
quickSpinIntegerNode StreamDroppedFrameCount
```

12.5.1.14 StreamID

```
quickSpinStringNode StreamID
```

12.5.1.15 StreamIncompleteFrameCount

```
quickSpinIntegerNode StreamIncompleteFrameCount
```

12.5.1.16 StreamInputBufferCount

```
quickSpinIntegerNode StreamInputBufferCount
```

12.5.1.17 StreamIsGrabbing

```
quickSpinBooleanNode StreamIsGrabbing
```

12.5.1.18 StreamLostFrameCount

```
quickSpinIntegerNode StreamLostFrameCount
```

12.5.1.19 StreamMissedPacketCount

```
quickSpinIntegerNode StreamMissedPacketCount
```

12.5.1.20 StreamMode

```
quickSpinEnumerationNode StreamMode
```

12.5.1.21 StreamOutputBufferCount

```
quickSpinIntegerNode StreamOutputBufferCount
```

12.5.1.22 StreamPacketResendEnable

```
quickSpinBooleanNode StreamPacketResendEnable
```

12.5.1.23 StreamPacketResendMaxRequests

```
quickSpinIntegerNode StreamPacketResendMaxRequests
```

12.5.1.24 StreamPacketResendReceivedPacketCount

```
quickSpinIntegerNode StreamPacketResendReceivedPacketCount
```

12.5.1.25 StreamPacketResendRequestCount

```
quickSpinIntegerNode StreamPacketResendRequestCount
```

12.5.1.26 StreamPacketResendRequestedPacketCount

```
quickSpinIntegerNode StreamPacketResendRequestedPacketCount
```

12.5.1.27 StreamPacketResendRequestSuccessCount

```
quickSpinIntegerNode StreamPacketResendRequestSuccessCount
```

12.5.1.28 StreamPacketResendTimeout

```
quickSpinIntegerNode StreamPacketResendTimeout
```

12.5.1.29 StreamReceivedFrameCount

```
quickSpinIntegerNode StreamReceivedFrameCount
```

12.5.1.30 StreamReceivedPacketCount

```
quickSpinIntegerNode StreamReceivedPacketCount
```

12.5.1.31 StreamStartedFrameCount

```
quickSpinIntegerNode StreamStartedFrameCount
```

12.5.1.32 StreamType

```
quickSpinEnumerationNode StreamType
```

The documentation for this struct was generated from the following file:

- include/spinc/TransportLayerStreamC.h

12.6 quickSpinTLSystem Struct Reference

Data Fields

- quickSpinStringNode TLID
- quickSpinStringNode TLVendorName
- quickSpinStringNode TLModelName
- quickSpinStringNode TLVersion
- quickSpinStringNode TLFileName
- quickSpinStringNode TLDisplayName
- quickSpinStringNode TLPath
- quickSpinEnumerationNode TLType
- quickSpinIntegerNode GenTLVersionMajor
- quickSpinIntegerNode GenTLVersionMinor
- quickSpinIntegerNode GenTLSFNCVersionMajor
- quickSpinIntegerNode GenTLSFNCVersionMinor
- quickSpinIntegerNode GenTLSFNCVersionSubMinor
- quickSpinIntegerNode GevVersionMajor
- quickSpinIntegerNode GevVersionMinor
- quickSpinCommandNode InterfaceUpdateList
- quickSpinIntegerNode InterfaceSelector
- quickSpinStringNode InterfaceID
- quickSpinStringNode InterfaceDisplayName
- quickSpinIntegerNode GevInterfaceMACAddress
- quickSpinIntegerNode GevInterfaceDefaultIPAddress
- quickSpinIntegerNode GevInterfaceDefaultSubnetMask
- quickSpinIntegerNode GevInterfaceDefaultGateway
- quickSpinBooleanNode EnumerateGEVInterfaces
- quickSpinBooleanNode EnumerateUSBInterfaces
- quickSpinBooleanNode EnumerateGen2Cameras

12.6.1 Field Documentation

12.6.1.1 EnumerateGen2Cameras

```
quickSpinBooleanNode EnumerateGen2Cameras
```

12.6.1.2 EnumerateGEVInterfaces

```
quickSpinBooleanNode EnumerateGEVInterfaces
```

12.6.1.3 EnumerateUSBInterfaces

```
quickSpinBooleanNode EnumerateUSBInterfaces
```

12.6.1.4 GenTLSFNCVersionMajor

```
quickSpinIntegerNode GenTLSFNCVersionMajor
```

12.6.1.5 GenTLSFNCVersionMinor

```
quickSpinIntegerNode GenTLSFNCVersionMinor
```

12.6.1.6 GenTLSFNCVersionSubMinor

```
quickSpinIntegerNode GenTLSFNCVersionSubMinor
```

12.6.1.7 GenTLVersionMajor

```
quickSpinIntegerNode GenTLVersionMajor
```

12.6.1.8 GenTLVersionMinor

```
quickSpinIntegerNode GenTLVersionMinor
```

12.6.1.9 GevInterfaceDefaultGateway

```
quickSpinIntegerNode GevInterfaceDefaultGateway
```

12.6.1.10 GevInterfaceDefaultIPAddress

```
quickSpinIntegerNode GevInterfaceDefaultIPAddress
```

12.6.1.11 GevInterfaceDefaultSubnetMask

```
quickSpinIntegerNode GevInterfaceDefaultSubnetMask
```

12.6.1.12 GevInterfaceMACAddress

```
quickSpinIntegerNode GevInterfaceMACAddress
```

12.6.1.13 GevVersionMajor

```
quickSpinIntegerNode GevVersionMajor
```

12.6.1.14 GevVersionMinor

```
quickSpinIntegerNode GevVersionMinor
```

12.6.1.15 InterfaceDisplayName

```
quickSpinStringNode InterfaceDisplayName
```

12.6.1.16 InterfaceID

```
quickSpinStringNode InterfaceID
```

12.6.1.17 InterfaceSelector

```
quickSpinIntegerNode InterfaceSelector
```

12.6.1.18 InterfaceUpdateList

```
quickSpinCommandNode InterfaceUpdateList
```

12.6.1.19 TLDisplayName

```
quickSpinStringNode TLDisplayName
```

12.6.1.20 TLFileName

```
quickSpinStringNode TLFileName
```

12.6.1.21 TLID

```
quickSpinStringNode TLID
```

12.6.1.22 TLModelName

```
quickSpinStringNode TLModelName
```

12.6.1.23 TLPath

```
quickSpinStringNode TLPath
```

12.6.1.24 TLType

```
quickSpinEnumerationNode TLType
```

12.6.1.25 TLVendorName

```
quickSpinStringNode TLVendorName
```

12.6.1.26 TLVersion

```
quickSpinStringNode TLVersion
```

The documentation for this struct was generated from the following file:

- include/spinc/TransportLayerSystemC.h

12.7 spinAVIOption Struct Reference

Options for saving uncompressed videos.

Data Fields

- float frameRate

  Frame rate of the stream.
- unsigned int width

  Width of source image.
- unsigned int height

  Height of source image.
- unsigned int reserved [192]

12.7.1 Detailed Description

Options for saving uncompressed videos.

Used in saving AVI videos with a call to spinAVIRecorderOpenUncompressed().

12.7.2 Field Documentation

12.7.2.1 frameRate

```
float frameRate
```

Frame rate of the stream.

12.7.2.2 height

```
unsigned int height
```

Height of source image.

12.7.2.3 reserved

```
unsigned int reserved[192]
```

12.7.2.4 width

```
unsigned int width
```

Width of source image.

The documentation for this struct was generated from the following file:

- include/spinc/SpinnakerDefsC.h

12.8 spinBMPOption Struct Reference

Options for saving BMP images.

Data Fields

- bool8_t indexedColor_8bit
- unsigned int reserved [16]

  Reserved for future use.

12.8.1 Detailed Description

Options for saving BMP images.

Used in saving PPM images with a call to spinImageSaveBmp().

12.8.2 Field Documentation

12.8.2.1 indexedColor_8bit

```
bool8_t indexedColor_8bit
```

12.8.2.2 reserved

```
unsigned int reserved[16]
```

Reserved for future use.

The documentation for this struct was generated from the following file:

- include/spinc/SpinnakerDefsC.h

12.9 spinChunkData Struct Reference

The type of information that can be obtained from image chunk data.

Data Fields

- double m_blackLevel
- int64_t m_frameID
- double m_exposureTime
- int64_t m_compressionMode
- double m_compressionRatio
- int64_t m_timestamp
- int64_t m_exposureEndLineStatusAll
- int64_t m_width
- int64_t m_image
- int64_t m_height
- double m_gain
- int64_t m_sequencerSetActive
- int64_t m_cRC
- int64_t m_offsetX
- int64_t m_offsetY
- int64_t m_serialDataLength
- int64_t m_partSelector
- int64_t m_pixelDynamicRangeMin
- int64_t m_pixelDynamicRangeMax
- int64_t m_timestampLatchValue
- int64_t m_lineStatusAll
- int64_t m_counterValue
- double m_timerValue
- int64_t m_scanLineSelector
- int64_t m_encoderValue
- int64_t m_linePitch
- int64_t m_transferBlockID
- int64_t m_transferQueueCurrentBlockCount
- int64_t m_streamChannelID
- double m_scan3dCoordinateScale
- double m_scan3dCoordinateOffset
- double m_scan3dInvalidDataValue
- double m_scan3dAxisMin
- double m_scan3dAxisMax
- double m_scan3dTransformValue
- double m_scan3dCoordinateReferenceValue
- int64_t m_inferenceFrameId
- int64_t m_inferenceResult
- double m_inferenceConfidence

12.9.1 Detailed Description

The type of information that can be obtained from image chunk data.

12.9.2 Field Documentation

12.9.2.1 m_blackLevel

```
double m_blackLevel
```

12.9.2.2 m_compressionMode

```
int64_t m_compressionMode
```

12.9.2.3 m_compressionRatio

```
double m_compressionRatio
```

12.9.2.4 m_counterValue

```
int64_t m_counterValue
```

12.9.2.5 m_cRC

```
int64_t m_cRC
```

12.9.2.6 m_encoderValue

```
int64_t m_encoderValue
```

12.9.2.7 m_exposureEndLineStatusAll

```
int64_t m_exposureEndLineStatusAll
```

12.9.2.8 m_exposureTime

```
double m_exposureTime
```

12.9.2.9 m_frameID

```
int64_t m_frameID
```

12.9.2.10 m_gain

```
double m_gain
```

12.9.2.11 m_height

```
int64_t m_height
```

12.9.2.12 m_image

```
int64_t m_image
```

12.9.2.13 m_inferenceConfidence

```
double m_inferenceConfidence
```

12.9.2.14 m_inferenceFrameId

```
int64_t m_inferenceFrameId
```

12.9.2.15 m_inferenceResult

```
int64_t m_inferenceResult
```

12.9.2.16 m_linePitch

```
int64_t m_linePitch
```

12.9.2.17 m_lineStatusAll

```
int64_t m_lineStatusAll
```

12.9.2.18 m_offsetX

```
int64_t m_offsetX
```

12.9.2.19 m_offsetY

```
int64_t m_offsetY
```

12.9.2.20 m_partSelector

```
int64_t m_partSelector
```

12.9.2.21 m_pixelDynamicRangeMax

```
int64_t m_pixelDynamicRangeMax
```

12.9.2.22 m_pixelDynamicRangeMin

```
int64_t m_pixelDynamicRangeMin
```

12.9.2.23 m_scan3dAxisMax

```
double m_scan3dAxisMax
```

12.9.2.24 m_scan3dAxisMin

```
double m_scan3dAxisMin
```

12.9.2.25 m_scan3dCoordinateOffset

```
double m_scan3dCoordinateOffset
```

12.9.2.26 m_scan3dCoordinateReferenceValue

```
double m_scan3dCoordinateReferenceValue
```

12.9.2.27 m_scan3dCoordinateScale

```
double m_scan3dCoordinateScale
```

12.9.2.28 m_scan3dInvalidDataValue

```
double m_scan3dInvalidDataValue
```

12.9.2.29 m_scan3dTransformValue

```
double m_scan3dTransformValue
```

12.9.2.30 m_scanLineSelector

```
int64_t m_scanLineSelector
```

12.9.2.31 m_sequencerSetActive

```
int64_t m_sequencerSetActive
```

12.9.2.32 m_serialDataLength

```
int64_t m_serialDataLength
```

12.9.2.33 m_streamChannelID

```
int64_t m_streamChannelID
```

12.9.2.34 m_timerValue

```
double m_timerValue
```

12.9.2.35 m_timestamp

```
int64_t m_timestamp
```

12.9.2.36 m_timestampLatchValue

```
int64_t m_timestampLatchValue
```

12.9.2.37 m_transferBlockID

```
int64_t m_transferBlockID
```

12.9.2.38 m_transferQueueCurrentBlockCount

```
int64_t m_transferQueueCurrentBlockCount
```

12.9.2.39 m_width

```
int64_t m_width
```

The documentation for this struct was generated from the following file:

- include/spinc/ChunkDataDefC.h

12.10 spinH264Option Struct Reference

Options for saving H264 videos.

Data Fields

- float frameRate

  Frame rate of the stream.
- unsigned int width

  Width of source image.
- unsigned int height

  Height of source image.
- unsigned int bitrate

  Bitrate to encode at.
- unsigned int reserved [256]

  Reserved for future use.

12.10.1 Detailed Description

Options for saving H264 videos.

Used in saving H264 videos with a call to spinAVIRecorderOpenH264().

12.10.2 Field Documentation

12.10.2.1 bitrate

```
unsigned int bitrate
```

Bitrate to encode at.

12.10.2.2 frameRate

```
float frameRate
```

Frame rate of the stream.

12.10.2.3 height

```
unsigned int height
```

Height of source image.

12.10.2.4 reserved

```
unsigned int reserved[256]
```

Reserved for future use.

12.10.2.5 width

```
unsigned int width
```

Width of source image.

The documentation for this struct was generated from the following file:

- include/spinc/SpinnakerDefsC.h

12.11 spinJPEGOption Struct Reference

Options for saving JPEG images.

Data Fields

- bool8_t progressive

  Whether to save as a progressive JPEG file.
- unsigned int quality

  JPEG image quality in range (0-100).
- unsigned int reserved [16]

  Reserved for future use.

12.11.1 Detailed Description

Options for saving JPEG images.

Used in saving PPM images with a call to spinImageSaveJpeg().

12.11.2 Field Documentation

12.11.2.1 progressive

```
bool8_t progressive
```

Whether to save as a progressive JPEG file.

12.11.2.2 quality

```
unsigned int quality
```

JPEG image quality in range (0-100).

- 100 - Superb quality.
- 75 - Good quality.
- 50 - Normal quality.
- 10 - Poor quality.

12.11.2.3 reserved

```
unsigned int reserved[16]
```

Reserved for future use.

The documentation for this struct was generated from the following file:

- include/spinc/SpinnakerDefsC.h

12.12 spinJPG2Option Struct Reference

Options for saving JPEG 2000 images.

Data Fields

- unsigned int quality

    JPEG saving quality in range (1-512).
- unsigned int reserved [16]

    Reserved for future use.

12.12.1 Detailed Description

Options for saving JPEG 2000 images.

Used in saving PPM images with a call to spinImageSaveJpg2().

12.12.2 Field Documentation

12.12.2.1 quality

```
unsigned int quality
```

JPEG saving quality in range (1-512).

12.12.2.2 reserved

```
unsigned int reserved[16]
```

Reserved for future use.

The documentation for this struct was generated from the following file:

- include/spinc/SpinnakerDefsC.h

12.13 spinLibraryVersion Struct Reference

Provides easier access to the current version of Spinnaker.

Data Fields

- unsigned int major

    Major version of the library.
- unsigned int minor

    Minor version of the library.
- unsigned int type

    Version type of the library.
- unsigned int build

    Build number of the library.

12.13.1 Detailed Description

Provides easier access to the current version of Spinnaker.

12.13.2 Field Documentation

12.13.2.1 build

```
unsigned int build
```

Build number of the library.

12.13.2.2 major

```
unsigned int major
```

Major version of the library.

12.13.2.3 minor

```
unsigned int minor
```

Minor version of the library.

12.13.2.4 type

```
unsigned int type
```

Version type of the library.

The documentation for this struct was generated from the following file:

- include/spinc/SpinnakerDefsC.h

12.14 spinMJPGOption Struct Reference

Options for saving MJPG videos.

Data Fields

- float frameRate

    Frame rate of the stream.
- unsigned int quality

    Image quality (1-100)
- unsigned int width

    Width of source image.
- unsigned int height

    Height of source image.
- unsigned int reserved [192]

12.14.1 Detailed Description

Options for saving MJPG videos.

Used in saving MJPG videos with a call to spinAVIRecorderOpenMJPG().

12.14.2 Field Documentation

12.14.2.1 frameRate

```
float frameRate
```

Frame rate of the stream.

12.14.2.2 height

```
unsigned int height
```

Height of source image.

12.14.2.3 quality

```
unsigned int quality
```

Image quality (1-100)

12.14.2.4 reserved

```
unsigned int reserved[192]
```

12.14.2.5 width

```
unsigned int width
```

Width of source image.

The documentation for this struct was generated from the following file:

- include/spinc/SpinnakerDefsC.h

12.15 spinPGMOption Struct Reference

Options for saving PGM images.

Data Fields

- bool8_t binaryFile
  Whether to save the PPM as a binary file.
- unsigned int reserved [16]
  Reserved for future use.

12.15.1 Detailed Description

Options for saving PGM images.

12.15.2 Field Documentation

12.15.2.1 binaryFile

```
bool8_t binaryFile
```

Whether to save the PPM as a binary file.

12.15.2.2 reserved

```
unsigned int reserved[16]
```

Reserved for future use.

The documentation for this struct was generated from the following file:

- include/spinc/SpinnakerDefsC.h

12.16 spinPNGOption Struct Reference

Options for saving PNG images.

Data Fields

- bool8_t interlaced

  Whether to save the PNG as interlaced.
- unsigned int compressionLevel

  Compression level (0-9).
- unsigned int reserved [16]

  Reserved for future use.

12.16.1 Detailed Description

Options for saving PNG images.

Used in saving PNG images with a call to spinImageSavePng().

12.16.2 Field Documentation

12.16.2.1 compressionLevel

```
unsigned int compressionLevel
```

Compression level (0-9).

0 is no compression, 9 is best compression.

12.16.2.2 interlaced

```
bool8_t interlaced
```

Whether to save the PNG as interlaced.

12.16.2.3 reserved

```
unsigned int reserved[16]
```

Reserved for future use.

The documentation for this struct was generated from the following file:

- include/spinc/SpinnakerDefsC.h

12.17 spinPPMOption Struct Reference

Options for saving PPM images.

Data Fields

- bool8_t binaryFile

  Whether to save the PPM as a binary file.
- unsigned int reserved [16]

  Reserved for future use.

12.17.1 Detailed Description

Options for saving PPM images.

Used in saving PPM images with a call to spinImageSavePpm().

12.17.2 Field Documentation

12.17.2.1 binaryFile

```
bool8_t binaryFile
```

Whether to save the PPM as a binary file.

12.17.2.2 reserved

```
unsigned int reserved[16]
```

Reserved for future use.

The documentation for this struct was generated from the following file:

- include/spinc/SpinnakerDefsC.h

12.18 spinTIFFOption Struct Reference

Options for saving TIFF images.

Data Fields

- spinTIFFCompressionMethod compression

    Compression method to use for encoding TIFF images.
- unsigned int reserved [16]

    Reserved for future use.

12.18.1 Detailed Description

Options for saving TIFF images.

Used in saving PPM images with a call to spinImageSaveTiff().

12.18.2 Field Documentation

12.18.2.1 compression

```
spinTIFFCompressionMethod compression
```

Compression method to use for encoding TIFF images.

12.18.2.2 reserved

```
unsigned int reserved[16]
```

Reserved for future use.

The documentation for this struct was generated from the following file:

- include/spinc/SpinnakerDefsC.h

Chapter 13

File Documentation

13.1 doc/spindocs/C/GettingStarted.dox File Reference

13.2 doc/spindocs/C/ProgrammerGuide.dox File Reference

13.3 doc/spindocs/shared/Benefits.dox File Reference

13.4 doc/spindocs/shared/FlyCapture2Comparison.dox File Reference

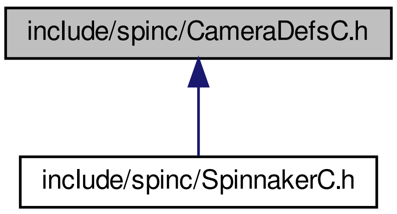
13.5 doc/spindocs/shared/GenICamGenTL.dox File Reference

13.6 doc/spindocs/shared/Licensing.dox File Reference

13.7 doc/spindocs/shared/Maintenance.dox File Reference

13.8 include/spinc/CameraDefsC.h File Reference

This graph shows which files directly or indirectly include this file:



Enumerations

- enum spinLUTSelectorEnums {
  LUTSelector_LUT1 ,
  NUM_LUTSELECTOR }

  The enum definitions for camera nodes.
- enum spinExposureModeEnums {
  ExposureMode_Timed ,
  ExposureMode_TriggerWidth ,
  NUM_EXPOSUREMODE }
- enum spinAcquisitionModeEnums {
  AcquisitionMode_Continuous ,
  AcquisitionMode_SingleFrame ,
  AcquisitionMode_MultiFrame ,
  NUM_ACQUISITIONMODE }
- enum spinTriggerSourceEnums {
  TriggerSource_Software ,
  TriggerSource_Line0 ,
  TriggerSource_Line1 ,
  TriggerSource_Line2 ,
  TriggerSource_Line3 ,
  TriggerSource_UserOutput0 ,
  TriggerSource_UserOutput1 ,
  TriggerSource_UserOutput2 ,
  TriggerSource_UserOutput3 ,
  TriggerSource_Counter0Start ,
  TriggerSource_Counter1Start ,
  TriggerSource_Counter0End ,
  TriggerSource_Counter1End ,
  TriggerSource_LogicBlock0 ,
  TriggerSource_LogicBlock1 ,
  TriggerSource_Action0 ,
  NUM_TRIGGERSOURCE }
- enum spinTriggerActivationEnums {
  TriggerActivation_LevelLow ,
  TriggerActivation_LevelHigh ,
  TriggerActivation_FallingEdge ,
  TriggerActivation_RisingEdge ,
  TriggerActivation_AnyEdge ,
  NUM_TRIGGERACTIVATION }
- enum spinSensorShutterModeEnums {
  SensorShutterMode_Global ,
  SensorShutterMode_Rolling ,
  SensorShutterMode_GlobalReset ,
  NUM_SENSORSHUTTERMODE }
- enum spinTriggerModeEnums {
  TriggerMode_Off ,
  TriggerMode_On ,
  NUM_TRIGGERMODE }
- enum spinTriggerOverlapEnums {
  TriggerOverlap_Off ,
  TriggerOverlap_ReadOut ,
  TriggerOverlap_PreviousFrame ,
  NUM_TRIGGEROVERLAP }
- enum spinTriggerSelectorEnums {
  TriggerSelector_AcquisitionStart ,
  TriggerSelector_FrameStart ,

TriggerSelector_FrameBurstStart ,
NUM_TRIGGERSELECTOR }
- enum spinExposureAutoEnums {
ExposureAuto_Off ,
ExposureAuto_Once ,
ExposureAuto_Continuous ,
NUM_EXPOSUREAUTO }
- enum spinEventSelectorEnums {
EventSelector_Error ,
EventSelector_ExposureEnd ,
EventSelector_SerialPortReceive ,
NUM_EVENTSELECTOR }
- enum spinEventNotificationEnums {
EventNotification_On ,
EventNotification_Off ,
NUM_EVENTNOTIFICATION }
- enum spinLogicBlockSelectorEnums {
LogicBlockSelector_LogicBlock0 ,
LogicBlockSelector_LogicBlock1 ,
NUM_LOGICBLOCKSELECTOR }
- enum spinLogicBlockLUTInputActivationEnums {
LogicBlockLUTInputActivation_LevelLow ,
LogicBlockLUTInputActivation_LevelHigh ,
LogicBlockLUTInputActivation_FallingEdge ,
LogicBlockLUTInputActivation_RisingEdge ,
LogicBlockLUTInputActivation_AnyEdge ,
NUM_LOGICBLOCKLUTINPUTACTIVATION }
- enum spinLogicBlockLUTInputSelectorEnums {
LogicBlockLUTInputSelector_Input0 ,
LogicBlockLUTInputSelector_Input1 ,
LogicBlockLUTInputSelector_Input2 ,
LogicBlockLUTInputSelector_Input3 ,
NUM_LOGICBLOCKLUTINPUTSELECTOR }
- enum spinLogicBlockLUTInputSourceEnums {
LogicBlockLUTInputSource_Zero ,
LogicBlockLUTInputSource_Line0 ,
LogicBlockLUTInputSource_Line1 ,
LogicBlockLUTInputSource_Line2 ,
LogicBlockLUTInputSource_Line3 ,
LogicBlockLUTInputSource_UserOutput0 ,
LogicBlockLUTInputSource_UserOutput1 ,
LogicBlockLUTInputSource_UserOutput2 ,
LogicBlockLUTInputSource_UserOutput3 ,
LogicBlockLUTInputSource_Counter0Start ,
LogicBlockLUTInputSource_Counter1Start ,
LogicBlockLUTInputSource_Counter0End ,
LogicBlockLUTInputSource_Counter1End ,
LogicBlockLUTInputSource_LogicBlock0 ,
LogicBlockLUTInputSource_LogicBlock1 ,
LogicBlockLUTInputSource_ExposureStart ,
LogicBlockLUTInputSource_ExposureEnd ,
LogicBlockLUTInputSource_FrameTriggerWait ,
LogicBlockLUTInputSource_AcquisitionActive ,
NUM_LOGICBLOCKLUTINPUTSOURCE }
- enum spinLogicBlockLUTSelectorEnums {
LogicBlockLUTSelector_Value ,
LogicBlockLUTSelector_Enable ,
NUM_LOGICBLOCKLUTSELECTOR }

- enum spinColorTransformationSelectorEnums {
  ColorTransformationSelector_RGBtoRGB ,
  ColorTransformationSelector_RGBtoYUV ,
  NUM_COLORTRANSFORMATIONSELECTOR }
- enum spinRgbTransformLightSourceEnums {
  RgbTransformLightSource_General ,
  RgbTransformLightSource_Tungsten2800K ,
  RgbTransformLightSource_WarmFluorescent3000K ,
  RgbTransformLightSource_CoolFluorescent4000K ,
  RgbTransformLightSource_Daylight5000K ,
  RgbTransformLightSource_Cloudy6500K ,
  RgbTransformLightSource_Shade8000K ,
  RgbTransformLightSource_Custom ,
  NUM_RGBTRANSFORMLIGHTSOURCE }
- enum spinColorTransformationValueSelectorEnums {
  ColorTransformationValueSelector_Gain00 ,
  ColorTransformationValueSelector_Gain01 ,
  ColorTransformationValueSelector_Gain02 ,
  ColorTransformationValueSelector_Gain10 ,
  ColorTransformationValueSelector_Gain11 ,
  ColorTransformationValueSelector_Gain12 ,
  ColorTransformationValueSelector_Gain20 ,
  ColorTransformationValueSelector_Gain21 ,
  ColorTransformationValueSelector_Gain22 ,
  ColorTransformationValueSelector_Offset0 ,
  ColorTransformationValueSelector_Offset1 ,
  ColorTransformationValueSelector_Offset2 ,
  NUM_COLORTRANSFORMATIONVALUESELECTOR }
- enum spinDeviceRegistersEndiannessEnums {
  DeviceRegistersEndianness_Little ,
  DeviceRegistersEndianness_Big ,
  NUM_DEVICEREGISTERSENDIANNESS }
- enum spinDeviceScanTypeEnums {
  DeviceScanType_Areascan ,
  NUM_DEVICESCANTYPE }
- enum spinDeviceCharacterSetEnums {
  DeviceCharacterSet_UTF8 ,
  DeviceCharacterSet_ASCII ,
  NUM_DEVICECHARACTERSET }
- enum spinDeviceTLTypeEnums {
  DeviceTLType_GigEVision ,
  DeviceTLType_CameraLink ,
  DeviceTLType_CameraLinkHS ,
  DeviceTLType_CoaXPress ,
  DeviceTLType_USB3Vision ,
  DeviceTLType_Custom ,
  NUM_DEVICETLTYPE }
- enum spinDevicePowerSupplySelectorEnums {
  DevicePowerSupplySelector_External ,
  NUM_DEVICEPOWERSUPPLYSELECTOR }
- enum spinDeviceTemperatureSelectorEnums {
  DeviceTemperatureSelector_Sensor ,
  NUM_DEVICETEMPERATURESELECTOR }
- enum spinDeviceIndicatorModeEnums {
  DeviceIndicatorMode_Inactive ,
  DeviceIndicatorMode_Active ,
  DeviceIndicatorMode_ErrorStatus ,
  NUM_DEVICEINDICATORMODE }

- enum spinAutoExposureControlPriorityEnums {
  AutoExposureControlPriority_Gain ,
  AutoExposureControlPriority_ExposureTime ,
  NUM_AUTOEXPOSURECONTROLPRIORITY }
- enum spinAutoExposureMeteringModeEnums {
  AutoExposureMeteringMode_Average ,
  AutoExposureMeteringMode_Spot ,
  AutoExposureMeteringMode_Partial ,
  AutoExposureMeteringMode_CenterWeighted ,
  AutoExposureMeteringMode_HistgramPeak ,
  NUM_AUTOEXPOSUREMETERINGMODE }
- enum spinBalanceWhiteAutoProfileEnums {
  BalanceWhiteAutoProfile_Indoor ,
  BalanceWhiteAutoProfile_Outdoor ,
  NUM_BALANCEWHITEAUTOPROFILE }
- enum spinAutoAlgorithmSelectorEnums {
  AutoAlgorithmSelector_Awb ,
  AutoAlgorithmSelector_Ae ,
  NUM_AUTOALGORITHMSELECTOR }
- enum spinAutoExposureTargetGreyValueAutoEnums {
  AutoExposureTargetGreyValueAuto_Off ,
  AutoExposureTargetGreyValueAuto_Continuous ,
  NUM_AUTOEXPOSURETARGETGREYVALUEAUTO }
- enum spinAutoExposureLightingModeEnums {
  AutoExposureLightingMode_AutoDetect ,
  AutoExposureLightingMode_Backlight ,
  AutoExposureLightingMode_Frontlight ,
  AutoExposureLightingMode_Normal ,
  NUM_AUTOEXPOSURELIGHTINGMODE }
- enum spinGevIEEE1588StatusEnums {
  GevIEEE1588Status_Initializing ,
  GevIEEE1588Status_Faulty ,
  GevIEEE1588Status_Disabled ,
  GevIEEE1588Status_Listening ,
  GevIEEE1588Status_PreMaster ,
  GevIEEE1588Status_Master ,
  GevIEEE1588Status_Passive ,
  GevIEEE1588Status_Uncalibrated ,
  GevIEEE1588Status_Slave ,
  NUM_GEVIEEE1588STATUS }
- enum spinGevIEEE1588ModeEnums {
  GevIEEE1588Mode_Auto ,
  GevIEEE1588Mode_SlaveOnly ,
  NUM_GEVIEEE1588MODE }
- enum spinGevIEEE1588ClockAccuracyEnums {
  GevIEEE1588ClockAccuracy_Unknown ,
  NUM_GEVIEEE1588CLOCKACCURACY }
- enum spinGevCCPEnums {
  GevCCP_OpenAccess ,
  GevCCP_ExclusiveAccess ,
  GevCCP_ControlAccess ,
  NUM_GEVCCP }
- enum spinGevSupportedOptionSelectorEnums {
  GevSupportedOptionSelector_UserDefinedName ,
  GevSupportedOptionSelector_SerialNumber ,
  GevSupportedOptionSelector_HeartbeatDisable ,
  GevSupportedOptionSelector_LinkSpeed ,
  GevSupportedOptionSelector_CCPApplicationSocket ,

GevSupportedOptionSelector_ManifestTable ,
GevSupportedOptionSelector_TestData ,
GevSupportedOptionSelector_DiscoveryAckDelay ,
GevSupportedOptionSelector_DiscoveryAckDelayWritable ,
GevSupportedOptionSelector_ExtendedStatusCodes ,
GevSupportedOptionSelector_Action ,
GevSupportedOptionSelector_PendingAck ,
GevSupportedOptionSelector_EventData ,
GevSupportedOptionSelector_Event ,
GevSupportedOptionSelector_PacketResend ,
GevSupportedOptionSelector_WriteMem ,
GevSupportedOptionSelector_CommandsConcatenation ,
GevSupportedOptionSelector_IPConfigurationLLA ,
GevSupportedOptionSelector_IPConfigurationDHCP ,
GevSupportedOptionSelector_IPConfigurationPersistentIP ,
GevSupportedOptionSelector_StreamChannelSourceSocket ,
GevSupportedOptionSelector_MessageChannelSourceSocket ,
NUM_GEVSUPPORTEDOPTIONSELECTOR }

- enum spinBlackLevelSelectorEnums {
BlackLevelSelector_All ,
BlackLevelSelector_Analog ,
BlackLevelSelector_Digital ,
NUM_BLACKLEVELSELECTOR }
- enum spinBalanceWhiteAutoEnums {
BalanceWhiteAuto_Off ,
BalanceWhiteAuto_Once ,
BalanceWhiteAuto_Continuous ,
NUM_BALANCEWHITEAUTO }
- enum spinGainAutoEnums {
GainAuto_Off ,
GainAuto_Once ,
GainAuto_Continuous ,
NUM_GAINAUTO }
- enum spinBalanceRatioSelectorEnums {
BalanceRatioSelector_Red ,
BalanceRatioSelector_Blue ,
NUM_BALANCERATIOSELECTOR }
- enum spinGainSelectorEnums {
GainSelector_All ,
NUM_GAINSELECTOR }
- enum spinDefectCorrectionModeEnums {
DefectCorrectionMode_Average ,
DefectCorrectionMode_Highlight ,
DefectCorrectionMode_Zero ,
NUM_DEFECTCORRECTIONMODE }
- enum spinUserSetSelectorEnums {
UserSetSelector_Default ,
UserSetSelector_UserSet0 ,
UserSetSelector_UserSet1 ,
NUM_USERSETSELECTOR }
- enum spinUserSetDefaultEnums {
UserSetDefault_Default ,
UserSetDefault_UserSet0 ,
UserSetDefault_UserSet1 ,
NUM_USERSETDEFAULT }
- enum spinSerialPortBaudRateEnums {
SerialPortBaudRate_Baud300 ,
SerialPortBaudRate_Baud600 ,

SerialPortBaudRate_Baud1200 ,
SerialPortBaudRate_Baud2400 ,
SerialPortBaudRate_Baud4800 ,
SerialPortBaudRate_Baud9600 ,
SerialPortBaudRate_Baud14400 ,
SerialPortBaudRate_Baud19200 ,
SerialPortBaudRate_Baud38400 ,
SerialPortBaudRate_Baud57600 ,
SerialPortBaudRate_Baud115200 ,
SerialPortBaudRate_Baud230400 ,
SerialPortBaudRate_Baud460800 ,
SerialPortBaudRate_Baud921600 ,
NUM_SERIALPORTBAUDRATE }

- enum spinSerialPortParityEnums {
  SerialPortParity_None ,
  SerialPortParity_Odd ,
  SerialPortParity_Even ,
  SerialPortParity_Mark ,
  SerialPortParity_Space ,
  NUM_SERIALPORTPARITY }
- enum spinSerialPortSelectorEnums {
  SerialPortSelector_SerialPort0 ,
  NUM_SERIALPORTSELECTOR }
- enum spinSerialPortStopBitsEnums {
  SerialPortStopBits_Bits1 ,
  SerialPortStopBits_Bits1AndAHalf ,
  SerialPortStopBits_Bits2 ,
  NUM_SERIALPORTSTOPBITS }
- enum spinSerialPortSourceEnums {
  SerialPortSource_Line0 ,
  SerialPortSource_Line1 ,
  SerialPortSource_Line2 ,
  SerialPortSource_Line3 ,
  SerialPortSource_Off ,
  NUM_SERIALPORTSOURCE }
- enum spinSequencerModeEnums {
  SequencerMode_Off ,
  SequencerMode_On ,
  NUM_SEQUENCERMODE }
- enum spinSequencerConfigurationValidEnums {
  SequencerConfigurationValid_No ,
  SequencerConfigurationValid_Yes ,
  NUM_SEQUENCERCONFIGURATIONVALID }
- enum spinSequencerSetValidEnums {
  SequencerSetValid_No ,
  SequencerSetValid_Yes ,
  NUM_SEQUENCERSETVALID }
- enum spinSequencerTriggerActivationEnums {
  SequencerTriggerActivation_RisingEdge ,
  SequencerTriggerActivation_FallingEdge ,
  SequencerTriggerActivation_AnyEdge ,
  SequencerTriggerActivation_LevelHigh ,
  SequencerTriggerActivation_LevelLow ,
  NUM_SEQUENCERTRIGGERACTIVATION }
- enum spinSequencerConfigurationModeEnums {
  SequencerConfigurationMode_Off ,
  SequencerConfigurationMode_On ,
  NUM_SEQUENCERCONFIGURATIONMODE }

- enum spinSequencerTriggerSourceEnums {
  SequencerTriggerSource_Off ,
  SequencerTriggerSource_FrameStart ,
  NUM_SEQUENCERTRIGGERSOURCE }
- enum spinTransferQueueModeEnums {
  TransferQueueMode_FirstInFirstOut ,
  NUM_TRANSFERQUEUEMODE }
- enum spinTransferOperationModeEnums {
  TransferOperationMode_Continuous ,
  TransferOperationMode_MultiBlock ,
  NUM_TRANSFEROPERATIONMODE }
- enum spinTransferControlModeEnums {
  TransferControlMode_Basic ,
  TransferControlMode_Automatic ,
  TransferControlMode_UserControlled ,
  NUM_TRANSFERCONTROLMODE }
- enum spinChunkGainSelectorEnums {
  ChunkGainSelector_All ,
  ChunkGainSelector_Red ,
  ChunkGainSelector_Green ,
  ChunkGainSelector_Blue ,
  NUM_CHUNKGAINSELECTOR }
- enum spinChunkSelectorEnums {
  ChunkSelector_Image ,
  ChunkSelector_CRC ,
  ChunkSelector_FrameID ,
  ChunkSelector_OffsetX ,
  ChunkSelector_OffsetY ,
  ChunkSelector_Width ,
  ChunkSelector_Height ,
  ChunkSelector_ExposureTime ,
  ChunkSelector_Gain ,
  ChunkSelector_BlackLevel ,
  ChunkSelector_PixelFormat ,
  ChunkSelector_Timestamp ,
  ChunkSelector_SequencerSetActive ,
  ChunkSelector_SerialData ,
  ChunkSelector_ExposureEndLineStatusAll ,
  NUM_CHUNKSELECTOR }
- enum spinChunkBlackLevelSelectorEnums {
  ChunkBlackLevelSelector_All ,
  NUM_CHUNKBLACKLEVELSELECTOR }
- enum spinChunkPixelFormatEnums {
  ChunkPixelFormat_Mono8 ,
  ChunkPixelFormat_Mono12Packed ,
  ChunkPixelFormat_Mono16 ,
  ChunkPixelFormat_RGB8Packed ,
  ChunkPixelFormat_YUV422Packed ,
  ChunkPixelFormat_BayerGR8 ,
  ChunkPixelFormat_BayerRG8 ,
  ChunkPixelFormat_BayerGB8 ,
  ChunkPixelFormat_BayerBG8 ,
  ChunkPixelFormat_YCbCr601_422_8_CbYCrY ,
  NUM_CHUNKPIXELFORMAT }
- enum spinFileOperationStatusEnums {
  FileOperationStatus_Success ,
  FileOperationStatus_Failure ,

FileOperationStatus_Overflow ,
NUM_FILEOPERATIONSTATUS }
- enum spinFileOpenModeEnums {
FileOpenMode_Read ,
FileOpenMode_Write ,
FileOpenMode_ReadWrite ,
NUM_FILEOPENMODE }
- enum spinFileOperationSelectorEnums {
FileOperationSelector_Open ,
FileOperationSelector_Close ,
FileOperationSelector_Read ,
FileOperationSelector_Write ,
FileOperationSelector_Delete ,
NUM_FILEOPERATIONSELECTOR }
- enum spinFileSelectorEnums {
FileSelector_UserSetDefault ,
FileSelector_UserSet0 ,
FileSelector_UserSet1 ,
FileSelector_UserFile1 ,
FileSelector_SerialPort0 ,
NUM_FILESELECTOR }
- enum spinBinningSelectorEnums {
BinningSelector_All ,
BinningSelector_Sensor ,
BinningSelector_ISP ,
NUM_BINNINGSELECTOR }
- enum spinTestPatternGeneratorSelectorEnums {
TestPatternGeneratorSelector_Sensor ,
TestPatternGeneratorSelector_PipelineStart ,
NUM_TESTPATTERNGENERATORSELECTOR }
- enum spinCompressionSaturationPriorityEnums {
CompressionSaturationPriority_DropFrame ,
CompressionSaturationPriority_ReduceFrameRate ,
NUM_COMPRESSIONSATURATIONPRIORITY }
- enum spinTestPatternEnums {
TestPattern_Off ,
TestPattern_Increment ,
TestPattern_SensorTestPattern ,
NUM_TESTPATTERN }
- enum spinPixelColorFilterEnums {
PixelColorFilter_None ,
PixelColorFilter_BayerRG ,
PixelColorFilter_BayerGB ,
PixelColorFilter_BayerGR ,
PixelColorFilter_BayerBG ,
NUM_PIXELCOLORFILTER }
- enum spinAdcBitDepthEnums {
AdcBitDepth_Bit8 ,
AdcBitDepth_Bit10 ,
AdcBitDepth_Bit12 ,
AdcBitDepth_Bit14 ,
NUM_ADCBITDEPTH }
- enum spinDecimationHorizontalModeEnums {
DecimationHorizontalMode_Discard ,
NUM_DECIMATIONHORIZONTALMODE }
- enum spinBinningVerticalModeEnums {
BinningVerticalMode_Sum ,

BinningVerticalMode_Average ,
NUM_BINNINGVERTICALMODE }
- enum spinPixelSizeEnums {
PixelSize_Bpp1 ,
PixelSize_Bpp2 ,
PixelSize_Bpp4 ,
PixelSize_Bpp8 ,
PixelSize_Bpp10 ,
PixelSize_Bpp12 ,
PixelSize_Bpp14 ,
PixelSize_Bpp16 ,
PixelSize_Bpp20 ,
PixelSize_Bpp24 ,
PixelSize_Bpp30 ,
PixelSize_Bpp32 ,
PixelSize_Bpp36 ,
PixelSize_Bpp48 ,
PixelSize_Bpp64 ,
PixelSize_Bpp96 ,
NUM_PIXELSIZE }
- enum spinDecimationSelectorEnums {
DecimationSelector_All ,
DecimationSelector_Sensor ,
NUM_DECIMATIONSELECTOR }
- enum spinImageCompressionModeEnums {
ImageCompressionMode_Off ,
ImageCompressionMode_Lossless ,
NUM_IMAGECOMPRESSIONMODE }
- enum spinBinningHorizontalModeEnums {
BinningHorizontalMode_Sum ,
BinningHorizontalMode_Average ,
NUM_BINNINGHORIZONTALMODE }
- enum spinPixelFormatEnums {
PixelFormat_Mono8 ,
PixelFormat_Mono16 ,
PixelFormat_RGB8Packed ,
PixelFormat_BayerGR8 ,
PixelFormat_BayerRG8 ,
PixelFormat_BayerGB8 ,
PixelFormat_BayerBG8 ,
PixelFormat_BayerGR16 ,
PixelFormat_BayerRG16 ,
PixelFormat_BayerGB16 ,
PixelFormat_BayerBG16 ,
PixelFormat_Mono12Packed ,
PixelFormat_BayerGR12Packed ,
PixelFormat_BayerRG12Packed ,
PixelFormat_BayerGB12Packed ,
PixelFormat_BayerBG12Packed ,
PixelFormat_YUV411Packed ,
PixelFormat_YUV422Packed ,
PixelFormat_YUV444Packed ,
PixelFormat_Mono12p ,
PixelFormat_BayerGR12p ,
PixelFormat_BayerRG12p ,
PixelFormat_BayerGB12p ,
PixelFormat_BayerBG12p ,
PixelFormat_YCbCr8 ,

PixelFormat_YCbCr422_8 ,
PixelFormat_YCbCr411_8 ,
PixelFormat_BGR8 ,
PixelFormat_BGRa8 ,
PixelFormat_Mono10Packed ,
PixelFormat_BayerGR10Packed ,
PixelFormat_BayerRG10Packed ,
PixelFormat_BayerGB10Packed ,
PixelFormat_BayerBG10Packed ,
PixelFormat_Mono10p ,
PixelFormat_BayerGR10p ,
PixelFormat_BayerRG10p ,
PixelFormat_BayerGB10p ,
PixelFormat_BayerBG10p ,
PixelFormat_Mono1p ,
PixelFormat_Mono2p ,
PixelFormat_Mono4p ,
PixelFormat_Mono8s ,
PixelFormat_Mono10 ,
PixelFormat_Mono12 ,
PixelFormat_Mono14 ,
PixelFormat_Mono16s ,
PixelFormat_Mono32f ,
PixelFormat_BayerBG10 ,
PixelFormat_BayerBG12 ,
PixelFormat_BayerGB10 ,
PixelFormat_BayerGB12 ,
PixelFormat_BayerGR10 ,
PixelFormat_BayerGR12 ,
PixelFormat_BayerRG10 ,
PixelFormat_BayerRG12 ,
PixelFormat_RGBa8 ,
PixelFormat_RGBa10 ,
PixelFormat_RGBa10p ,
PixelFormat_RGBa12 ,
PixelFormat_RGBa12p ,
PixelFormat_RGBa14 ,
PixelFormat_RGBa16 ,
PixelFormat_RGB8 ,
PixelFormat_RGB8_Planar ,
PixelFormat_RGB10 ,
PixelFormat_RGB10_Planar ,
PixelFormat_RGB10p ,
PixelFormat_RGB10p32 ,
PixelFormat_RGB12 ,
PixelFormat_RGB12_Planar ,
PixelFormat_RGB12p ,
PixelFormat_RGB14 ,
PixelFormat_RGB16 ,
PixelFormat_RGB16s ,
PixelFormat_RGB32f ,
PixelFormat_RGB16_Planar ,
PixelFormat_RGB565p ,
PixelFormat_BGRa10 ,
PixelFormat_BGRa10p ,
PixelFormat_BGRa12 ,
PixelFormat_BGRa12p ,
PixelFormat_BGRa14 ,

PixelFormat_BGRa16 ,
PixelFormat_RGBa32f ,
PixelFormat_BGR10 ,
PixelFormat_BGR10p ,
PixelFormat_BGR12 ,
PixelFormat_BGR12p ,
PixelFormat_BGR14 ,
PixelFormat_BGR16 ,
PixelFormat_BGR565p ,
PixelFormat_R8 ,
PixelFormat_R10 ,
PixelFormat_R12 ,
PixelFormat_R16 ,
PixelFormat_G8 ,
PixelFormat_G10 ,
PixelFormat_G12 ,
PixelFormat_G16 ,
PixelFormat_B8 ,
PixelFormat_B10 ,
PixelFormat_B12 ,
PixelFormat_B16 ,
PixelFormat_Coord3D_ABC8 ,
PixelFormat_Coord3D_ABC8_Planar ,
PixelFormat_Coord3D_ABC10p ,
PixelFormat_Coord3D_ABC10p_Planar ,
PixelFormat_Coord3D_ABC12p ,
PixelFormat_Coord3D_ABC12p_Planar ,
PixelFormat_Coord3D_ABC16 ,
PixelFormat_Coord3D_ABC16_Planar ,
PixelFormat_Coord3D_ABC32f ,
PixelFormat_Coord3D_ABC32f_Planar ,
PixelFormat_Coord3D_AC8 ,
PixelFormat_Coord3D_AC8_Planar ,
PixelFormat_Coord3D_AC10p ,
PixelFormat_Coord3D_AC10p_Planar ,
PixelFormat_Coord3D_AC12p ,
PixelFormat_Coord3D_AC12p_Planar ,
PixelFormat_Coord3D_AC16 ,
PixelFormat_Coord3D_AC16_Planar ,
PixelFormat_Coord3D_AC32f ,
PixelFormat_Coord3D_AC32f_Planar ,
PixelFormat_Coord3D_A8 ,
PixelFormat_Coord3D_A10p ,
PixelFormat_Coord3D_A12p ,
PixelFormat_Coord3D_A16 ,
PixelFormat_Coord3D_A32f ,
PixelFormat_Coord3D_B8 ,
PixelFormat_Coord3D_B10p ,
PixelFormat_Coord3D_B12p ,
PixelFormat_Coord3D_B16 ,
PixelFormat_Coord3D_B32f ,
PixelFormat_Coord3D_C8 ,
PixelFormat_Coord3D_C10p ,
PixelFormat_Coord3D_C12p ,
PixelFormat_Coord3D_C16 ,
PixelFormat_Coord3D_C32f ,
PixelFormat_Confidence1 ,
PixelFormat_Confidence1p ,

PixelFormat_Confidence8 ,
PixelFormat_Confidence16 ,
PixelFormat_Confidence32f ,
PixelFormat_BiColorBGRG8 ,
PixelFormat_BiColorBGRG10 ,
PixelFormat_BiColorBGRG10p ,
PixelFormat_BiColorBGRG12 ,
PixelFormat_BiColorBGRG12p ,
PixelFormat_BiColorRGBG8 ,
PixelFormat_BiColorRGBG10 ,
PixelFormat_BiColorRGBG10p ,
PixelFormat_BiColorRGBG12 ,
PixelFormat_BiColorRGBG12p ,
PixelFormat_SCF1WBWG8 ,
PixelFormat_SCF1WBWG10 ,
PixelFormat_SCF1WBWG10p ,
PixelFormat_SCF1WBWG12 ,
PixelFormat_SCF1WBWG12p ,
PixelFormat_SCF1WBWG14 ,
PixelFormat_SCF1WBWG16 ,
PixelFormat_SCF1WGWB8 ,
PixelFormat_SCF1WGWB10 ,
PixelFormat_SCF1WGWB10p ,
PixelFormat_SCF1WGWB12 ,
PixelFormat_SCF1WGWB12p ,
PixelFormat_SCF1WGWB14 ,
PixelFormat_SCF1WGWB16 ,
PixelFormat_SCF1WGWR8 ,
PixelFormat_SCF1WGWR10 ,
PixelFormat_SCF1WGWR10p ,
PixelFormat_SCF1WGWR12 ,
PixelFormat_SCF1WGWR12p ,
PixelFormat_SCF1WGWR14 ,
PixelFormat_SCF1WGWR16 ,
PixelFormat_SCF1WRWG8 ,
PixelFormat_SCF1WRWG10 ,
PixelFormat_SCF1WRWG10p ,
PixelFormat_SCF1WRWG12 ,
PixelFormat_SCF1WRWG12p ,
PixelFormat_SCF1WRWG14 ,
PixelFormat_SCF1WRWG16 ,
PixelFormat_YCbCr8_CbYCr ,
PixelFormat_YCbCr10_CbYCr ,
PixelFormat_YCbCr10p_CbYCr ,
PixelFormat_YCbCr12_CbYCr ,
PixelFormat_YCbCr12p_CbYCr ,
PixelFormat_YCbCr411_8_CbYYCrYY ,
PixelFormat_YCbCr422_8_CbYCrY ,
PixelFormat_YCbCr422_10 ,
PixelFormat_YCbCr422_10_CbYCrY ,
PixelFormat_YCbCr422_10p ,
PixelFormat_YCbCr422_10p_CbYCrY ,
PixelFormat_YCbCr422_12 ,
PixelFormat_YCbCr422_12_CbYCrY ,
PixelFormat_YCbCr422_12p ,
PixelFormat_YCbCr422_12p_CbYCrY ,
PixelFormat_YCbCr601_8_CbYCr ,
PixelFormat_YCbCr601_10_CbYCr ,

PixelFormat_YCbCr601_10p_CbYCr ,
PixelFormat_YCbCr601_12_CbYCr ,
PixelFormat_YCbCr601_12p_CbYCr ,
PixelFormat_YCbCr601_411_8_CbYYCrYY ,
PixelFormat_YCbCr601_422_8 ,
PixelFormat_YCbCr601_422_8_CbYCrY ,
PixelFormat_YCbCr601_422_10 ,
PixelFormat_YCbCr601_422_10_CbYCrY ,
PixelFormat_YCbCr601_422_10p ,
PixelFormat_YCbCr601_422_10p_CbYCrY ,
PixelFormat_YCbCr601_422_12 ,
PixelFormat_YCbCr601_422_12_CbYCrY ,
PixelFormat_YCbCr601_422_12p ,
PixelFormat_YCbCr601_422_12p_CbYCrY ,
PixelFormat_YCbCr709_8_CbYCr ,
PixelFormat_YCbCr709_10_CbYCr ,
PixelFormat_YCbCr709_10p_CbYCr ,
PixelFormat_YCbCr709_12_CbYCr ,
PixelFormat_YCbCr709_12p_CbYCr ,
PixelFormat_YCbCr709_411_8_CbYYCrYY ,
PixelFormat_YCbCr709_422_8 ,
PixelFormat_YCbCr709_422_8_CbYCrY ,
PixelFormat_YCbCr709_422_10 ,
PixelFormat_YCbCr709_422_10_CbYCrY ,
PixelFormat_YCbCr709_422_10p ,
PixelFormat_YCbCr709_422_10p_CbYCrY ,
PixelFormat_YCbCr709_422_12 ,
PixelFormat_YCbCr709_422_12_CbYCrY ,
PixelFormat_YCbCr709_422_12p ,
PixelFormat_YCbCr709_422_12p_CbYCrY ,
PixelFormat_YUV8_UYV ,
PixelFormat_YUV411_8_UYYVYY ,
PixelFormat_YUV422_8 ,
PixelFormat_YUV422_8_UYVY ,
PixelFormat_Polarized8 ,
PixelFormat_Polarized10p ,
PixelFormat_Polarized12p ,
PixelFormat_Polarized16 ,
PixelFormat_BayerRGPolarized8 ,
PixelFormat_BayerRGPolarized10p ,
PixelFormat_BayerRGPolarized12p ,
PixelFormat_BayerRGPolarized16 ,
PixelFormat_LLCMono8 ,
PixelFormat_LLCBayerRG8 ,
PixelFormat_JPEGMono8 ,
PixelFormat_JPEGColor8 ,
PixelFormat_Raw16 ,
PixelFormat_Raw8 ,
PixelFormat_R12_Jpeg ,
PixelFormat_GR12_Jpeg ,
PixelFormat_GB12_Jpeg ,
PixelFormat_B12_Jpeg ,
PixelFormat_GR12 ,
PixelFormat_GB12 ,
UNKNOWN_PIXELFORMAT ,
NUM_PIXELFORMAT }

- enum spinDecimationVerticalModeEnums {
  DecimationVerticalMode_Discard ,

NUM_DECIMATIONVERTICALMODE }
- enum spinLineModeEnums {
  LineMode_Input ,
  LineMode_Output ,
  NUM_LINEMODE }
- enum spinLineSourceEnums {
  LineSource_Off ,
  LineSource_Line0 ,
  LineSource_Line1 ,
  LineSource_Line2 ,
  LineSource_Line3 ,
  LineSource_UserOutput0 ,
  LineSource_UserOutput1 ,
  LineSource_UserOutput2 ,
  LineSource_UserOutput3 ,
  LineSource_Counter0Active ,
  LineSource_Counter1Active ,
  LineSource_LogicBlock0 ,
  LineSource_LogicBlock1 ,
  LineSource_ExposureActive ,
  LineSource_FrameTriggerWait ,
  LineSource_SerialPort0 ,
  LineSource_PPSSignal ,
  LineSource_AllPixel ,
  LineSource_AnyPixel ,
  NUM_LINESOURCE }
- enum spinLineInputFilterSelectorEnums {
  LineInputFilterSelector_Deglitch ,
  LineInputFilterSelector_Debounce ,
  NUM_LINEINPUTFILTERSELECTOR }
- enum spinUserOutputSelectorEnums {
  UserOutputSelector_UserOutput0 ,
  UserOutputSelector_UserOutput1 ,
  UserOutputSelector_UserOutput2 ,
  UserOutputSelector_UserOutput3 ,
  NUM_USEROUTPUTSELECTOR }
- enum spinLineFormatEnums {
  LineFormat_NoConnect ,
  LineFormat_TriState ,
  LineFormat_TTL ,
  LineFormat_LVDS ,
  LineFormat_RS422 ,
  LineFormat_OptoCoupled ,
  LineFormat_OpenDrain ,
  NUM_LINEFORMAT }
- enum spinLineSelectorEnums {
  LineSelector_Line0 ,
  LineSelector_Line1 ,
  LineSelector_Line2 ,
  LineSelector_Line3 ,
  NUM_LINESELECTOR }
- enum spinExposureActiveModeEnums {
  ExposureActiveMode_Line1 ,
  ExposureActiveMode_AnyPixels ,
  ExposureActiveMode_AllPixels ,
  NUM_EXPOSUREACTIVEMODE }
- enum spinCounterTriggerActivationEnums {
  CounterTriggerActivation_LevelLow ,

CounterTriggerActivation_LevelHigh ,
CounterTriggerActivation_FallingEdge ,
CounterTriggerActivation_RisingEdge ,
CounterTriggerActivation_AnyEdge ,
NUM_COUNTERTRIGGERACTIVATION }

- enum spinCounterSelectorEnums {
  CounterSelector_Counter0 ,
  CounterSelector_Counter1 ,
  NUM_COUNTERSELECTOR }
- enum spinCounterStatusEnums {
  CounterStatus_CounterIdle ,
  CounterStatus_CounterTriggerWait ,
  CounterStatus_CounterActive ,
  CounterStatus_CounterCompleted ,
  CounterStatus_CounterOverflow ,
  NUM_COUNTERSTATUS }
- enum spinCounterTriggerSourceEnums {
  CounterTriggerSource_Off ,
  CounterTriggerSource_Line0 ,
  CounterTriggerSource_Line1 ,
  CounterTriggerSource_Line2 ,
  CounterTriggerSource_Line3 ,
  CounterTriggerSource_UserOutput0 ,
  CounterTriggerSource_UserOutput1 ,
  CounterTriggerSource_UserOutput2 ,
  CounterTriggerSource_UserOutput3 ,
  CounterTriggerSource_Counter0Start ,
  CounterTriggerSource_Counter1Start ,
  CounterTriggerSource_Counter0End ,
  CounterTriggerSource_Counter1End ,
  CounterTriggerSource_LogicBlock0 ,
  CounterTriggerSource_LogicBlock1 ,
  CounterTriggerSource_ExposureStart ,
  CounterTriggerSource_ExposureEnd ,
  CounterTriggerSource_FrameTriggerWait ,
  NUM_COUNTERTRIGGERSOURCE }
- enum spinCounterResetSourceEnums {
  CounterResetSource_Off ,
  CounterResetSource_Line0 ,
  CounterResetSource_Line1 ,
  CounterResetSource_Line2 ,
  CounterResetSource_Line3 ,
  CounterResetSource_UserOutput0 ,
  CounterResetSource_UserOutput1 ,
  CounterResetSource_UserOutput2 ,
  CounterResetSource_UserOutput3 ,
  CounterResetSource_Counter0Start ,
  CounterResetSource_Counter1Start ,
  CounterResetSource_Counter0End ,
  CounterResetSource_Counter1End ,
  CounterResetSource_LogicBlock0 ,
  CounterResetSource_LogicBlock1 ,
  CounterResetSource_ExposureStart ,
  CounterResetSource_ExposureEnd ,
  CounterResetSource_FrameTriggerWait ,
  NUM_COUNTERRESETSOURCE }
- enum spinCounterEventSourceEnums {
  CounterEventSource_Off ,

CounterEventSource_MHzTick ,
CounterEventSource_Line0 ,
CounterEventSource_Line1 ,
CounterEventSource_Line2 ,
CounterEventSource_Line3 ,
CounterEventSource_UserOutput0 ,
CounterEventSource_UserOutput1 ,
CounterEventSource_UserOutput2 ,
CounterEventSource_UserOutput3 ,
CounterEventSource_Counter0Start ,
CounterEventSource_Counter1Start ,
CounterEventSource_Counter0End ,
CounterEventSource_Counter1End ,
CounterEventSource_LogicBlock0 ,
CounterEventSource_LogicBlock1 ,
CounterEventSource_ExposureStart ,
CounterEventSource_ExposureEnd ,
CounterEventSource_FrameTriggerWait ,
NUM_COUNTEREVENTSOURCE }

- enum spinCounterEventActivationEnums {
  CounterEventActivation_LevelLow ,
  CounterEventActivation_LevelHigh ,
  CounterEventActivation_FallingEdge ,
  CounterEventActivation_RisingEdge ,
  CounterEventActivation_AnyEdge ,
  NUM_COUNTEREVENTACTIVATION }
- enum spinCounterResetActivationEnums {
  CounterResetActivation_LevelLow ,
  CounterResetActivation_LevelHigh ,
  CounterResetActivation_FallingEdge ,
  CounterResetActivation_RisingEdge ,
  CounterResetActivation_AnyEdge ,
  NUM_COUNTERRESETACTIVATION }
- enum spinDeviceTypeEnums {
  DeviceType_Transmitter ,
  DeviceType_Receiver ,
  DeviceType_Transceiver ,
  DeviceType_Peripheral ,
  NUM_DEVICETYPE }
- enum spinDeviceConnectionStatusEnums {
  DeviceConnectionStatus_Active ,
  DeviceConnectionStatus_Inactive ,
  NUM_DEVICECONNECTIONSTATUS }
- enum spinDeviceLinkThroughputLimitModeEnums {
  DeviceLinkThroughputLimitMode_On ,
  DeviceLinkThroughputLimitMode_Off ,
  NUM_DEVICELINKTHROUGHPUTLIMITMODE }
- enum spinDeviceLinkHeartbeatModeEnums {
  DeviceLinkHeartbeatMode_On ,
  DeviceLinkHeartbeatMode_Off ,
  NUM_DEVICELINKHEARTBEATMODE }
- enum spinDeviceStreamChannelTypeEnums {
  DeviceStreamChannelType_Transmitter ,
  DeviceStreamChannelType_Receiver ,
  NUM_DEVICESTREAMCHANNELTYPE }
- enum spinDeviceStreamChannelEndiannessEnums {
  DeviceStreamChannelEndianness_Big ,

DeviceStreamChannelEndianness_Little ,
NUM_DEVICESTREAMCHANNELENDIANNESS }
- enum spinDeviceClockSelectorEnums {
DeviceClockSelector_Sensor ,
DeviceClockSelector_SensorDigitization ,
DeviceClockSelector_CameraLink ,
NUM_DEVICECLOCKSELECTOR }
- enum spinDeviceSerialPortSelectorEnums {
DeviceSerialPortSelector_CameraLink ,
NUM_DEVICESERIALPORTSELECTOR }
- enum spinDeviceSerialPortBaudRateEnums {
DeviceSerialPortBaudRate_Baud9600 ,
DeviceSerialPortBaudRate_Baud19200 ,
DeviceSerialPortBaudRate_Baud38400 ,
DeviceSerialPortBaudRate_Baud57600 ,
DeviceSerialPortBaudRate_Baud115200 ,
DeviceSerialPortBaudRate_Baud230400 ,
DeviceSerialPortBaudRate_Baud460800 ,
DeviceSerialPortBaudRate_Baud921600 ,
NUM_DEVICESERIALPORTBAUDRATE }
- enum spinSensorTapsEnums {
SensorTaps_One ,
SensorTaps_Two ,
SensorTaps_Three ,
SensorTaps_Four ,
SensorTaps_Eight ,
SensorTaps_Ten ,
NUM_SENSORTAPS }
- enum spinSensorDigitizationTapsEnums {
SensorDigitizationTaps_One ,
SensorDigitizationTaps_Two ,
SensorDigitizationTaps_Three ,
SensorDigitizationTaps_Four ,
SensorDigitizationTaps_Eight ,
SensorDigitizationTaps_Ten ,
NUM_SENSORDIGITIZATIONTAPS }
- enum spinRegionSelectorEnums {
RegionSelector_Region0 ,
RegionSelector_Region1 ,
RegionSelector_Region2 ,
RegionSelector_All ,
NUM_REGIONSELECTOR }
- enum spinRegionModeEnums {
RegionMode_Off ,
RegionMode_On ,
NUM_REGIONMODE }
- enum spinRegionDestinationEnums {
RegionDestination_Stream0 ,
RegionDestination_Stream1 ,
RegionDestination_Stream2 ,
NUM_REGIONDESTINATION }
- enum spinImageComponentSelectorEnums {
ImageComponentSelector_Intensity ,
ImageComponentSelector_Color ,
ImageComponentSelector_Infrared ,
ImageComponentSelector_Ultraviolet ,
ImageComponentSelector_Range ,
ImageComponentSelector_Disparity ,

ImageComponentSelector_Confidence ,
ImageComponentSelector_Scatter ,
NUM_IMAGECOMPONENTSELECTOR }
- enum spinPixelFormatInfoSelectorEnums {
PixelFormatInfoSelector_Mono1p ,
PixelFormatInfoSelector_Mono2p ,
PixelFormatInfoSelector_Mono4p ,
PixelFormatInfoSelector_Mono8 ,
PixelFormatInfoSelector_Mono8s ,
PixelFormatInfoSelector_Mono10 ,
PixelFormatInfoSelector_Mono10p ,
PixelFormatInfoSelector_Mono12 ,
PixelFormatInfoSelector_Mono12p ,
PixelFormatInfoSelector_Mono14 ,
PixelFormatInfoSelector_Mono16 ,
PixelFormatInfoSelector_Mono16s ,
PixelFormatInfoSelector_Mono32f ,
PixelFormatInfoSelector_BayerBG8 ,
PixelFormatInfoSelector_BayerBG10 ,
PixelFormatInfoSelector_BayerBG10p ,
PixelFormatInfoSelector_BayerBG12 ,
PixelFormatInfoSelector_BayerBG12p ,
PixelFormatInfoSelector_BayerBG16 ,
PixelFormatInfoSelector_BayerGB8 ,
PixelFormatInfoSelector_BayerGB10 ,
PixelFormatInfoSelector_BayerGB10p ,
PixelFormatInfoSelector_BayerGB12 ,
PixelFormatInfoSelector_BayerGB12p ,
PixelFormatInfoSelector_BayerGB16 ,
PixelFormatInfoSelector_BayerGR8 ,
PixelFormatInfoSelector_BayerGR10 ,
PixelFormatInfoSelector_BayerGR10p ,
PixelFormatInfoSelector_BayerGR12 ,
PixelFormatInfoSelector_BayerGR12p ,
PixelFormatInfoSelector_BayerGR16 ,
PixelFormatInfoSelector_BayerRG8 ,
PixelFormatInfoSelector_BayerRG10 ,
PixelFormatInfoSelector_BayerRG10p ,
PixelFormatInfoSelector_BayerRG12 ,
PixelFormatInfoSelector_BayerRG12p ,
PixelFormatInfoSelector_BayerRG16 ,
PixelFormatInfoSelector_RGBa8 ,
PixelFormatInfoSelector_RGBa10 ,
PixelFormatInfoSelector_RGBa10p ,
PixelFormatInfoSelector_RGBa12 ,
PixelFormatInfoSelector_RGBa12p ,
PixelFormatInfoSelector_RGBa14 ,
PixelFormatInfoSelector_RGBa16 ,
PixelFormatInfoSelector_RGB8 ,
PixelFormatInfoSelector_RGB8_Planar ,
PixelFormatInfoSelector_RGB10 ,
PixelFormatInfoSelector_RGB10_Planar ,
PixelFormatInfoSelector_RGB10p ,
PixelFormatInfoSelector_RGB10p32 ,
PixelFormatInfoSelector_RGB12 ,
PixelFormatInfoSelector_RGB12_Planar ,
PixelFormatInfoSelector_RGB12p ,
PixelFormatInfoSelector_RGB14 ,

PixelFormatInfoSelector_RGB16 ,
PixelFormatInfoSelector_RGB16s ,
PixelFormatInfoSelector_RGB32f ,
PixelFormatInfoSelector_RGB16_Planar ,
PixelFormatInfoSelector_RGB565p ,
PixelFormatInfoSelector_BGRa8 ,
PixelFormatInfoSelector_BGRa10 ,
PixelFormatInfoSelector_BGRa10p ,
PixelFormatInfoSelector_BGRa12 ,
PixelFormatInfoSelector_BGRa12p ,
PixelFormatInfoSelector_BGRa14 ,
PixelFormatInfoSelector_BGRa16 ,
PixelFormatInfoSelector_RGBa32f ,
PixelFormatInfoSelector_BGR8 ,
PixelFormatInfoSelector_BGR10 ,
PixelFormatInfoSelector_BGR10p ,
PixelFormatInfoSelector_BGR12 ,
PixelFormatInfoSelector_BGR12p ,
PixelFormatInfoSelector_BGR14 ,
PixelFormatInfoSelector_BGR16 ,
PixelFormatInfoSelector_BGR565p ,
PixelFormatInfoSelector_R8 ,
PixelFormatInfoSelector_R10 ,
PixelFormatInfoSelector_R12 ,
PixelFormatInfoSelector_R16 ,
PixelFormatInfoSelector_G8 ,
PixelFormatInfoSelector_G10 ,
PixelFormatInfoSelector_G12 ,
PixelFormatInfoSelector_G16 ,
PixelFormatInfoSelector_B8 ,
PixelFormatInfoSelector_B10 ,
PixelFormatInfoSelector_B12 ,
PixelFormatInfoSelector_B16 ,
PixelFormatInfoSelector_Coord3D_ABC8 ,
PixelFormatInfoSelector_Coord3D_ABC8_Planar ,
PixelFormatInfoSelector_Coord3D_ABC10p ,
PixelFormatInfoSelector_Coord3D_ABC10p_Planar ,
PixelFormatInfoSelector_Coord3D_ABC12p ,
PixelFormatInfoSelector_Coord3D_ABC12p_Planar ,
PixelFormatInfoSelector_Coord3D_ABC16 ,
PixelFormatInfoSelector_Coord3D_ABC16_Planar ,
PixelFormatInfoSelector_Coord3D_ABC32f ,
PixelFormatInfoSelector_Coord3D_ABC32f_Planar ,
PixelFormatInfoSelector_Coord3D_AC8 ,
PixelFormatInfoSelector_Coord3D_AC8_Planar ,
PixelFormatInfoSelector_Coord3D_AC10p ,
PixelFormatInfoSelector_Coord3D_AC10p_Planar ,
PixelFormatInfoSelector_Coord3D_AC12p ,
PixelFormatInfoSelector_Coord3D_AC12p_Planar ,
PixelFormatInfoSelector_Coord3D_AC16 ,
PixelFormatInfoSelector_Coord3D_AC16_Planar ,
PixelFormatInfoSelector_Coord3D_AC32f ,
PixelFormatInfoSelector_Coord3D_AC32f_Planar ,
PixelFormatInfoSelector_Coord3D_A8 ,
PixelFormatInfoSelector_Coord3D_A10p ,
PixelFormatInfoSelector_Coord3D_A12p ,
PixelFormatInfoSelector_Coord3D_A16 ,
PixelFormatInfoSelector_Coord3D_A32f ,

PixelFormatInfoSelector_Coord3D_B8 ,
PixelFormatInfoSelector_Coord3D_B10p ,
PixelFormatInfoSelector_Coord3D_B12p ,
PixelFormatInfoSelector_Coord3D_B16 ,
PixelFormatInfoSelector_Coord3D_B32f ,
PixelFormatInfoSelector_Coord3D_C8 ,
PixelFormatInfoSelector_Coord3D_C10p ,
PixelFormatInfoSelector_Coord3D_C12p ,
PixelFormatInfoSelector_Coord3D_C16 ,
PixelFormatInfoSelector_Coord3D_C32f ,
PixelFormatInfoSelector_Confidence1 ,
PixelFormatInfoSelector_Confidence1p ,
PixelFormatInfoSelector_Confidence8 ,
PixelFormatInfoSelector_Confidence16 ,
PixelFormatInfoSelector_Confidence32f ,
PixelFormatInfoSelector_BiColorBGRG8 ,
PixelFormatInfoSelector_BiColorBGRG10 ,
PixelFormatInfoSelector_BiColorBGRG10p ,
PixelFormatInfoSelector_BiColorBGRG12 ,
PixelFormatInfoSelector_BiColorBGRG12p ,
PixelFormatInfoSelector_BiColorRGBG8 ,
PixelFormatInfoSelector_BiColorRGBG10 ,
PixelFormatInfoSelector_BiColorRGBG10p ,
PixelFormatInfoSelector_BiColorRGBG12 ,
PixelFormatInfoSelector_BiColorRGBG12p ,
PixelFormatInfoSelector_SCF1WBWG8 ,
PixelFormatInfoSelector_SCF1WBWG10 ,
PixelFormatInfoSelector_SCF1WBWG10p ,
PixelFormatInfoSelector_SCF1WBWG12 ,
PixelFormatInfoSelector_SCF1WBWG12p ,
PixelFormatInfoSelector_SCF1WBWG14 ,
PixelFormatInfoSelector_SCF1WBWG16 ,
PixelFormatInfoSelector_SCF1WGWB8 ,
PixelFormatInfoSelector_SCF1WGWB10 ,
PixelFormatInfoSelector_SCF1WGWB10p ,
PixelFormatInfoSelector_SCF1WGWB12 ,
PixelFormatInfoSelector_SCF1WGWB12p ,
PixelFormatInfoSelector_SCF1WGWB14 ,
PixelFormatInfoSelector_SCF1WGWB16 ,
PixelFormatInfoSelector_SCF1WGWR8 ,
PixelFormatInfoSelector_SCF1WGWR10 ,
PixelFormatInfoSelector_SCF1WGWR10p ,
PixelFormatInfoSelector_SCF1WGWR12 ,
PixelFormatInfoSelector_SCF1WGWR12p ,
PixelFormatInfoSelector_SCF1WGWR14 ,
PixelFormatInfoSelector_SCF1WGWR16 ,
PixelFormatInfoSelector_SCF1WRWG8 ,
PixelFormatInfoSelector_SCF1WRWG10 ,
PixelFormatInfoSelector_SCF1WRWG10p ,
PixelFormatInfoSelector_SCF1WRWG12 ,
PixelFormatInfoSelector_SCF1WRWG12p ,
PixelFormatInfoSelector_SCF1WRWG14 ,
PixelFormatInfoSelector_SCF1WRWG16 ,
PixelFormatInfoSelector_YCbCr8 ,
PixelFormatInfoSelector_YCbCr8_CbYCr ,
PixelFormatInfoSelector_YCbCr10_CbYCr ,
PixelFormatInfoSelector_YCbCr10p_CbYCr ,
PixelFormatInfoSelector_YCbCr12_CbYCr ,

PixelFormatInfoSelector_YCbCr12p_CbYCr ,
PixelFormatInfoSelector_YCbCr411_8 ,
PixelFormatInfoSelector_YCbCr411_8_CbYYCrYY ,
PixelFormatInfoSelector_YCbCr422_8 ,
PixelFormatInfoSelector_YCbCr422_8_CbYCrY ,
PixelFormatInfoSelector_YCbCr422_10 ,
PixelFormatInfoSelector_YCbCr422_10_CbYCrY ,
PixelFormatInfoSelector_YCbCr422_10p ,
PixelFormatInfoSelector_YCbCr422_10p_CbYCrY ,
PixelFormatInfoSelector_YCbCr422_12 ,
PixelFormatInfoSelector_YCbCr422_12_CbYCrY ,
PixelFormatInfoSelector_YCbCr422_12p ,
PixelFormatInfoSelector_YCbCr422_12p_CbYCrY ,
PixelFormatInfoSelector_YCbCr601_8_CbYCr ,
PixelFormatInfoSelector_YCbCr601_10_CbYCr ,
PixelFormatInfoSelector_YCbCr601_10p_CbYCr ,
PixelFormatInfoSelector_YCbCr601_12_CbYCr ,
PixelFormatInfoSelector_YCbCr601_12p_CbYCr ,
PixelFormatInfoSelector_YCbCr601_411_8_CbYYCrYY ,
PixelFormatInfoSelector_YCbCr601_422_8 ,
PixelFormatInfoSelector_YCbCr601_422_8_CbYCrY ,
PixelFormatInfoSelector_YCbCr601_422_10 ,
PixelFormatInfoSelector_YCbCr601_422_10_CbYCrY ,
PixelFormatInfoSelector_YCbCr601_422_10p ,
PixelFormatInfoSelector_YCbCr601_422_10p_CbYCrY ,
PixelFormatInfoSelector_YCbCr601_422_12 ,
PixelFormatInfoSelector_YCbCr601_422_12_CbYCrY ,
PixelFormatInfoSelector_YCbCr601_422_12p ,
PixelFormatInfoSelector_YCbCr601_422_12p_CbYCrY ,
PixelFormatInfoSelector_YCbCr709_8_CbYCr ,
PixelFormatInfoSelector_YCbCr709_10_CbYCr ,
PixelFormatInfoSelector_YCbCr709_10p_CbYCr ,
PixelFormatInfoSelector_YCbCr709_12_CbYCr ,
PixelFormatInfoSelector_YCbCr709_12p_CbYCr ,
PixelFormatInfoSelector_YCbCr709_411_8_CbYYCrYY ,
PixelFormatInfoSelector_YCbCr709_422_8 ,
PixelFormatInfoSelector_YCbCr709_422_8_CbYCrY ,
PixelFormatInfoSelector_YCbCr709_422_10 ,
PixelFormatInfoSelector_YCbCr709_422_10_CbYCrY ,
PixelFormatInfoSelector_YCbCr709_422_10p ,
PixelFormatInfoSelector_YCbCr709_422_10p_CbYCrY ,
PixelFormatInfoSelector_YCbCr709_422_12 ,
PixelFormatInfoSelector_YCbCr709_422_12_CbYCrY ,
PixelFormatInfoSelector_YCbCr709_422_12p ,
PixelFormatInfoSelector_YCbCr709_422_12p_CbYCrY ,
PixelFormatInfoSelector_YUV8_UYV ,
PixelFormatInfoSelector_YUV411_8_UYYVYY ,
PixelFormatInfoSelector_YUV422_8 ,
PixelFormatInfoSelector_YUV422_8_UYVY ,
PixelFormatInfoSelector_Polarized8 ,
PixelFormatInfoSelector_Polarized10p ,
PixelFormatInfoSelector_Polarized12p ,
PixelFormatInfoSelector_Polarized16 ,
PixelFormatInfoSelector_BayerRGPolarized8 ,
PixelFormatInfoSelector_BayerRGPolarized10p ,
PixelFormatInfoSelector_BayerRGPolarized12p ,
PixelFormatInfoSelector_BayerRGPolarized16 ,
PixelFormatInfoSelector_LLCMono8 ,

PixelFormatInfoSelector_LLCBayerRG8 ,
PixelFormatInfoSelector_JPEGMono8 ,
PixelFormatInfoSelector_JPEGColor8 ,
NUM_PIXELFORMATINFOSELECTOR }
- enum spinDeinterlacingEnums {
Deinterlacing_Off ,
Deinterlacing_LineDuplication ,
Deinterlacing_Weave ,
NUM_DEINTERLACING }
- enum spinImageCompressionRateOptionEnums {
ImageCompressionRateOption_FixBitrate ,
ImageCompressionRateOption_FixQuality ,
NUM_IMAGECOMPRESSIONRATEOPTION }
- enum spinImageCompressionJPEGFormatOptionEnums {
ImageCompressionJPEGFormatOption_Lossless ,
ImageCompressionJPEGFormatOption_BaselineStandard ,
ImageCompressionJPEGFormatOption_BaselineOptimized ,
ImageCompressionJPEGFormatOption_Progressive ,
NUM_IMAGECOMPRESSIONJPEGFORMATOPTION }
- enum spinAcquisitionStatusSelectorEnums {
AcquisitionStatusSelector_AcquisitionTriggerWait ,
AcquisitionStatusSelector_AcquisitionActive ,
AcquisitionStatusSelector_AcquisitionTransfer ,
AcquisitionStatusSelector_FrameTriggerWait ,
AcquisitionStatusSelector_FrameActive ,
AcquisitionStatusSelector_ExposureActive ,
NUM_ACQUISITIONSTATUSSELECTOR }
- enum spinExposureTimeModeEnums {
ExposureTimeMode_Common ,
ExposureTimeMode_Individual ,
NUM_EXPOSURETIMEMODE }
- enum spinExposureTimeSelectorEnums {
ExposureTimeSelector_Common ,
ExposureTimeSelector_Red ,
ExposureTimeSelector_Green ,
ExposureTimeSelector_Blue ,
ExposureTimeSelector_Cyan ,
ExposureTimeSelector_Magenta ,
ExposureTimeSelector_Yellow ,
ExposureTimeSelector_Infrared ,
ExposureTimeSelector_Ultraviolet ,
ExposureTimeSelector_Stage1 ,
ExposureTimeSelector_Stage2 ,
NUM_EXPOSURETIMESELECTOR }
- enum spinGainAutoBalanceEnums {
GainAutoBalance_Off ,
GainAutoBalance_Once ,
GainAutoBalance_Continuous ,
NUM_GAINAUTOBALANCE }
- enum spinBlackLevelAutoEnums {
BlackLevelAuto_Off ,
BlackLevelAuto_Once ,
BlackLevelAuto_Continuous ,
NUM_BLACKLEVELAUTO }
- enum spinBlackLevelAutoBalanceEnums {
BlackLevelAutoBalance_Off ,
BlackLevelAutoBalance_Once ,

BlackLevelAutoBalance_Continuous ,
NUM_BLACKLEVELAUTOBALANCE }
- enum spinWhiteClipSelectorEnums {
WhiteClipSelector_All ,
WhiteClipSelector_Red ,
WhiteClipSelector_Green ,
WhiteClipSelector_Blue ,
WhiteClipSelector_Y ,
WhiteClipSelector_U ,
WhiteClipSelector_V ,
WhiteClipSelector_Tap1 ,
WhiteClipSelector_Tap2 ,
NUM_WHITECLIPSELECTOR }
- enum spinTimerSelectorEnums {
TimerSelector_Timer0 ,
TimerSelector_Timer1 ,
TimerSelector_Timer2 ,
NUM_TIMERSELECTOR }
- enum spinTimerStatusEnums {
TimerStatus_TimerIdle ,
TimerStatus_TimerTriggerWait ,
TimerStatus_TimerActive ,
TimerStatus_TimerCompleted ,
NUM_TIMERSTATUS }
- enum spinTimerTriggerSourceEnums {
TimerTriggerSource_Off ,
TimerTriggerSource_AcquisitionTrigger ,
TimerTriggerSource_AcquisitionStart ,
TimerTriggerSource_AcquisitionEnd ,
TimerTriggerSource_FrameTrigger ,
TimerTriggerSource_FrameStart ,
TimerTriggerSource_FrameEnd ,
TimerTriggerSource_FrameBurstStart ,
TimerTriggerSource_FrameBurstEnd ,
TimerTriggerSource_LineTrigger ,
TimerTriggerSource_LineStart ,
TimerTriggerSource_LineEnd ,
TimerTriggerSource_ExposureStart ,
TimerTriggerSource_ExposureEnd ,
TimerTriggerSource_Line0 ,
TimerTriggerSource_Line1 ,
TimerTriggerSource_Line2 ,
TimerTriggerSource_UserOutput0 ,
TimerTriggerSource_UserOutput1 ,
TimerTriggerSource_UserOutput2 ,
TimerTriggerSource_Counter0Start ,
TimerTriggerSource_Counter1Start ,
TimerTriggerSource_Counter2Start ,
TimerTriggerSource_Counter0End ,
TimerTriggerSource_Counter1End ,
TimerTriggerSource_Counter2End ,
TimerTriggerSource_Timer0Start ,
TimerTriggerSource_Timer1Start ,
TimerTriggerSource_Timer2Start ,
TimerTriggerSource_Timer0End ,
TimerTriggerSource_Timer1End ,
TimerTriggerSource_Timer2End ,
TimerTriggerSource_Encoder0 ,

TimerTriggerSource_Encoder1 ,
TimerTriggerSource_Encoder2 ,
TimerTriggerSource_SoftwareSignal0 ,
TimerTriggerSource_SoftwareSignal1 ,
TimerTriggerSource_SoftwareSignal2 ,
TimerTriggerSource_Action0 ,
TimerTriggerSource_Action1 ,
TimerTriggerSource_Action2 ,
TimerTriggerSource_LinkTrigger0 ,
TimerTriggerSource_LinkTrigger1 ,
TimerTriggerSource_LinkTrigger2 ,
NUM_TIMERTRIGGERSOURCE }

- enum spinTimerTriggerActivationEnums {
  TimerTriggerActivation_RisingEdge ,
  TimerTriggerActivation_FallingEdge ,
  TimerTriggerActivation_AnyEdge ,
  TimerTriggerActivation_LevelHigh ,
  TimerTriggerActivation_LevelLow ,
  NUM_TIMERTRIGGERACTIVATION }
- enum spinEncoderSelectorEnums {
  EncoderSelector_Encoder0 ,
  EncoderSelector_Encoder1 ,
  EncoderSelector_Encoder2 ,
  NUM_ENCODERSELECTOR }
- enum spinEncoderSourceAEnums {
  EncoderSourceA_Off ,
  EncoderSourceA_Line0 ,
  EncoderSourceA_Line1 ,
  EncoderSourceA_Line2 ,
  NUM_ENCODERSOURCEA }
- enum spinEncoderSourceBEnums {
  EncoderSourceB_Off ,
  EncoderSourceB_Line0 ,
  EncoderSourceB_Line1 ,
  EncoderSourceB_Line2 ,
  NUM_ENCODERSOURCEB }
- enum spinEncoderModeEnums {
  EncoderMode_FourPhase ,
  EncoderMode_HighResolution ,
  NUM_ENCODERMODE }
- enum spinEncoderOutputModeEnums {
  EncoderOutputMode_Off ,
  EncoderOutputMode_PositionUp ,
  EncoderOutputMode_PositionDown ,
  EncoderOutputMode_DirectionUp ,
  EncoderOutputMode_DirectionDown ,
  EncoderOutputMode_Motion ,
  NUM_ENCODEROUTPUTMODE }
- enum spinEncoderStatusEnums {
  EncoderStatus_EncoderUp ,
  EncoderStatus_EncoderDown ,
  EncoderStatus_EncoderIdle ,
  EncoderStatus_EncoderStatic ,
  NUM_ENCODERSTATUS }
- enum spinEncoderResetSourceEnums {
  EncoderResetSource_Off ,
  EncoderResetSource_AcquisitionTrigger ,
  EncoderResetSource_AcquisitionStart ,

EncoderResetSource_AcquisitionEnd ,
EncoderResetSource_FrameTrigger ,
EncoderResetSource_FrameStart ,
EncoderResetSource_FrameEnd ,
EncoderResetSource_ExposureStart ,
EncoderResetSource_ExposureEnd ,
EncoderResetSource_Line0 ,
EncoderResetSource_Line1 ,
EncoderResetSource_Line2 ,
EncoderResetSource_Counter0Start ,
EncoderResetSource_Counter1Start ,
EncoderResetSource_Counter2Start ,
EncoderResetSource_Counter0End ,
EncoderResetSource_Counter1End ,
EncoderResetSource_Counter2End ,
EncoderResetSource_Timer0Start ,
EncoderResetSource_Timer1Start ,
EncoderResetSource_Timer2Start ,
EncoderResetSource_Timer0End ,
EncoderResetSource_Timer1End ,
EncoderResetSource_Timer2End ,
EncoderResetSource_UserOutput0 ,
EncoderResetSource_UserOutput1 ,
EncoderResetSource_UserOutput2 ,
EncoderResetSource_SoftwareSignal0 ,
EncoderResetSource_SoftwareSignal1 ,
EncoderResetSource_SoftwareSignal2 ,
EncoderResetSource_Action0 ,
EncoderResetSource_Action1 ,
EncoderResetSource_Action2 ,
EncoderResetSource_LinkTrigger0 ,
EncoderResetSource_LinkTrigger1 ,
EncoderResetSource_LinkTrigger2 ,
NUM_ENCODERRESETSOURCE }

- enum spinEncoderResetActivationEnums {
  EncoderResetActivation_RisingEdge ,
  EncoderResetActivation_FallingEdge ,
  EncoderResetActivation_AnyEdge ,
  EncoderResetActivation_LevelHigh ,
  EncoderResetActivation_LevelLow ,
  NUM_ENCODERRESETACTIVATION }
- enum spinSoftwareSignalSelectorEnums {
  SoftwareSignalSelector_SoftwareSignal0 ,
  SoftwareSignalSelector_SoftwareSignal1 ,
  SoftwareSignalSelector_SoftwareSignal2 ,
  NUM_SOFTWARESIGNALSELECTOR }
- enum spinActionUnconditionalModeEnums {
  ActionUnconditionalMode_Off ,
  ActionUnconditionalMode_On ,
  NUM_ACTIONUNCONDITIONALMODE }
- enum spinSourceSelectorEnums {
  SourceSelector_Source0 ,
  SourceSelector_Source1 ,
  SourceSelector_Source2 ,
  SourceSelector_All ,
  NUM_SOURCESELECTOR }
- enum spinTransferSelectorEnums {
  TransferSelector_Stream0 ,

TransferSelector_Stream1 ,
TransferSelector_Stream2 ,
TransferSelector_All ,
NUM_TRANSFERSELECTOR }
- enum spinTransferTriggerSelectorEnums {
TransferTriggerSelector_TransferStart ,
TransferTriggerSelector_TransferStop ,
TransferTriggerSelector_TransferAbort ,
TransferTriggerSelector_TransferPause ,
TransferTriggerSelector_TransferResume ,
TransferTriggerSelector_TransferActive ,
TransferTriggerSelector_TransferBurstStart ,
TransferTriggerSelector_TransferBurstStop ,
NUM_TRANSFERTRIGGERSELECTOR }
- enum spinTransferTriggerModeEnums {
TransferTriggerMode_Off ,
TransferTriggerMode_On ,
NUM_TRANSFERTRIGGERMODE }
- enum spinTransferTriggerSourceEnums {
TransferTriggerSource_Line0 ,
TransferTriggerSource_Line1 ,
TransferTriggerSource_Line2 ,
TransferTriggerSource_Counter0Start ,
TransferTriggerSource_Counter1Start ,
TransferTriggerSource_Counter2Start ,
TransferTriggerSource_Counter0End ,
TransferTriggerSource_Counter1End ,
TransferTriggerSource_Counter2End ,
TransferTriggerSource_Timer0Start ,
TransferTriggerSource_Timer1Start ,
TransferTriggerSource_Timer2Start ,
TransferTriggerSource_Timer0End ,
TransferTriggerSource_Timer1End ,
TransferTriggerSource_Timer2End ,
TransferTriggerSource_SoftwareSignal0 ,
TransferTriggerSource_SoftwareSignal1 ,
TransferTriggerSource_SoftwareSignal2 ,
TransferTriggerSource_Action0 ,
TransferTriggerSource_Action1 ,
TransferTriggerSource_Action2 ,
NUM_TRANSFERTRIGGERSOURCE }
- enum spinTransferTriggerActivationEnums {
TransferTriggerActivation_RisingEdge ,
TransferTriggerActivation_FallingEdge ,
TransferTriggerActivation_AnyEdge ,
TransferTriggerActivation_LevelHigh ,
TransferTriggerActivation_LevelLow ,
NUM_TRANSFERTRIGGERACTIVATION }
- enum spinTransferStatusSelectorEnums {
TransferStatusSelector_Streaming ,
TransferStatusSelector_Paused ,
TransferStatusSelector_Stopping ,
TransferStatusSelector_Stopped ,
TransferStatusSelector_QueueOverflow ,
NUM_TRANSFERSTATUSSELECTOR }
- enum spinTransferComponentSelectorEnums {
TransferComponentSelector_Red ,
TransferComponentSelector_Green ,

TransferComponentSelector_Blue ,
TransferComponentSelector_All ,
NUM_TRANSFERCOMPONENTSELECTOR }

- enum spinScan3dDistanceUnitEnums {
Scan3dDistanceUnit_Millimeter ,
Scan3dDistanceUnit_Inch ,
NUM_SCAN3DDISTANCEUNIT }
- enum spinScan3dCoordinateSystemEnums {
Scan3dCoordinateSystem_Cartesian ,
Scan3dCoordinateSystem_Spherical ,
Scan3dCoordinateSystem_Cylindrical ,
NUM_SCAN3DCOORDINATESYSTEM }
- enum spinScan3dOutputModeEnums {
Scan3dOutputMode_UncalibratedC ,
Scan3dOutputMode_CalibratedABC_Grid ,
Scan3dOutputMode_CalibratedABC_PointCloud ,
Scan3dOutputMode_CalibratedAC ,
Scan3dOutputMode_CalibratedAC_Linescan ,
Scan3dOutputMode_CalibratedC ,
Scan3dOutputMode_CalibratedC_Linescan ,
Scan3dOutputMode_RectifiedC ,
Scan3dOutputMode_RectifiedC_Linescan ,
Scan3dOutputMode_DisparityC ,
Scan3dOutputMode_DisparityC_Linescan ,
NUM_SCAN3DOUTPUTMODE }
- enum spinScan3dCoordinateSystemReferenceEnums {
Scan3dCoordinateSystemReference_Anchor ,
Scan3dCoordinateSystemReference_Transformed ,
NUM_SCAN3DCOORDINATESYSTEMREFERENCE }
- enum spinScan3dCoordinateSelectorEnums {
Scan3dCoordinateSelector_CoordinateA ,
Scan3dCoordinateSelector_CoordinateB ,
Scan3dCoordinateSelector_CoordinateC ,
NUM_SCAN3DCOORDINATESELECTOR }
- enum spinScan3dCoordinateTransformSelectorEnums {
Scan3dCoordinateTransformSelector_RotationX ,
Scan3dCoordinateTransformSelector_RotationY ,
Scan3dCoordinateTransformSelector_RotationZ ,
Scan3dCoordinateTransformSelector_TranslationX ,
Scan3dCoordinateTransformSelector_TranslationY ,
Scan3dCoordinateTransformSelector_TranslationZ ,
NUM_SCAN3DCOORDINATETRANSFORMSELECTOR }
- enum spinScan3dCoordinateReferenceSelectorEnums {
Scan3dCoordinateReferenceSelector_RotationX ,
Scan3dCoordinateReferenceSelector_RotationY ,
Scan3dCoordinateReferenceSelector_RotationZ ,
Scan3dCoordinateReferenceSelector_TranslationX ,
Scan3dCoordinateReferenceSelector_TranslationY ,
Scan3dCoordinateReferenceSelector_TranslationZ ,
NUM_SCAN3DCOORDINATEREFERENCESELECTOR }
- enum spinChunkImageComponentEnums {
ChunkImageComponent_Intensity ,
ChunkImageComponent_Color ,
ChunkImageComponent_Infrared ,
ChunkImageComponent_Ultraviolet ,
ChunkImageComponent_Range ,
ChunkImageComponent_Disparity ,
ChunkImageComponent_Confidence ,

ChunkImageComponent_Scatter ,
NUM_CHUNKIMAGECOMPONENT }
- enum spinChunkCounterSelectorEnums {
ChunkCounterSelector_Counter0 ,
ChunkCounterSelector_Counter1 ,
ChunkCounterSelector_Counter2 ,
NUM_CHUNKCOUNTERSELECTOR }
- enum spinChunkTimerSelectorEnums {
ChunkTimerSelector_Timer0 ,
ChunkTimerSelector_Timer1 ,
ChunkTimerSelector_Timer2 ,
NUM_CHUNKTIMERSELECTOR }
- enum spinChunkEncoderSelectorEnums {
ChunkEncoderSelector_Encoder0 ,
ChunkEncoderSelector_Encoder1 ,
ChunkEncoderSelector_Encoder2 ,
NUM_CHUNKENCODERSELECTOR }
- enum spinChunkEncoderStatusEnums {
ChunkEncoderStatus_EncoderUp ,
ChunkEncoderStatus_EncoderDown ,
ChunkEncoderStatus_EncoderIdle ,
ChunkEncoderStatus_EncoderStatic ,
NUM_CHUNKENCODERSTATUS }
- enum spinChunkExposureTimeSelectorEnums {
ChunkExposureTimeSelector_Common ,
ChunkExposureTimeSelector_Red ,
ChunkExposureTimeSelector_Green ,
ChunkExposureTimeSelector_Blue ,
ChunkExposureTimeSelector_Cyan ,
ChunkExposureTimeSelector_Magenta ,
ChunkExposureTimeSelector_Yellow ,
ChunkExposureTimeSelector_Infrared ,
ChunkExposureTimeSelector_Ultraviolet ,
ChunkExposureTimeSelector_Stage1 ,
ChunkExposureTimeSelector_Stage2 ,
NUM_CHUNKEXPOSURETIMESELECTOR }
- enum spinChunkSourceIDEnums {
ChunkSourceID_Source0 ,
ChunkSourceID_Source1 ,
ChunkSourceID_Source2 ,
NUM_CHUNKSOURCEID }
- enum spinChunkRegionIDEnums {
ChunkRegionID_Region0 ,
ChunkRegionID_Region1 ,
ChunkRegionID_Region2 ,
NUM_CHUNKREGIONID }
- enum spinChunkTransferStreamIDEnums {
ChunkTransferStreamID_Stream0 ,
ChunkTransferStreamID_Stream1 ,
ChunkTransferStreamID_Stream2 ,
ChunkTransferStreamID_Stream3 ,
NUM_CHUNKTRANSFERSTREAMID }
- enum spinChunkScan3dDistanceUnitEnums {
ChunkScan3dDistanceUnit_Millimeter ,
ChunkScan3dDistanceUnit_Inch ,
NUM_CHUNKSCAN3DDISTANCEUNIT }
- enum spinChunkScan3dOutputModeEnums {
ChunkScan3dOutputMode_UncalibratedC ,

ChunkScan3dOutputMode_CalibratedABC_Grid ,
ChunkScan3dOutputMode_CalibratedABC_PointCloud ,
ChunkScan3dOutputMode_CalibratedAC ,
ChunkScan3dOutputMode_CalibratedAC_Linescan ,
ChunkScan3dOutputMode_CalibratedC ,
ChunkScan3dOutputMode_CalibratedC_Linescan ,
ChunkScan3dOutputMode_RectifiedC ,
ChunkScan3dOutputMode_RectifiedC_Linescan ,
ChunkScan3dOutputMode_DisparityC ,
ChunkScan3dOutputMode_DisparityC_Linescan ,
NUM_CHUNKSCAN3DOUTPUTMODE }

- enum spinChunkScan3dCoordinateSystemEnums {
  ChunkScan3dCoordinateSystem_Cartesian ,
  ChunkScan3dCoordinateSystem_Spherical ,
  ChunkScan3dCoordinateSystem_Cylindrical ,
  NUM_CHUNKSCAN3DCOORDINATESYSTEM }
- enum spinChunkScan3dCoordinateSystemReferenceEnums {
  ChunkScan3dCoordinateSystemReference_Anchor ,
  ChunkScan3dCoordinateSystemReference_Transformed ,
  NUM_CHUNKSCAN3DCOORDINATESYSTEMREFERENCE }
- enum spinChunkScan3dCoordinateSelectorEnums {
  ChunkScan3dCoordinateSelector_CoordinateA ,
  ChunkScan3dCoordinateSelector_CoordinateB ,
  ChunkScan3dCoordinateSelector_CoordinateC ,
  NUM_CHUNKSCAN3DCOORDINATESELECTOR }
- enum spinChunkScan3dCoordinateTransformSelectorEnums {
  ChunkScan3dCoordinateTransformSelector_RotationX ,
  ChunkScan3dCoordinateTransformSelector_RotationY ,
  ChunkScan3dCoordinateTransformSelector_RotationZ ,
  ChunkScan3dCoordinateTransformSelector_TranslationX ,
  ChunkScan3dCoordinateTransformSelector_TranslationY ,
  ChunkScan3dCoordinateTransformSelector_TranslationZ ,
  NUM_CHUNKSCAN3DCOORDINATETRANSFORMSELECTOR }
- enum spinChunkScan3dCoordinateReferenceSelectorEnums {
  ChunkScan3dCoordinateReferenceSelector_RotationX ,
  ChunkScan3dCoordinateReferenceSelector_RotationY ,
  ChunkScan3dCoordinateReferenceSelector_RotationZ ,
  ChunkScan3dCoordinateReferenceSelector_TranslationX ,
  ChunkScan3dCoordinateReferenceSelector_TranslationY ,
  ChunkScan3dCoordinateReferenceSelector_TranslationZ ,
  NUM_CHUNKSCAN3DCOORDINATEREFERENCESELECTOR }
- enum spinDeviceTapGeometryEnums {
  DeviceTapGeometry_Geometry_1X_1Y ,
  DeviceTapGeometry_Geometry_1X2_1Y ,
  DeviceTapGeometry_Geometry_1X2_1Y2 ,
  DeviceTapGeometry_Geometry_2X_1Y ,
  DeviceTapGeometry_Geometry_2X_1Y2Geometry_2XE_1Y ,
  DeviceTapGeometry_Geometry_2XE_1Y2 ,
  DeviceTapGeometry_Geometry_2XM_1Y ,
  DeviceTapGeometry_Geometry_2XM_1Y2 ,
  DeviceTapGeometry_Geometry_1X_1Y2 ,
  DeviceTapGeometry_Geometry_1X_2YE ,
  DeviceTapGeometry_Geometry_1X3_1Y ,
  DeviceTapGeometry_Geometry_3X_1Y ,
  DeviceTapGeometry_Geometry_1X ,
  DeviceTapGeometry_Geometry_1X2 ,
  DeviceTapGeometry_Geometry_2X ,
  DeviceTapGeometry_Geometry_2XE ,

DeviceTapGeometry_Geometry_2XM ,
DeviceTapGeometry_Geometry_1X3 ,
DeviceTapGeometry_Geometry_3X ,
DeviceTapGeometry_Geometry_1X4_1Y ,
DeviceTapGeometry_Geometry_4X_1Y ,
DeviceTapGeometry_Geometry_2X2_1Y ,
DeviceTapGeometry_Geometry_2X2E_1YGeometry_2X2M_1Y ,
DeviceTapGeometry_Geometry_1X2_2YE ,
DeviceTapGeometry_Geometry_2X_2YE ,
DeviceTapGeometry_Geometry_2XE_2YE ,
DeviceTapGeometry_Geometry_2XM_2YE ,
DeviceTapGeometry_Geometry_1X4 ,
DeviceTapGeometry_Geometry_4X ,
DeviceTapGeometry_Geometry_2X2 ,
DeviceTapGeometry_Geometry_2X2E ,
DeviceTapGeometry_Geometry_2X2M ,
DeviceTapGeometry_Geometry_1X8_1Y ,
DeviceTapGeometry_Geometry_8X_1Y ,
DeviceTapGeometry_Geometry_4X2_1Y ,
DeviceTapGeometry_Geometry_2X2E_2YE ,
DeviceTapGeometry_Geometry_1X8 ,
DeviceTapGeometry_Geometry_8X ,
DeviceTapGeometry_Geometry_4X2 ,
DeviceTapGeometry_Geometry_4X2E ,
DeviceTapGeometry_Geometry_4X2E_1Y ,
DeviceTapGeometry_Geometry_1X10_1Y ,
DeviceTapGeometry_Geometry_10X_1Y ,
DeviceTapGeometry_Geometry_1X10 ,
DeviceTapGeometry_Geometry_10X ,
NUM_DEVICETAPGEOMETRY }

- enum spinGevPhysicalLinkConfigurationEnums {
GevPhysicalLinkConfiguration_SingleLink ,
GevPhysicalLinkConfiguration_MultiLink ,
GevPhysicalLinkConfiguration_StaticLAG ,
GevPhysicalLinkConfiguration_DynamicLAG ,
NUM_GEVPHYSICALLINKCONFIGURATION }
- enum spinGevCurrentPhysicalLinkConfigurationEnums {
GevCurrentPhysicalLinkConfiguration_SingleLink ,
GevCurrentPhysicalLinkConfiguration_MultiLink ,
GevCurrentPhysicalLinkConfiguration_StaticLAG ,
GevCurrentPhysicalLinkConfiguration_DynamicLAG ,
NUM_GEVCURRENTPHYSICALLINKCONFIGURATION }
- enum spinGevIPConfigurationStatusEnums {
GevIPConfigurationStatus_None ,
GevIPConfigurationStatus_PersistentIP ,
GevIPConfigurationStatus_DHCP ,
GevIPConfigurationStatus_LLA ,
GevIPConfigurationStatus_ForceIP ,
NUM_GEVIPCONFIGURATIONSTATUS }
- enum spinGevGVCPExtendedStatusCodesSelectorEnums {
GevGVCPExtendedStatusCodesSelector_Version1_1 ,
GevGVCPExtendedStatusCodesSelector_Version2_0 ,
NUM_GEVGVCPEXTENDEDSTATUSCODESSELECTOR }
- enum spinGevGVSPExtendedIDModeEnums {
GevGVSPExtendedIDMode_Off ,
GevGVSPExtendedIDMode_On ,
NUM_GEVGVSPEXTENDEDIDMODE }
- enum spinClConfigurationEnums {

ClConfiguration_Base ,
ClConfiguration_Medium ,
ClConfiguration_Full ,
ClConfiguration_DualBase ,
ClConfiguration_EightyBit ,
NUM_CLCONFIGURATION }
- enum spinClTimeSlotsCountEnums {
ClTimeSlotsCount_One ,
ClTimeSlotsCount_Two ,
ClTimeSlotsCount_Three ,
NUM_CLTIMESLOTSCOUNT }
- enum spinCxpLinkConfigurationStatusEnums {
CxpLinkConfigurationStatus_None ,
CxpLinkConfigurationStatus_Pending ,
CxpLinkConfigurationStatus_CXP1_X1 ,
CxpLinkConfigurationStatus_CXP2_X1 ,
CxpLinkConfigurationStatus_CXP3_X1 ,
CxpLinkConfigurationStatus_CXP5_X1 ,
CxpLinkConfigurationStatus_CXP6_X1 ,
CxpLinkConfigurationStatus_CXP1_X2 ,
CxpLinkConfigurationStatus_CXP2_X2 ,
CxpLinkConfigurationStatus_CXP3_X2 ,
CxpLinkConfigurationStatus_CXP5_X2 ,
CxpLinkConfigurationStatus_CXP6_X2 ,
CxpLinkConfigurationStatus_CXP1_X3 ,
CxpLinkConfigurationStatus_CXP2_X3 ,
CxpLinkConfigurationStatus_CXP3_X3 ,
CxpLinkConfigurationStatus_CXP5_X3 ,
CxpLinkConfigurationStatus_CXP6_X3 ,
CxpLinkConfigurationStatus_CXP1_X4 ,
CxpLinkConfigurationStatus_CXP2_X4 ,
CxpLinkConfigurationStatus_CXP3_X4 ,
CxpLinkConfigurationStatus_CXP5_X4 ,
CxpLinkConfigurationStatus_CXP6_X4 ,
CxpLinkConfigurationStatus_CXP1_X5 ,
CxpLinkConfigurationStatus_CXP2_X5 ,
CxpLinkConfigurationStatus_CXP3_X5 ,
CxpLinkConfigurationStatus_CXP5_X5 ,
CxpLinkConfigurationStatus_CXP6_X5 ,
CxpLinkConfigurationStatus_CXP1_X6 ,
CxpLinkConfigurationStatus_CXP2_X6 ,
CxpLinkConfigurationStatus_CXP3_X6 ,
CxpLinkConfigurationStatus_CXP5_X6 ,
CxpLinkConfigurationStatus_CXP6_X6 ,
NUM_CXPLINKCONFIGURATIONSTATUS }
- enum spinCxpLinkConfigurationPreferredEnums {
CxpLinkConfigurationPreferred_CXP1_X1 ,
CxpLinkConfigurationPreferred_CXP2_X1 ,
CxpLinkConfigurationPreferred_CXP3_X1 ,
CxpLinkConfigurationPreferred_CXP5_X1 ,
CxpLinkConfigurationPreferred_CXP6_X1 ,
CxpLinkConfigurationPreferred_CXP1_X2 ,
CxpLinkConfigurationPreferred_CXP2_X2 ,
CxpLinkConfigurationPreferred_CXP3_X2 ,
CxpLinkConfigurationPreferred_CXP5_X2 ,
CxpLinkConfigurationPreferred_CXP6_X2 ,
CxpLinkConfigurationPreferred_CXP1_X3 ,
CxpLinkConfigurationPreferred_CXP2_X3 ,

CxpLinkConfigurationPreferred_CXP3_X3 ,
CxpLinkConfigurationPreferred_CXP5_X3 ,
CxpLinkConfigurationPreferred_CXP6_X3 ,
CxpLinkConfigurationPreferred_CXP1_X4 ,
CxpLinkConfigurationPreferred_CXP2_X4 ,
CxpLinkConfigurationPreferred_CXP3_X4 ,
CxpLinkConfigurationPreferred_CXP5_X4 ,
CxpLinkConfigurationPreferred_CXP6_X4 ,
CxpLinkConfigurationPreferred_CXP1_X5 ,
CxpLinkConfigurationPreferred_CXP2_X5 ,
CxpLinkConfigurationPreferred_CXP3_X5 ,
CxpLinkConfigurationPreferred_CXP5_X5 ,
CxpLinkConfigurationPreferred_CXP6_X5 ,
CxpLinkConfigurationPreferred_CXP1_X6 ,
CxpLinkConfigurationPreferred_CXP2_X6 ,
CxpLinkConfigurationPreferred_CXP3_X6 ,
CxpLinkConfigurationPreferred_CXP5_X6 ,
CxpLinkConfigurationPreferred_CXP6_X6 ,
NUM_CXPLINKCONFIGURATIONPREFERRED }

- enum spinCxpLinkConfigurationEnums {
CxpLinkConfiguration_Auto ,
CxpLinkConfiguration_CXP1_X1 ,
CxpLinkConfiguration_CXP2_X1 ,
CxpLinkConfiguration_CXP3_X1 ,
CxpLinkConfiguration_CXP5_X1 ,
CxpLinkConfiguration_CXP6_X1 ,
CxpLinkConfiguration_CXP1_X2 ,
CxpLinkConfiguration_CXP2_X2 ,
CxpLinkConfiguration_CXP3_X2 ,
CxpLinkConfiguration_CXP5_X2 ,
CxpLinkConfiguration_CXP6_X2 ,
CxpLinkConfiguration_CXP1_X3 ,
CxpLinkConfiguration_CXP2_X3 ,
CxpLinkConfiguration_CXP3_X3 ,
CxpLinkConfiguration_CXP5_X3 ,
CxpLinkConfiguration_CXP6_X3 ,
CxpLinkConfiguration_CXP1_X4 ,
CxpLinkConfiguration_CXP2_X4 ,
CxpLinkConfiguration_CXP3_X4 ,
CxpLinkConfiguration_CXP5_X4 ,
CxpLinkConfiguration_CXP6_X4 ,
CxpLinkConfiguration_CXP1_X5 ,
CxpLinkConfiguration_CXP2_X5 ,
CxpLinkConfiguration_CXP3_X5 ,
CxpLinkConfiguration_CXP5_X5 ,
CxpLinkConfiguration_CXP6_X5 ,
CxpLinkConfiguration_CXP1_X6 ,
CxpLinkConfiguration_CXP2_X6 ,
CxpLinkConfiguration_CXP3_X6 ,
CxpLinkConfiguration_CXP5_X6 ,
CxpLinkConfiguration_CXP6_X6 ,
NUM_CXPLINKCONFIGURATION }
- enum spinCxpConnectionTestModeEnums {
CxpConnectionTestMode_Off ,
CxpConnectionTestMode_Mode1 ,
NUM_CXPCONNECTIONTESTMODE }
- enum spinCxpPoCxpStatusEnums {
CxpPoCxpStatus_Auto ,

CxpPoCxpStatus_Off ,
CxpPoCxpStatus_Tripped ,
NUM_CXPPOCXPSTATUS }

13.8.1 Enumeration Type Documentation

13.8.1.1 spinAcquisitionModeEnums

```
enum spinAcquisitionModeEnums
```

< Sets the acquisition mode of the device. Continuous: acquires images continuously. Multi Frame: acquires a specified number of images before stopping acquisition. Single Frame: acquires 1 image before stopping acquisition.

Enumerator

AcquisitionMode_Continuous	
AcquisitionMode_SingleFrame	
AcquisitionMode_MultiFrame	
NUM_ACQUISITIONMODE	

13.8.1.2 spinAcquisitionStatusSelectorEnums

```
enum spinAcquisitionStatusSelectorEnums
```

< Selects the internal acquisition signal to read using AcquisitionStatus.

Enumerator

AcquisitionStatusSelector_AcquisitionTriggerWait	Device is currently waiting for a trigger for the capture of one or many frames.
AcquisitionStatusSelector_AcquisitionActive	Device is currently doing an acquisition of one or many frames.
AcquisitionStatusSelector_AcquisitionTransfer	Device is currently transferring an acquisition of one or many frames.
AcquisitionStatusSelector_FrameTriggerWait	Device is currently waiting for a frame start trigger.
AcquisitionStatusSelector_FrameActive	Device is currently doing the capture of a frame.
AcquisitionStatusSelector_ExposureActive	Device is doing the exposure of a frame.
NUM_ACQUISITIONSTATUSSELECTOR	

13.8.1.3 spinActionUnconditionalModeEnums

```
enum spinActionUnconditionalModeEnums
```

< Enables the unconditional action command mode where action commands are processed even when the primary control channel is closed.

Enumerator

ActionUnconditionalMode_Off	Unconditional mode is disabled.
ActionUnconditionalMode_On	Unconditional mode is enabled.
NUM_ACTIONUNCONDITIONALMODE	

13.8.1.4 spinAdcBitDepthEnums

```
enum spinAdcBitDepthEnums
```

< Selects which ADC bit depth to use. A higher ADC bit depth results in better image quality but slower maximum frame rate.

Enumerator

AdcBitDepth_Bit8	
AdcBitDepth_Bit10	
AdcBitDepth_Bit12	
AdcBitDepth_Bit14	
NUM_ADCBITDEPTH	

13.8.1.5 spinAutoAlgorithmSelectorEnums

```
enum spinAutoAlgorithmSelectorEnums
```

< Selects which Auto Algorithm is controlled by the RoiEnable, OffsetX, OffsetY, Width, Height features.

Enumerator

AutoAlgorithmSelector_Awb	Selects the Auto White Balance algorithm.
AutoAlgorithmSelector_Ae	Selects the Auto Exposure algorithm.
NUM_AUTOALGORITHMSELECTOR	

13.8.1.6 spinAutoExposureControlPriorityEnums

```
enum spinAutoExposureControlPriorityEnums
```

< Selects whether to adjust gain or exposure first. When gain priority is selected, the camera fixes the gain to 0 dB, and the exposure is adjusted according to the target grey level. If the maximum exposure is reached before the target grey level is hit, the gain starts to change to meet the target. This mode is used to have the minimum noise. When exposure priority is selected, the camera sets the exposure to a small value (default is 5 ms). The gain is adjusted according to the target grey level. If maximum gain is reached before the target grey level is hit, the exposure starts to change to meet the target. This mode is used to capture fast motion.

Enumerator

AutoExposureControlPriority_Gain	
AutoExposureControlPriority_ExposureTime	
NUM_AUTOEXPOSURECONTROLPRIORITY	

13.8.1.7 spinAutoExposureLightingModeEnums

```
enum spinAutoExposureLightingModeEnums
```

< Selects a lighting mode: Backlight, Frontlight or Normal (default). a. Backlight compensation: used when a strong light is coming from the back of the object. b. Frontlight compensation: used when a strong light is shining in the front of the object while the background is dark. c. Normal lighting: used when the object is not under backlight or frontlight conditions. When normal lighting is selected, metering modes are available.

Enumerator

AutoExposureLightingMode_AutoDetect	
AutoExposureLightingMode_Backlight	
AutoExposureLightingMode_Frontlight	
AutoExposureLightingMode_Normal	
NUM_AUTOEXPOSURELIGHTINGMODE	

13.8.1.8 spinAutoExposureMeteringModeEnums

```
enum spinAutoExposureMeteringModeEnums
```

< Selects a metering mode: average, spot, or partial metering. a. Average: Measures the light from the entire scene uniformly to determine the final exposure value. Every portion of the exposed area has the same contribution. b. Spot: Measures a small area (about 3%) in the center of the scene while the rest of the scene is ignored. This mode is used when the scene has a high contrast and the object of interest is relatively small. c. Partial: Measures the light from a larger area (about 11%) in the center of the scene. This mode is used when very dark or bright regions appear at the edge of the frame. Note: Metering mode is available only when Lighting Mode Selector is Normal.

Enumerator

AutoExposureMeteringMode_Average	
AutoExposureMeteringMode_Spot	
AutoExposureMeteringMode_Partial	
AutoExposureMeteringMode_CenterWeighted	
AutoExposureMeteringMode_HistgramPeak	
NUM_AUTOEXPOSUREMETERINGMODE	

13.8.1.9 spinAutoExposureTargetGreyValueAutoEnums

```
enum spinAutoExposureTargetGreyValueAutoEnums
```

< This indicates whether the target image grey level is automatically set by the camera or manually set by the user. Note that the target grey level is in the linear domain before gamma correction is applied.

Enumerator

AutoExposureTargetGreyValueAuto_Off	Target grey value is manually controlled
AutoExposureTargetGreyValueAuto_Continuous	Target grey value is constantly adapted by the device to maximize the dynamic range.
NUM_AUTOEXPOSURETARGETGREYVALUEAUTO	

13.8.1.10 spinBalanceRatioSelectorEnums

```
enum spinBalanceRatioSelectorEnums
```

< Selects a balance ratio to configure once a balance ratio control has been selected.

Enumerator

BalanceRatioSelector_Red	Selects the red balance ratio control for adjustment. The red balance ratio is relative to the green channel.
BalanceRatioSelector_Blue	Selects the blue balance ratio control for adjustment. The blue balance ratio is relative to the green channel.
NUM_BALANCERATIOSELECTOR	

13.8.1.11 spinBalanceWhiteAutoEnums

```
enum spinBalanceWhiteAutoEnums
```

< White Balance compensates for color shifts caused by different lighting conditions. It can be automatically or manually controlled. For manual control, set to Off. For automatic control, set to Once or Continuous.

Enumerator

BalanceWhiteAuto_Off	Sets operation mode to Off, which is manual control.
BalanceWhiteAuto_Once	Sets operation mode to once. Once runs for a number of iterations and then sets White Balance Auto to Off.
BalanceWhiteAuto_Continuous	Sets operation mode to continuous. Continuous automatically adjusts values if the colors are imbalanced.
NUM_BALANCEWHITEAUTO	

13.8.1.12 spinBalanceWhiteAutoProfileEnums

```
enum spinBalanceWhiteAutoProfileEnums
```

< Selects the profile used by BalanceWhiteAuto.

Enumerator

BalanceWhiteAutoProfile_Indoor	Indoor auto white balance Profile. Can be used to compensate for artificial lighting.
BalanceWhiteAutoProfile_Outdoor	Outdoor auto white balance profile. Designed for scenes with natural lighting.
NUM_BALANCEWHITEAUTOPROFILE	

13.8.1.13 spinBinningHorizontalModeEnums

```
enum spinBinningHorizontalModeEnums
```

<

Enumerator

BinningHorizontalMode_Sum	The response from the combined horizontal cells is added, resulting in increased sensitivity (a brighter image).
BinningHorizontalMode_Average	The response from the combined horizontal cells is averaged, resulting in increased signal/noise ratio. Not all sensors support average binning.
NUM_BINNINGHORIZONTALMODE	

13.8.1.14 spinBinningSelectorEnums

```
enum spinBinningSelectorEnums
```

< Selects which binning engine is controlled by the BinningHorizontal and BinningVertical features.

Enumerator

BinningSelector_All	The total amount of binning to be performed on the captured sensor data.
BinningSelector_Sensor	The portion of binning to be performed on the sensor directly.
BinningSelector_ISP	The portion of binning to be performed by the image signal processing engine (ISP) outside of the sensor. Note: the ISP can be disabled.
NUM_BINNINGSELECTOR	

13.8.1.15 spinBinningVerticalModeEnums

```
enum spinBinningVerticalModeEnums
```

<

Enumerator

BinningVerticalMode_Sum	The response from the combined vertical cells is added, resulting in increased sensitivity (a brighter image).
BinningVerticalMode_Average	The response from the combined vertical cells is averaged, resulting in increased signal/noise ratio. Not all sensors support average binning.
NUM_BINNINGVERTICALMODE	

13.8.1.16 spinBlackLevelAutoBalanceEnums

```
enum spinBlackLevelAutoBalanceEnums
```

< Controls the mode for automatic black level balancing between the sensor color channels or taps. The black level coefficients of each channel are adjusted so they are matched.

Enumerator

BlackLevelAutoBalance_Off	Black level tap balancing is user controlled using BlackLevel.
BlackLevelAutoBalance_Once	Black level tap balancing is automatically adjusted once by the device. Once it has converged, it automatically returns to the Off state.
BlackLevelAutoBalance_Continuous	Black level tap balancing is constantly adjusted by the device.
NUM_BLACKLEVELAUTOBALANCE	

13.8.1.17 spinBlackLevelAutoEnums

```
enum spinBlackLevelAutoEnums
```

< Controls the mode for automatic black level adjustment. The exact algorithm used to implement this adjustment is device-specific.

Enumerator

BlackLevelAuto_Off	Analog black level is user controlled using BlackLevel.
BlackLevelAuto_Once	Analog black level is automatically adjusted once by the device. Once it has converged, it automatically returns to the Off state.
BlackLevelAuto_Continuous	Analog black level is constantly adjusted by the device.
NUM_BLACKLEVELAUTO	

13.8.1.18 spinBlackLevelSelectorEnums

```
enum spinBlackLevelSelectorEnums
```

< Selects which black level to control. Only All can be set by the user. Analog and Digital are read-only.

Enumerator

BlackLevelSelector_All	
BlackLevelSelector_Analog	
BlackLevelSelector_Digital	
NUM_BLACKLEVELSELECTOR	

13.8.1.19 spinChunkBlackLevelSelectorEnums

```
enum spinChunkBlackLevelSelectorEnums
```

< Selects which black level to retrieve

Enumerator

ChunkBlackLevelSelector_All	
NUM_CHUNKBLACKLEVELSELECTOR	

13.8.1.20 spinChunkCounterSelectorEnums

```
enum spinChunkCounterSelectorEnums
```

< Selects which counter to retrieve data from.

Enumerator

ChunkCounterSelector_Counter0	Selects the counter 0.
ChunkCounterSelector_Counter1	Selects the counter 1.
ChunkCounterSelector_Counter2	Selects the counter 2.
NUM_CHUNKCOUNTERSELECTOR	

13.8.1.21 spinChunkEncoderSelectorEnums

```
enum spinChunkEncoderSelectorEnums
```

< Selects which Encoder to retrieve data from.

Enumerator

ChunkEncoderSelector_Encoder0	Selects the first Encoder.
ChunkEncoderSelector_Encoder1	Selects the first Encoder.
ChunkEncoderSelector_Encoder2	Selects the second Encoder.
NUM_CHUNKENCODERSELECTOR	

13.8.1.22 spinChunkEncoderStatusEnums

```
enum spinChunkEncoderStatusEnums
```

< Returns the motion status of the selected encoder.

Enumerator

ChunkEncoderStatus_EncoderUp	The encoder counter last incremented.
ChunkEncoderStatus_EncoderDown	The encoder counter last decremented.
ChunkEncoderStatus_EncoderIdle	The encoder is not active.
ChunkEncoderStatus_EncoderStatic	No motion within the EncoderTimeout time.
NUM_CHUNKENCODERSTATUS	

13.8.1.23 spinChunkExposureTimeSelectorEnums

```
enum spinChunkExposureTimeSelectorEnums
```

< Selects which exposure time is read by the ChunkExposureTime feature.

Enumerator

ChunkExposureTimeSelector_Common	Selects the common ExposureTime.
ChunkExposureTimeSelector_Red	Selects the red common ExposureTime.
ChunkExposureTimeSelector_Green	Selects the green ExposureTime.
ChunkExposureTimeSelector_Blue	Selects the blue ExposureTime.
ChunkExposureTimeSelector_Cyan	Selects the cyan common ExposureTime..
ChunkExposureTimeSelector_Magenta	Selects the magenta ExposureTime..
ChunkExposureTimeSelector_Yellow	Selects the yellow ExposureTime..
ChunkExposureTimeSelector_Infrared	Selects the infrared ExposureTime.
ChunkExposureTimeSelector_Ultraviolet	Selects the ultraviolet ExposureTime.
ChunkExposureTimeSelector_Stage1	Selects the first stage ExposureTime.
ChunkExposureTimeSelector_Stage2	Selects the second stage ExposureTime.
NUM_CHUNKEXPOSURETIMESELECTOR	

13.8.1.24 spinChunkGainSelectorEnums

```
enum spinChunkGainSelectorEnums
```

< Selects which gain to retrieve

Enumerator

ChunkGainSelector_All	
ChunkGainSelector_Red	
ChunkGainSelector_Green	
ChunkGainSelector_Blue	
NUM_CHUNKGAINSELECTOR	

13.8.1.25 spinChunkImageComponentEnums

```
enum spinChunkImageComponentEnums
```

< Returns the component of the payload image. This can be used to identify the image component of a generic part in a multipart transfer.

Enumerator

ChunkImageComponent_Intensity	The image data is the intensity component.
ChunkImageComponent_Color	The image data is color component.
ChunkImageComponent_Infrared	The image data is infrared component.
ChunkImageComponent_Ultraviolet	The image data is the ultraviolet component.
ChunkImageComponent_Range	The image data is the range (distance) component.
ChunkImageComponent_Disparity	The image data is the disparity component.

Enumerator

ChunkImageComponent_Confidence	The image data is the confidence map component.
ChunkImageComponent_Scatter	The image data is the scatter component.
NUM_CHUNKIMAGECOMPONENT	

13.8.1.26 spinChunkPixelFormatEnums

```
enum spinChunkPixelFormatEnums
```

< Format of the pixel provided by the camera

Enumerator

ChunkPixelFormat_Mono8	
ChunkPixelFormat_Mono12Packed	
ChunkPixelFormat_Mono16	
ChunkPixelFormat_RGB8Packed	
ChunkPixelFormat_YUV422Packed	
ChunkPixelFormat_BayerGR8	
ChunkPixelFormat_BayerRG8	
ChunkPixelFormat_BayerGB8	
ChunkPixelFormat_BayerBG8	
ChunkPixelFormat_YCbCr601_422_8_CbYCrY	
NUM_CHUNKPIXELFORMAT	

13.8.1.27 spinChunkRegionIDEnums

```
enum spinChunkRegionIDEnums
```

< Returns the identifier of Region that the image comes from.

Enumerator

ChunkRegionID_Region0	Image comes from the Region 0.
ChunkRegionID_Region1	Image comes from the Region 1.
ChunkRegionID_Region2	Image comes from the Region 2.
NUM_CHUNKREGIONID	

13.8.1.28 spinChunkScan3dCoordinateReferenceSelectorEnums

```
enum spinChunkScan3dCoordinateReferenceSelectorEnums
```

< Selector to read a coordinate system reference value defining the transform of a point from one system to the other.

Enumerator

ChunkScan3dCoordinateReferenceSelector_RotationX	Rotation around X axis.
ChunkScan3dCoordinateReferenceSelector_RotationY	Rotation around Y axis.
ChunkScan3dCoordinateReferenceSelector_RotationZ	Rotation around Z axis.
ChunkScan3dCoordinateReferenceSelector_TranslationX	X axis translation.
ChunkScan3dCoordinateReferenceSelector_TranslationY	Y axis translation.
ChunkScan3dCoordinateReferenceSelector_TranslationZ	Z axis translation.
NUM_CHUNKSCAN3DCOORDINATEREFERENCESELECTOR	

13.8.1.29 spinChunkScan3dCoordinateSelectorEnums

```
enum spinChunkScan3dCoordinateSelectorEnums
```

< Selects which Coordinate to retrieve data from.

Enumerator

ChunkScan3dCoordinateSelector_CoordinateA	The first (X or Theta) coordinate
ChunkScan3dCoordinateSelector_CoordinateB	The second (Y or Phi) coordinate
ChunkScan3dCoordinateSelector_CoordinateC	The third (Z or Rho) coordinate.
NUM_CHUNKSCAN3DCOORDINATESELECTOR	

13.8.1.30 spinChunkScan3dCoordinateSystemEnums

```
enum spinChunkScan3dCoordinateSystemEnums
```

< Returns the Coordinate System of the image included in the payload.

Enumerator

ChunkScan3dCoordinateSystem_Cartesian	Default value. 3-axis orthogonal, right-hand X-Y-Z.
ChunkScan3dCoordinateSystem_Spherical	A Theta-Phi-Rho coordinate system.
ChunkScan3dCoordinateSystem_Cylindrical	A Theta-Y-Rho coordinate system.
NUM_CHUNKSCAN3DCOORDINATESYSTEM	

13.8.1.31 spinChunkScan3dCoordinateSystemReferenceEnums

```
enum spinChunkScan3dCoordinateSystemReferenceEnums
```

< Returns the Coordinate System Position of the image included in the payload.

Enumerator

ChunkScan3dCoordinateSystemReference_Anchor	Default value. Original fixed reference. The coordinate system fixed relative the camera reference point marker is used.
ChunkScan3dCoordinateSystemReference_↩ Transformed	Transformed reference system. The transformed coordinate system is used according to the definition in the rotation and translation matrices.
NUM_CHUNKSCAN3↩ DCOORDINATESYSTEMREFERENCE	

13.8.1.32 spinChunkScan3dCoordinateTransformSelectorEnums

```
enum spinChunkScan3dCoordinateTransformSelectorEnums
```

< Selector for transform values.

Enumerator

ChunkScan3dCoordinateTransformSelector_RotationX	Rotation around X axis.
ChunkScan3dCoordinateTransformSelector_RotationY	Rotation around Y axis.
ChunkScan3dCoordinateTransformSelector_RotationZ	Rotation around Z axis.
ChunkScan3dCoordinateTransformSelector_TranslationX	Translation along X axis.
ChunkScan3dCoordinateTransformSelector_TranslationY	Translation along Y axis.
ChunkScan3dCoordinateTransformSelector_TranslationZ	Translation along Z axis.
NUM_CHUNKSCAN3DCOORDINATETRANSFORMSELECTOR	

13.8.1.33 spinChunkScan3dDistanceUnitEnums

```
enum spinChunkScan3dDistanceUnitEnums
```

< Returns the Distance Unit of the payload image.

Enumerator

ChunkScan3dDistanceUnit_Millimeter	Default value. Distance values are in millimeter units.
ChunkScan3dDistanceUnit_Inch	Distance values are in inch units.
NUM_CHUNKSCAN3DDISTANCEUNIT	

13.8.1.34 spinChunkScan3dOutputModeEnums

```
enum spinChunkScan3dOutputModeEnums
```

< Returns the Calibrated Mode of the payload image.

Enumerator

ChunkScan3dOutputMode_UncalibratedC	Uncalibrated 2.5D Depth map. The distance data does not represent a physical unit and may be non-linear. The data is a 2.5D range map only.
ChunkScan3dOutputMode_CalibratedABC_Grid	3 Coordinates in grid organization. The full 3 coordinate data with the grid array organization from the sensor kept.
ChunkScan3dOutputMode_CalibratedABC_Point↩ Cloud	3 Coordinates without organization. The full 3 coordinate data without any organization of data points. Typically only valid points transmitted giving varying image size.
ChunkScan3dOutputMode_CalibratedAC	2 Coordinates with fixed B sampling. The data is sent as a A and C coordinates (X,Z or Theta,Rho). The B (Y) axis uses the scale and offset parameters for the B axis.
ChunkScan3dOutputMode_CalibratedAC_Linescan	2 Coordinates with varying sampling. The data is sent as a A and C coordinates (X,Z or Theta,Rho). The B (Y) axis comes from the encoder chunk value.
ChunkScan3dOutputMode_CalibratedC	Calibrated 2.5D Depth map. The distance data is expressed in the chosen distance unit. The data is a 2.5D range map. No information on X-Y axes available.
ChunkScan3dOutputMode_CalibratedC_Linescan	Depth Map with varying B sampling. The distance data is expressed in the chosen distance unit. The data is a 2.5D range map. The B (Y) axis comes from the encoder chunk value.
ChunkScan3dOutputMode_RectifiedC	Rectified 2.5D Depth map. The distance data has been rectified to a uniform sampling pattern in the X and Y direction. The data is a 2.5D range map only. If a complete 3D point cloud is rectified but transmitted as explicit coordinates it should be transmitted as one of the "CalibratedABC" formats.
ChunkScan3dOutputMode_RectifiedC_Linescan	Rectified 2.5D Depth map with varying B sampling. The data is sent as rectified 1D profiles using Coord3D_C pixels. The B (Y) axis comes from the encoder chunk value.
ChunkScan3dOutputMode_DisparityC	Disparity 2.5D Depth map. The distance is inversely proportional to the pixel (disparity) value.
ChunkScan3dOutputMode_DisparityC_Linescan	Disparity 2.5D Depth map with varying B sampling. The distance is inversely proportional to the pixel (disparity) value. The B (Y) axis comes from the encoder chunk value.
NUM_CHUNKSCAN3DOUTPUTMODE	

13.8.1.35 spinChunkSelectorEnums

```
enum spinChunkSelectorEnums
```

< Selects which chunk data to enable or disable.

Enumerator

ChunkSelector_Image	
ChunkSelector_CRC	
ChunkSelector_FrameID	
ChunkSelector_OffsetX	
ChunkSelector_OffsetY	
ChunkSelector_Width	
ChunkSelector_Height	
ChunkSelector_ExposureTime	
ChunkSelector_Gain	
ChunkSelector_BlackLevel	
ChunkSelector_PixelFormat	
ChunkSelector_Timestamp	
ChunkSelector_SequencerSetActive	
ChunkSelector_SerialData	
ChunkSelector_ExposureEndLineStatusAll	
NUM_CHUNKSELECTOR	

13.8.1.36 spinChunkSourceIDEnums

```
enum spinChunkSourceIDEnums
```

< Returns the identifier of Source that the image comes from.

Enumerator

ChunkSourceID_Source0	Image comes from the Source 0.
ChunkSourceID_Source1	Image comes from the Source 1.
ChunkSourceID_Source2	Image comes from the Source 2.
NUM_CHUNKSOURCEID	

13.8.1.37 spinChunkTimerSelectorEnums

```
enum spinChunkTimerSelectorEnums
```

< Selects which Timer to retrieve data from.

Enumerator

ChunkTimerSelector_Timer0	Selects the first Timer.
ChunkTimerSelector_Timer1	Selects the first Timer.
ChunkTimerSelector_Timer2	Selects the second Timer.
NUM_CHUNKTIMERSELECTOR	

13.8.1.38 spinChunkTransferStreamIDEnums

```
enum spinChunkTransferStreamIDEnums
```

< Returns identifier of the stream that generated this block.

Enumerator

ChunkTransferStreamID_Stream0	Data comes from Stream0.
ChunkTransferStreamID_Stream1	Data comes from Stream1.
ChunkTransferStreamID_Stream2	Data comes from Stream2.
ChunkTransferStreamID_Stream3	Data comes from Stream3.
NUM_CHUNKTRANSFERSTREAMID	

13.8.1.39 spinClConfigurationEnums

```
enum spinClConfigurationEnums
```

< This Camera Link specific feature describes the configuration used by the camera. It helps especially when a camera is capable of operation in a non-standard configuration, and when the features PixelSize, SensorDigitization↩ Taps, and DeviceTapGeometry do not provide enough information for interpretation of the image data provided by the camera.

Enumerator

ClConfiguration_Base	Standard base configuration described by the Camera Link standard.
ClConfiguration_Medium	Standard medium configuration described by the Camera Link standard.
ClConfiguration_Full	Standard full configuration described by the Camera Link standard.
ClConfiguration_DualBase	The camera streams the data from multiple taps (that do not fit in the standard base configuration) through two Camera Link base ports. It is responsibility of the application or frame grabber to reconstruct the full image. Only one of the ports (fixed) serves as the "master" for serial communication and triggering.
ClConfiguration_EightyBit	Standard 80-bit configuration with 10 taps of 8 bits or 8 taps of 10 bits, as described by the Camera Link standard.
NUM_CLCONFIGURATION	

13.8.1.40 spinClTimeSlotsCountEnums

```
enum spinClTimeSlotsCountEnums
```

< This Camera Link specific feature describes the time multiplexing of the camera link connection to transfer more than the configuration allows, in one single clock.

Enumerator

ClTimeSlotsCount_One	One
ClTimeSlotsCount_Two	Two
ClTimeSlotsCount_Three	Three
NUM_CLTIMESLOTSCOUNT	

13.8.1.41 spinColorTransformationSelectorEnums

```
enum spinColorTransformationSelectorEnums
```

< Selects which Color Transformation module is controlled by the various Color Transformation features

Enumerator

ColorTransformationSelector_RGBtoRGB	
ColorTransformationSelector_RGBtoYUV	
NUM_COLORTRANSFORMATIONSELECTOR	

13.8.1.42 spinColorTransformationValueSelectorEnums

```
enum spinColorTransformationValueSelectorEnums
```

< Selects the Gain factor or Offset of the Transformation matrix to access in the selected Color Transformation module

Enumerator

ColorTransformationValueSelector_Gain00	
ColorTransformationValueSelector_Gain01	
ColorTransformationValueSelector_Gain02	
ColorTransformationValueSelector_Gain10	
ColorTransformationValueSelector_Gain11	
ColorTransformationValueSelector_Gain12	
ColorTransformationValueSelector_Gain20	
ColorTransformationValueSelector_Gain21	
ColorTransformationValueSelector_Gain22	
ColorTransformationValueSelector_Offset0	
ColorTransformationValueSelector_Offset1	
ColorTransformationValueSelector_Offset2	
NUM_COLORTRANSFORMATIONVALUESELECTOR	

13.8.1.43 spinCompressionSaturationPriorityEnums

```
enum spinCompressionSaturationPriorityEnums
```

< When FrameRate is enabled, camera drops frames if datarate is saturated. If FrameRate is disabled, camera adjusts the framerate to match the maximum achievable datarate.

Enumerator

CompressionSaturationPriority_DropFrame	Frames which will cause the MaxDatarateThreshold to be exceeded will not be transmitted. Requires FrameRateEnable to be True
CompressionSaturationPriority_ReduceFrameRate	AcquisitionFrameRate is dynamically adjusted to the highest possible value without exceeding the MaxDatarateThreshold.
NUM_COMPRESSIONSATURATIONPRIORITY	

13.8.1.44 spinCounterEventActivationEnums

```
enum spinCounterEventActivationEnums
```

< Selects the activation mode of the event to increment the Counter.

Enumerator

CounterEventActivation_LevelLow	
CounterEventActivation_LevelHigh	
CounterEventActivation_FallingEdge	
CounterEventActivation_RisingEdge	
CounterEventActivation_AnyEdge	
NUM_COUNTEREVENTACTIVATION	

13.8.1.45 spinCounterEventSourceEnums

```
enum spinCounterEventSourceEnums
```

< Selects the event that will increment the counter

Enumerator

CounterEventSource_Off	Off
CounterEventSource_MHzTick	MHzTick
CounterEventSource_Line0	Line0

Enumerator

CounterEventSource_Line1	Line1
CounterEventSource_Line2	Line2
CounterEventSource_Line3	Line3
CounterEventSource_UserOutput0	UserOutput0
CounterEventSource_UserOutput1	UserOutput1
CounterEventSource_UserOutput2	UserOutput2
CounterEventSource_UserOutput3	UserOutput3
CounterEventSource_Counter0Start	Counter0Start
CounterEventSource_Counter1Start	Counter1Start
CounterEventSource_Counter0End	Counter0End
CounterEventSource_Counter1End	Counter1End
CounterEventSource_LogicBlock0	LogicBlock0
CounterEventSource_LogicBlock1	LogicBlock1
CounterEventSource_ExposureStart	ExposureStart
CounterEventSource_ExposureEnd	ExposureEnd
CounterEventSource_FrameTriggerWait	FrameTriggerWait
NUM_COUNTEREVENTSOURCE	

13.8.1.46 spinCounterResetActivationEnums

`enum spinCounterResetActivationEnums`

< Selects the Activation mode of the Counter Reset Source signal.

Enumerator

CounterResetActivation_LevelLow	
CounterResetActivation_LevelHigh	
CounterResetActivation_FallingEdge	
CounterResetActivation_RisingEdge	
CounterResetActivation_AnyEdge	
NUM_COUNTERRESETACTIVATION	

13.8.1.47 spinCounterResetSourceEnums

`enum spinCounterResetSourceEnums`

< Selects the signal that will be the source to reset the counter.

Enumerator

CounterResetSource_Off	Off

Enumerator

CounterResetSource_Line0	Line0
CounterResetSource_Line1	Line1
CounterResetSource_Line2	Line2
CounterResetSource_Line3	Line3
CounterResetSource_UserOutput0	UserOutput0
CounterResetSource_UserOutput1	UserOutput1
CounterResetSource_UserOutput2	UserOutput2
CounterResetSource_UserOutput3	UserOutput3
CounterResetSource_Counter0Start	Counter0Start
CounterResetSource_Counter1Start	Counter1Start
CounterResetSource_Counter0End	Counter0End
CounterResetSource_Counter1End	Counter1End
CounterResetSource_LogicBlock0	LogicBlock0
CounterResetSource_LogicBlock1	LogicBlock1
CounterResetSource_ExposureStart	ExposureStart
CounterResetSource_ExposureEnd	ExposureEnd
CounterResetSource_FrameTriggerWait	FrameTriggerWait
NUM_COUNTERRESETSOURCE	

13.8.1.48 spinCounterSelectorEnums

```
enum spinCounterSelectorEnums
```

< Selects which counter to configure

Enumerator

CounterSelector_Counter0	
CounterSelector_Counter1	
NUM_COUNTERSELECTOR	

13.8.1.49 spinCounterStatusEnums

```
enum spinCounterStatusEnums
```

< Returns the current status of the counter.

Enumerator

CounterStatus_CounterIdle	The counter is idle.
CounterStatus_CounterTriggerWait	The counter is waiting for a start trigger.
CounterStatus_CounterActive	The counter is counting for the specified duration.
CounterStatus_CounterCompleted	The counter reached the CounterDuration count.
CounterStatus_CounterOverflow	The counter reached its maximum possible count.
NUM_COUNTERSTATUS	

13.8.1.50 spinCounterTriggerActivationEnums

```
enum spinCounterTriggerActivationEnums
```

< Selects the activation mode of the trigger to start the counter.

Enumerator

CounterTriggerActivation_LevelLow	
CounterTriggerActivation_LevelHigh	
CounterTriggerActivation_FallingEdge	
CounterTriggerActivation_RisingEdge	
CounterTriggerActivation_AnyEdge	
NUM_COUNTERTRIGGERACTIVATION	

13.8.1.51 spinCounterTriggerSourceEnums

```
enum spinCounterTriggerSourceEnums
```

< Selects the source of the trigger to start the counter

Enumerator

CounterTriggerSource_Off	Off
CounterTriggerSource_Line0	Line0
CounterTriggerSource_Line1	Line1
CounterTriggerSource_Line2	Line2
CounterTriggerSource_Line3	Line3
CounterTriggerSource_UserOutput0	UserOutput0
CounterTriggerSource_UserOutput1	UserOutput1
CounterTriggerSource_UserOutput2	UserOutput2
CounterTriggerSource_UserOutput3	UserOutput3
CounterTriggerSource_Counter0Start	Counter0Start
CounterTriggerSource_Counter1Start	Counter1Start
CounterTriggerSource_Counter0End	Counter0End
CounterTriggerSource_Counter1End	Counter1End
CounterTriggerSource_LogicBlock0	LogicBlock0
CounterTriggerSource_LogicBlock1	LogicBlock1
CounterTriggerSource_ExposureStart	ExposureStart
CounterTriggerSource_ExposureEnd	ExposureEnd
CounterTriggerSource_FrameTriggerWait	FrameTriggerWait
NUM_COUNTERTRIGGERSOURCE	

13.8.1.52 spinCxpConnectionTestModeEnums

```
enum spinCxpConnectionTestModeEnums
```

< Enables the test mode for an individual physical connection of the Device.

Enumerator

CxpConnectionTestMode_Off	Off
CxpConnectionTestMode_Mode1	Mode 1
NUM_CXPCONNECTIONTESTMODE	

13.8.1.53 spinCxpLinkConfigurationEnums

```
enum spinCxpLinkConfigurationEnums
```

< This feature allows specifying the Link configuration for the communication between the Receiver and Transmitter Device. In most cases this feature does not need to be written because automatic discovery will set configuration correctly to the value returned by CxpLinkConfigurationPreferred. Note that the currently active configuration of the Link can be read using CxpLinkConfigurationStatus.

Enumerator

CxpLinkConfiguration_Auto	Sets Automatic discovery for the Link Configuration.
CxpLinkConfiguration_CXP1_X1	Force the Link to 1 Connection operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfiguration_CXP2_X1	Force the Link to 1 Connection operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfiguration_CXP3_X1	Force the Link to 1 Connection operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfiguration_CXP5_X1	Force the Link to 1 Connection operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfiguration_CXP6_X1	Force the Link to 1 Connection operating at CXP-6 speed (6.25 Gbps).
CxpLinkConfiguration_CXP1_X2	Force the Link to 2 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfiguration_CXP2_X2	Force the Link to 2 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfiguration_CXP3_X2	Force the Link to 2 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfiguration_CXP5_X2	Force the Link to 2 Connections operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfiguration_CXP6_X2	Force the Link to 3 Connections operating at CXP-6 speed (6.25 Gbps).
CxpLinkConfiguration_CXP1_X3	Force the Link to 3 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfiguration_CXP2_X3	Force the Link to 3 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfiguration_CXP3_X3	Force the Link to 3 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfiguration_CXP5_X3	Force the Link to 3 Connections operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfiguration_CXP6_X3	Force the Link to 3 Connections operating at CXP-6 speed (6.25 Gbps).
CxpLinkConfiguration_CXP1_X4	Force the Link to 4 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfiguration_CXP2_X4	Force the Link to 4 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfiguration_CXP3_X4	Force the Link to 4 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfiguration_CXP5_X4	Force the Link to 4 Connections operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfiguration_CXP6_X4	Force the Link to 4 Connections operating at CXP-6 speed (6.25 Gbps).

Enumerator

CxpLinkConfiguration_CXP1_X5	Force the Link to 5 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfiguration_CXP2_X5	Force the Link to 5 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfiguration_CXP3_X5	Force the Link to 5 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfiguration_CXP5_X5	Force the Link to 5 Connections operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfiguration_CXP6_X5	Force the Link to 5 Connections operating at CXP-6 speed (6.25 Gbps).
CxpLinkConfiguration_CXP1_X6	Force the Link to 6 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfiguration_CXP2_X6	Force the Link to 6 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfiguration_CXP3_X6	Force the Link to 6 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfiguration_CXP5_X6	Force the Link to 6 Connections operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfiguration_CXP6_X6	Force the Link to 6 Connections operating at CXP-6 speed (6.25 Gbps).
NUM_CXPLINKCONFIGURATION	

13.8.1.54 spinCxpLinkConfigurationPreferredEnums

```
enum spinCxpLinkConfigurationPreferredEnums
```

< Provides the Link configuration that allows the Transmitter Device to operate in its default mode.

Enumerator

CxpLinkConfigurationPreferred_CXP1_X1	1 Connection operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfigurationPreferred_CXP2_X1	1 Connection operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfigurationPreferred_CXP3_X1	1 Connection operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfigurationPreferred_CXP5_X1	1 Connection operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfigurationPreferred_CXP6_X1	1 Connection operating at CXP-6 speed (6.25 Gbps).
CxpLinkConfigurationPreferred_CXP1_X2	2 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfigurationPreferred_CXP2_X2	2 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfigurationPreferred_CXP3_X2	2 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfigurationPreferred_CXP5_X2	2 Connections operating at CXP-4 speed (5.00 Gbps).
CxpLinkConfigurationPreferred_CXP6_X2	3 Connections operating at CXP-5 speed (6.25 Gbps).
CxpLinkConfigurationPreferred_CXP1_X3	3 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfigurationPreferred_CXP2_X3	3 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfigurationPreferred_CXP3_X3	3 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfigurationPreferred_CXP5_X3	3 Connections operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfigurationPreferred_CXP6_X3	3 Connections operating at CXP-6 speed (6.25 Gbps).
CxpLinkConfigurationPreferred_CXP1_X4	4 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfigurationPreferred_CXP2_X4	4 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfigurationPreferred_CXP3_X4	4 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfigurationPreferred_CXP5_X4	4 Connections operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfigurationPreferred_CXP6_X4	4 Connections operating at CXP-6 speed (6.25 Gbps).
CxpLinkConfigurationPreferred_CXP1_X5	5 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfigurationPreferred_CXP2_X5	5 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfigurationPreferred_CXP3_X5	5 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfigurationPreferred_CXP5_X5	5 Connections operating at CXP-5 speed (5.00 Gbps).

Enumerator

CxpLinkConfigurationPreferred_CXP6_X5	5 Connections operating at CXP-6 speed (6.25 Gbps).
CxpLinkConfigurationPreferred_CXP1_X6	6 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfigurationPreferred_CXP2_X6	6 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfigurationPreferred_CXP3_X6	6 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfigurationPreferred_CXP5_X6	6 Connections operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfigurationPreferred_CXP6_X6	6 Connections operating at CXP-6 speed (6.25 Gbps).
NUM_CXPLINKCONFIGURATIONPREFERRED	

13.8.1.55 spinCxpLinkConfigurationStatusEnums

```
enum spinCxpLinkConfigurationStatusEnums
```

< This feature indicates the current and active Link configuration used by the Device.

Enumerator

CxpLinkConfigurationStatus_None	The Link configuration of the Device is unknown. Either the configuration operation has failed or there is nothing connected.
CxpLinkConfigurationStatus_Pending	The Device is in the process of configuring the Link. The Link cannot be used yet.
CxpLinkConfigurationStatus_CXP1_X1	1 Connection operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfigurationStatus_CXP2_X1	1 Connection operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfigurationStatus_CXP3_X1	1 Connection operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfigurationStatus_CXP5_X1	1 Connection operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfigurationStatus_CXP6_X1	1 Connection operating at CXP-6 speed (6.25 Gbps).
CxpLinkConfigurationStatus_CXP1_X2	2 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfigurationStatus_CXP2_X2	2 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfigurationStatus_CXP3_X2	2 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfigurationStatus_CXP5_X2	2 Connections operating at CXP-4 speed (5.00 Gbps).
CxpLinkConfigurationStatus_CXP6_X2	3 Connections operating at CXP-5 speed (6.25 Gbps).
CxpLinkConfigurationStatus_CXP1_X3	3 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfigurationStatus_CXP2_X3	3 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfigurationStatus_CXP3_X3	3 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfigurationStatus_CXP5_X3	3 Connections operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfigurationStatus_CXP6_X3	3 Connections operating at CXP-6 speed (6.25 Gbps).
CxpLinkConfigurationStatus_CXP1_X4	4 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfigurationStatus_CXP2_X4	4 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfigurationStatus_CXP3_X4	4 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfigurationStatus_CXP5_X4	4 Connections operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfigurationStatus_CXP6_X4	4 Connections operating at CXP-6 speed (6.25 Gbps).
CxpLinkConfigurationStatus_CXP1_X5	5 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfigurationStatus_CXP2_X5	5 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfigurationStatus_CXP3_X5	5 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfigurationStatus_CXP5_X5	5 Connections operating at CXP-5 speed (5.00 Gbps).

Enumerator

CxpLinkConfigurationStatus_CXP6_X5	5 Connections operating at CXP-6 speed (6.25 Gbps).
CxpLinkConfigurationStatus_CXP1_X6	6 Connections operating at CXP-1 speed (1.25 Gbps).
CxpLinkConfigurationStatus_CXP2_X6	6 Connections operating at CXP-2 speed (2.50 Gbps).
CxpLinkConfigurationStatus_CXP3_X6	6 Connections operating at CXP-3 speed (3.125 Gbps).
CxpLinkConfigurationStatus_CXP5_X6	6 Connections operating at CXP-5 speed (5.00 Gbps).
CxpLinkConfigurationStatus_CXP6_X6	6 Connections operating at CXP-6 speed (6.25 Gbps).
NUM_CXPLINKCONFIGURATIONSTATUS	

13.8.1.56 spinCxpPoCxpStatusEnums

```
enum spinCxpPoCxpStatusEnums
```

< Returns the Power over CoaXPress (PoCXP) status of the Device.

Enumerator

CxpPoCxpStatus_Auto	Normal automatic PoCXP operation.
CxpPoCxpStatus_Off	PoCXP is forced off.
CxpPoCxpStatus_Tripped	The Link has shut down because of an over-current trip.
NUM_CXPPOCXPSTATUS	

13.8.1.57 spinDecimationHorizontalModeEnums

```
enum spinDecimationHorizontalModeEnums
```

< The mode used to reduce the horizontal resolution when DecimationHorizontal is used. The current implementation only supports a single decimation mode: Discard. Average should be achieved via Binning.

Enumerator

DecimationHorizontalMode_Discard	The value of every Nth pixel is kept, others are discarded.
NUM_DECIMATIONHORIZONTALMODE	

13.8.1.58 spinDecimationSelectorEnums

```
enum spinDecimationSelectorEnums
```

< Selects which decimation layer is controlled by the DecimationHorizontal and DecimationVertical features.

Enumerator

DecimationSelector_All	The total amount of decimation to be performed on the captured image data.
DecimationSelector_Sensor	The portion of decimation to be performed on the sensor directly. Currently this is the only decimation layer available and hence is identical to the "All" layer. All decimation modification should therefore be done via the "All" layer only.
NUM_DECIMATIONSELECTOR	

13.8.1.59 spinDecimationVerticalModeEnums

```
enum spinDecimationVerticalModeEnums
```

< The mode used to reduce the vertical resolution when DecimationVertical is used. The current implementation only supports a single decimation mode: Discard. Average should be achieved via Binning.

Enumerator

DecimationVerticalMode_Discard	The value of every Nth pixel is kept, others are discarded.
NUM_DECIMATIONVERTICALMODE	

13.8.1.60 spinDefectCorrectionModeEnums

```
enum spinDefectCorrectionModeEnums
```

< Controls the method used for replacing defective pixels.

Enumerator

DefectCorrectionMode_Average	Pixels are replaced with the average of their neighbours. This is the normal mode of operation.
DefectCorrectionMode_Highlight	Pixels are replaced with the maximum pixel value (i.e., 255 for 8-bit images). Can be used for debugging the table.
DefectCorrectionMode_Zero	Pixels are replaced by the value zero. Can be used for testing the table.
NUM_DEFECTCORRECTIONMODE	

13.8.1.61 spinDeinterlacingEnums

```
enum spinDeinterlacingEnums
```

< Controls how the device performs de-interlacing.

Enumerator

Deinterlacing_Off	The device doesn't perform de-interlacing.
Deinterlacing_LineDuplication	The device performs de-interlacing by outputting each line of each field twice.
Deinterlacing_Weave	The device performs de-interlacing by interleaving the lines of all fields.
NUM_DEINTERLACING	

13.8.1.62 spinDeviceCharacterSetEnums

```
enum spinDeviceCharacterSetEnums
```

< Character set used by the strings of the device`s bootstrap registers.

Enumerator

DeviceCharacterSet_UTF8	
DeviceCharacterSet_ASCII	
NUM_DEVICECHARACTERSET	

13.8.1.63 spinDeviceClockSelectorEnums

```
enum spinDeviceClockSelectorEnums
```

< Selects the clock frequency to access from the device.

Enumerator

DeviceClockSelector_Sensor	Clock frequency of the image sensor of the camera.
DeviceClockSelector_SensorDigitization	Clock frequency of the camera A/D conversion stage.
DeviceClockSelector_CameraLink	Frequency of the Camera Link clock.
NUM_DEVICECLOCKSELECTOR	

13.8.1.64 spinDeviceConnectionStatusEnums

```
enum spinDeviceConnectionStatusEnums
```

< Indicates the status of the specified Connection.

Enumerator

DeviceConnectionStatus_Active	Connection is in use.
DeviceConnectionStatus_Inactive	Connection is not in use.
NUM_DEVICECONNECTIONSTATUS	

13.8.1.65 spinDeviceIndicatorModeEnums

```
enum spinDeviceIndicatorModeEnums
```

< Controls the LED behaviour: Inactive (off), Active (current status), or Error Status (off unless an error occurs).

Enumerator

DeviceIndicatorMode_Inactive	
DeviceIndicatorMode_Active	
DeviceIndicatorMode_ErrorStatus	
NUM_DEVICEINDICATORMODE	

13.8.1.66 spinDeviceLinkHeartbeatModeEnums

```
enum spinDeviceLinkHeartbeatModeEnums
```

< Activate or deactivate the Link's heartbeat.

Enumerator

DeviceLinkHeartbeatMode_On	Enables the Link heartbeat.
DeviceLinkHeartbeatMode_Off	Disables the Link heartbeat.
NUM_DEVICELINKHEARTBEATMODE	

13.8.1.67 spinDeviceLinkThroughputLimitModeEnums

```
enum spinDeviceLinkThroughputLimitModeEnums
```

< Controls if the DeviceLinkThroughputLimit is active. When disabled, lower level TL specific features are expected to control the throughput. When enabled, DeviceLinkThroughputLimit controls the overall throughput.

Enumerator

DeviceLinkThroughputLimitMode_On	Enables the DeviceLinkThroughputLimit feature.
DeviceLinkThroughputLimitMode_Off	Disables the DeviceLinkThroughputLimit feature.
NUM_DEVICELINKTHROUGHPUTLIMITMODE	

13.8.1.68 spinDevicePowerSupplySelectorEnums

```
enum spinDevicePowerSupplySelectorEnums
```

< Selects the power supply source to control or read.

Enumerator

DevicePowerSupplySelector_External	
NUM_DEVICEPOWERSUPPLYSELECTOR	

13.8.1.69 spinDeviceRegistersEndiannessEnums

```
enum spinDeviceRegistersEndiannessEnums
```

< Endianness of the registers of the device.

Enumerator

DeviceRegistersEndianness_Little	
DeviceRegistersEndianness_Big	
NUM_DEVICEREGISTERSENDIANNESS	

13.8.1.70 spinDeviceScanTypeEnums

```
enum spinDeviceScanTypeEnums
```

< Scan type of the sensor of the device.

Enumerator

DeviceScanType_Areascan	
NUM_DEVICESCANTYPE	

13.8.1.71 spinDeviceSerialPortBaudRateEnums

```
enum spinDeviceSerialPortBaudRateEnums
```

< This feature controls the baud rate used by the selected serial port.

Enumerator

DeviceSerialPortBaudRate_Baud9600	Serial port speed of 9600 baud.
DeviceSerialPortBaudRate_Baud19200	Serial port speed of 19200 baud.
DeviceSerialPortBaudRate_Baud38400	Serial port speed of 38400 baud.
DeviceSerialPortBaudRate_Baud57600	Serial port speed of 57600 baud.
DeviceSerialPortBaudRate_Baud115200	Serial port speed of 115200 baud.
DeviceSerialPortBaudRate_Baud230400	Serial port speed of 230400 baud.
DeviceSerialPortBaudRate_Baud460800	Serial port speed of 460800 baud.
DeviceSerialPortBaudRate_Baud921600	Serial port speed of 921600 baud.
NUM_DEVICESERIALPORTBAUDRATE	

13.8.1.72 spinDeviceSerialPortSelectorEnums

```
enum spinDeviceSerialPortSelectorEnums
```

< Selects which serial port of the device to control.

Enumerator

DeviceSerialPortSelector_CameraLink	Serial port associated to the Camera link connection.
NUM_DEVICESERIALPORTSELECTOR	

13.8.1.73 spinDeviceStreamChannelEndiannessEnums

```
enum spinDeviceStreamChannelEndiannessEnums
```

< Endianness of multi-byte pixel data for this stream.

Enumerator

DeviceStreamChannelEndianness_Big	Stream channel data is big Endian.
DeviceStreamChannelEndianness_Little	Stream channel data is little Endian.
NUM_DEVICESTREAMCHANNELENDIANNESS	

13.8.1.74 spinDeviceStreamChannelTypeEnums

```
enum spinDeviceStreamChannelTypeEnums
```

< Reports the type of the stream channel.

Enumerator

DeviceStreamChannelType_Transmitter	Data stream transmitter channel.
DeviceStreamChannelType_Receiver	Data stream receiver channel.
NUM_DEVICESTREAMCHANNELTYPE	

13.8.1.75 spinDeviceTapGeometryEnums

```
enum spinDeviceTapGeometryEnums
```

< This device tap geometry feature describes the geometrical properties characterizing the taps of a camera as presented at the output of the device.

Enumerator

DeviceTapGeometry_Geometry_1X_1Y	Geometry_1X_1Y
DeviceTapGeometry_Geometry_1X2_1Y	Geometry_1X2_1Y
DeviceTapGeometry_Geometry_1X2_1Y2	Geometry_1X2_1Y2
DeviceTapGeometry_Geometry_2X_1Y	Geometry_2X_1Y
DeviceTapGeometry_Geometry_2X_1Y2Geometry_2XE_1Y	Geometry_2X_1Y2Geometry_2XE_1Y
DeviceTapGeometry_Geometry_2XE_1Y2	Geometry_2XE_1Y2
DeviceTapGeometry_Geometry_2XM_1Y	Geometry_2XM_1Y
DeviceTapGeometry_Geometry_2XM_1Y2	Geometry_2XM_1Y2
DeviceTapGeometry_Geometry_1X_1Y2	Geometry_1X_1Y2
DeviceTapGeometry_Geometry_1X_2YE	Geometry_1X_2YE
DeviceTapGeometry_Geometry_1X3_1Y	Geometry_1X3_1Y
DeviceTapGeometry_Geometry_3X_1Y	Geometry_3X_1Y
DeviceTapGeometry_Geometry_1X	Geometry_1X
DeviceTapGeometry_Geometry_1X2	Geometry_1X2
DeviceTapGeometry_Geometry_2X	Geometry_2X
DeviceTapGeometry_Geometry_2XE	Geometry_2XE
DeviceTapGeometry_Geometry_2XM	Geometry_2XM
DeviceTapGeometry_Geometry_1X3	Geometry_1X3
DeviceTapGeometry_Geometry_3X	Geometry_3X
DeviceTapGeometry_Geometry_1X4_1Y	Geometry_1X4_1Y
DeviceTapGeometry_Geometry_4X_1Y	Geometry_4X_1Y
DeviceTapGeometry_Geometry_2X2_1Y	Geometry_2X2_1Y
DeviceTapGeometry_Geometry_2X2E_1YGeometry_2X2M_1Y	Geometry_2X2E_1YGeometry_2X2M_1Y
DeviceTapGeometry_Geometry_1X2_2YE	Geometry_1X2_2YE
DeviceTapGeometry_Geometry_2X_2YE	Geometry_2X_2YE
DeviceTapGeometry_Geometry_2XE_2YE	Geometry_2XE_2YE
DeviceTapGeometry_Geometry_2XM_2YE	Geometry_2XM_2YE
DeviceTapGeometry_Geometry_1X4	Geometry_1X4
DeviceTapGeometry_Geometry_4X	Geometry_4X
DeviceTapGeometry_Geometry_2X2	Geometry_2X2
DeviceTapGeometry_Geometry_2X2E	Geometry_2X2E

Enumerator

DeviceTapGeometry_Geometry_2X2M	Geometry_2X2M
DeviceTapGeometry_Geometry_1X8_1Y	Geometry_1X8_1Y
DeviceTapGeometry_Geometry_8X_1Y	Geometry_8X_1Y
DeviceTapGeometry_Geometry_4X2_1Y	Geometry_4X2_1Y
DeviceTapGeometry_Geometry_2X2E_2YE	Geometry_2X2E_2YE
DeviceTapGeometry_Geometry_1X8	Geometry_1X8
DeviceTapGeometry_Geometry_8X	Geometry_8X
DeviceTapGeometry_Geometry_4X2	Geometry_4X2
DeviceTapGeometry_Geometry_4X2E	Geometry_4X2E
DeviceTapGeometry_Geometry_4X2E_1Y	Geometry_4X2E_1Y
DeviceTapGeometry_Geometry_1X10_1Y	Geometry_1X10_1Y
DeviceTapGeometry_Geometry_10X_1Y	Geometry_10X_1Y
DeviceTapGeometry_Geometry_1X10	Geometry_1X10
DeviceTapGeometry_Geometry_10X	Geometry_10X
NUM_DEVICETAPGEOMETRY	

13.8.1.76 spinDeviceTemperatureSelectorEnums

```
enum spinDeviceTemperatureSelectorEnums
```

< Selects the location within the device, where the temperature will be measured.

Enumerator

DeviceTemperatureSelector_Sensor	
NUM_DEVICETEMPERATURESELECTOR	

13.8.1.77 spinDeviceTLTypeEnums

```
enum spinDeviceTLTypeEnums
```

< Transport Layer type of the device.

Enumerator

DeviceTLType_GigEVision	
DeviceTLType_CameraLink	
DeviceTLType_CameraLinkHS	
DeviceTLType_CoaXPress	
DeviceTLType_USB3Vision	
DeviceTLType_Custom	
NUM_DEVICETLTYPE	

13.8.1.78 spinDeviceTypeEnums

```
enum spinDeviceTypeEnums
```

< Returns the device type.

Enumerator

DeviceType_Transmitter	Data stream transmitter device.
DeviceType_Receiver	Data stream receiver device.
DeviceType_Transceiver	Data stream receiver and transmitter device.
DeviceType_Peripheral	Controllable device (with no data stream handling).
NUM_DEVICETYPE	

13.8.1.79 spinEncoderModeEnums

```
enum spinEncoderModeEnums
```

< Selects if the count of encoder uses FourPhase mode with jitter filtering or the HighResolution mode without jitter filtering.

Enumerator

EncoderMode_FourPhase	The counter increments or decrements 1 for every full quadrature cycle with jitter filtering.
EncoderMode_HighResolution	The counter increments or decrements every quadrature phase for high resolution counting, but without jitter filtering.
NUM_ENCODERMODE	

13.8.1.80 spinEncoderOutputModeEnums

```
enum spinEncoderOutputModeEnums
```

< Selects the conditions for the Encoder interface to generate a valid Encoder output signal.

Enumerator

EncoderOutputMode_Off	No output pulse are generated.
EncoderOutputMode_PositionUp	Output pulses are generated at all new positions in the positive direction. If the encoder reverses no output pulse are generated until it has again passed the position where the reversal started.

Enumerator

EncoderOutputMode_PositionDown	Output pulses are generated at all new positions in the negative direction. If the encoder reverses no output pulse are generated until it has again passed the position where the reversal started.
EncoderOutputMode_DirectionUp	Output pulses are generated at all position increments in the positive direction while ignoring negative direction motion.
EncoderOutputMode_DirectionDown	Output pulses are generated at all position increments in the negative direction while ignoring positive direction motion.
EncoderOutputMode_Motion	Output pulses are generated at all motion increments in both directions.
NUM_ENCODEROUTPUTMODE	

13.8.1.81 spinEncoderResetActivationEnums

```
enum spinEncoderResetActivationEnums
```

< Selects the Activation mode of the Encoder Reset Source signal.

Enumerator

EncoderResetActivation_RisingEdge	Resets the Encoder on the Rising Edge of the signal.
EncoderResetActivation_FallingEdge	Resets the Encoder on the Falling Edge of the signal.
EncoderResetActivation_AnyEdge	Resets the Encoder on the Falling or rising Edge of the selected signal.
EncoderResetActivation_LevelHigh	Resets the Encoder as long as the selected signal level is High.
EncoderResetActivation_LevelLow	Resets the Encoder as long as the selected signal level is Low.
NUM_ENCODERRESETACTIVATION	

13.8.1.82 spinEncoderResetSourceEnums

```
enum spinEncoderResetSourceEnums
```

< Selects the signals that will be the source to reset the Encoder.

Enumerator

EncoderResetSource_Off	Disable the Encoder Reset trigger.
EncoderResetSource_AcquisitionTrigger	Resets with the reception of the Acquisition Trigger.
EncoderResetSource_AcquisitionStart	Resets with the reception of the Acquisition Start.
EncoderResetSource_AcquisitionEnd	Resets with the reception of the Acquisition End.
EncoderResetSource_FrameTrigger	Resets with the reception of the Frame Start Trigger.
EncoderResetSource_FrameStart	Resets with the reception of the Frame Start.
EncoderResetSource_FrameEnd	Resets with the reception of the Frame End.
EncoderResetSource_ExposureStart	Resets with the reception of the Exposure Start.

Enumerator

EncoderResetSource_ExposureEnd	Resets with the reception of the Exposure End.
EncoderResetSource_Line0	Resets by the chosen I/O Line.
EncoderResetSource_Line1	Resets by the chosen I/O Line.
EncoderResetSource_Line2	Resets by the chosen I/O Line.
EncoderResetSource_Counter0Start	Resets with the reception of the Counter Start.
EncoderResetSource_Counter1Start	Resets with the reception of the Counter Start.
EncoderResetSource_Counter2Start	Resets with the reception of the Counter Start.
EncoderResetSource_Counter0End	Resets with the reception of the Counter End.
EncoderResetSource_Counter1End	Resets with the reception of the Counter End.
EncoderResetSource_Counter2End	Resets with the reception of the Counter End.
EncoderResetSource_Timer0Start	Resets with the reception of the Timer Start.
EncoderResetSource_Timer1Start	Resets with the reception of the Timer Start.
EncoderResetSource_Timer2Start	Resets with the reception of the Timer Start.
EncoderResetSource_Timer0End	Resets with the reception of the Timer End.
EncoderResetSource_Timer1End	Resets with the reception of the Timer End.
EncoderResetSource_Timer2End	Resets with the reception of the Timer End.
EncoderResetSource_UserOutput0	Resets by the chosen User Output bit.
EncoderResetSource_UserOutput1	Resets by the chosen User Output bit.
EncoderResetSource_UserOutput2	Resets by the chosen User Output bit.
EncoderResetSource_SoftwareSignal0	Resets on the reception of the Software Signal.
EncoderResetSource_SoftwareSignal1	Resets on the reception of the Software Signal.
EncoderResetSource_SoftwareSignal2	Resets on the reception of the Software Signal.
EncoderResetSource_Action0	Resets on assertions of the chosen action signal (Broadcasted signal on the transport layer).
EncoderResetSource_Action1	Resets on assertions of the chosen action signal (Broadcasted signal on the transport layer).
EncoderResetSource_Action2	Resets on assertions of the chosen action signal (Broadcasted signal on the transport layer).
EncoderResetSource_LinkTrigger0	Resets on the reception of the chosen Link Trigger (received from the transport layer).
EncoderResetSource_LinkTrigger1	Resets on the reception of the chosen Link Trigger (received from the transport layer).
EncoderResetSource_LinkTrigger2	Resets on the reception of the chosen Link Trigger (received from the transport layer).
NUM_ENCODERRESETSOURCE	

13.8.1.83 spinEncoderSelectorEnums

```
enum spinEncoderSelectorEnums
```

< Selects which Encoder to configure.

Enumerator

EncoderSelector_Encoder0	Selects Encoder 0.
EncoderSelector_Encoder1	Selects Encoder 1.
EncoderSelector_Encoder2	Selects Encoder 2.
NUM_ENCODERSELECTOR	

13.8.1.84 spinEncoderSourceAEnums

```
enum spinEncoderSourceAEnums
```

< Selects the signal which will be the source of the A input of the Encoder.

Enumerator

EncoderSourceA_Off	Counter is stopped.
EncoderSourceA_Line0	Encoder Forward input is taken from the chosen I/O Line.
EncoderSourceA_Line1	Encoder Forward input is taken from the chosen I/O Line.
EncoderSourceA_Line2	Encoder Forward input is taken from the chosen I/O Line.
NUM_ENCODERSOURCEA	

13.8.1.85 spinEncoderSourceBEnums

```
enum spinEncoderSourceBEnums
```

< Selects the signal which will be the source of the B input of the Encoder.

Enumerator

EncoderSourceB_Off	Counter is stopped.
EncoderSourceB_Line0	Encoder Reverse input is taken from the chosen I/O Line..
EncoderSourceB_Line1	Encoder Reverse input is taken from the chosen I/O Line..
EncoderSourceB_Line2	Encoder Reverse input is taken from the chosen I/O Line..
NUM_ENCODERSOURCEB	

13.8.1.86 spinEncoderStatusEnums

```
enum spinEncoderStatusEnums
```

< Returns the motion status of the encoder.

Enumerator

EncoderStatus_EncoderUp	The encoder counter last incremented.
EncoderStatus_EncoderDown	The encoder counter last decremented.
EncoderStatus_EncoderIdle	The encoder is not active.
EncoderStatus_EncoderStatic	No motion within the EncoderTimeout time.
NUM_ENCODERSTATUS	

13.8.1.87 spinEventNotificationEnums

```
enum spinEventNotificationEnums
```

< Enables/Disables the selected event.

Enumerator

EventNotification_On	
EventNotification_Off	
NUM_EVENTNOTIFICATION	

13.8.1.88 spinEventSelectorEnums

```
enum spinEventSelectorEnums
```

< Selects which Event to enable or disable.

Enumerator

EventSelector_Error	
EventSelector_ExposureEnd	
EventSelector_SerialPortReceive	
NUM_EVENTSELECTOR	

13.8.1.89 spinExposureActiveModeEnums

```
enum spinExposureActiveModeEnums
```

< Control sensor active exposure mode.

Enumerator

ExposureActiveMode_Line1	
ExposureActiveMode_AnyPixels	
ExposureActiveMode_AllPixels	
NUM_EXPOSUREACTIVEMODE	

13.8.1.90 spinExposureAutoEnums

```
enum spinExposureAutoEnums
```

< Sets the automatic exposure mode

Enumerator

ExposureAuto_Off	Exposure time is manually controlled using ExposureTime
ExposureAuto_Once	Exposure time is adapted once by the device. Once it has converged, it returns to the Off state.
ExposureAuto_Continuous	Exposure time is constantly adapted by the device to maximize the dynamic range.
NUM_EXPOSUREAUTO	

13.8.1.91 spinExposureModeEnums

```
enum spinExposureModeEnums
```

< Sets the operation mode of the Exposure.

Enumerator

ExposureMode_Timed	Timed exposure. The exposure time is set using the ExposureTime or ExposureAuto features and the exposure starts with the FrameStart or LineStart.
ExposureMode_TriggerWidth	Uses the width of the current Frame trigger signal pulse to control the exposure time.
NUM_EXPOSUREMODE	

13.8.1.92 spinExposureTimeModeEnums

```
enum spinExposureTimeModeEnums
```

< Sets the configuration mode of the ExposureTime feature.

Enumerator

ExposureTimeMode_Common	The exposure time is common to all the color components. The common ExposureTime value to use can be set selecting it with ExposureTimeSelector[Common].
ExposureTimeMode_Individual	The exposure time is individual for each color component. Each individual ExposureTime values to use can be set by selecting them with ExposureTimeSelector.
NUM_EXPOSURETIMEMODE	

13.8.1.93 spinExposureTimeSelectorEnums

```
enum spinExposureTimeSelectorEnums
```

< Selects which exposure time is controlled by the ExposureTime feature. This allows for independent control over the exposure components.

Enumerator

ExposureTimeSelector_Common	Selects the common ExposureTime.
ExposureTimeSelector_Red	Selects the red common ExposureTime.
ExposureTimeSelector_Green	Selects the green ExposureTime.
ExposureTimeSelector_Blue	Selects the blue ExposureTime.
ExposureTimeSelector_Cyan	Selects the cyan common ExposureTime.
ExposureTimeSelector_Magenta	Selects the magenta ExposureTime.
ExposureTimeSelector_Yellow	Selects the yellow ExposureTime.
ExposureTimeSelector_Infrared	Selects the infrared ExposureTime.
ExposureTimeSelector_Ultraviolet	Selects the ultraviolet ExposureTime.
ExposureTimeSelector_Stage1	Selects the first stage ExposureTime.
ExposureTimeSelector_Stage2	Selects the second stage ExposureTime.
NUM_EXPOSURETIMESELECTOR	

13.8.1.94 spinFileOpenModeEnums

```
enum spinFileOpenModeEnums
```

< The mode of the file when it is opened. The file can be opened for reading, writting or both. This must be set before opening the file.

Enumerator

FileOpenMode_Read	
FileOpenMode_Write	
FileOpenMode_ReadWrite	
NUM_FILEOPENMODE	

13.8.1.95 spinFileOperationSelectorEnums

```
enum spinFileOperationSelectorEnums
```

< Sets operation to execute on the selected file when the execute command is given.

Enumerator

FileOperationSelector_Open	
FileOperationSelector_Close	
FileOperationSelector_Read	
FileOperationSelector_Write	
FileOperationSelector_Delete	
NUM_FILEOPERATIONSELECTOR	

13.8.1.96 spinFileOperationStatusEnums

```
enum spinFileOperationStatusEnums
```

< Represents the file operation execution status.

Enumerator

FileOperationStatus_Success	File Operation was sucessful.
FileOperationStatus_Failure	File Operation failed.
FileOperationStatus_Overflow	An overflow occurred while executing the File Operation.
NUM_FILEOPERATIONSTATUS	

13.8.1.97 spinFileSelectorEnums

```
enum spinFileSelectorEnums
```

< Selects which file is being operated on. This must be set before performing any file operations.

Enumerator

FileSelector_UserSetDefault	
FileSelector_UserSet0	
FileSelector_UserSet1	
FileSelector_UserFile1	
FileSelector_SerialPort0	
NUM_FILESELECTOR	

13.8.1.98 spinGainAutoBalanceEnums

```
enum spinGainAutoBalanceEnums
```

< Sets the mode for automatic gain balancing between the sensor color channels or taps. The gain coefficients of each channel or tap are adjusted so they are matched.

Enumerator

GainAutoBalance_Off	Gain tap balancing is user controlled using Gain .
GainAutoBalance_Once	Gain tap balancing is automatically adjusted once by the device. Once it has converged, it automatically returns to the Off state.
GainAutoBalance_Continuous	Gain tap balancing is constantly adjusted by the device.
NUM_GAINAUTOBALANCE	

13.8.1.99 spinGainAutoEnums

```
enum spinGainAutoEnums
```

< Sets the automatic gain mode. Set to Off for manual control. Set to Once for a single automatic adjustment then return to Off. Set to Continuous for constant adjustment. In automatic modes, the camera adjusts the gain to maximize the dynamic range.

Enumerator

GainAuto_Off	Gain is manually controlled
GainAuto_Once	Gain is adapted once by the device. Once it has converged, it returns to the Off state.
GainAuto_Continuous	Gain is constantly adapted by the device to maximize the dynamic range.
NUM_GAINAUTO	

13.8.1.100 spinGainSelectorEnums

```
enum spinGainSelectorEnums
```

< Selects which gain to control. The All selection is a total amplification across all channels (or taps).

Enumerator

GainSelector_All	
NUM_GAINSELECTOR	

13.8.1.101 spinGevCCPEnums

```
enum spinGevCCPEnums
```

< Controls the device access privilege of an application.

Enumerator

GevCCP_OpenAccess	
GevCCP_ExclusiveAccess	
GevCCP_ControlAccess	
NUM_GEVCCP	

13.8.1.102 spinGevCurrentPhysicalLinkConfigurationEnums

```
enum spinGevCurrentPhysicalLinkConfigurationEnums
```

< Indicates the current physical link configuration of the device.

Enumerator

GevCurrentPhysicalLinkConfiguration_SingleLink	Single Link
GevCurrentPhysicalLinkConfiguration_MultiLink	Multi Link
GevCurrentPhysicalLinkConfiguration_StaticLAG	Static LAG
GevCurrentPhysicalLinkConfiguration_DynamicLAG	Dynamic LAG
NUM_GEVCURRENTPHYSICALLINKCONFIGURATION	

13.8.1.103 spinGevGVCPExtendedStatusCodesSelectorEnums

```
enum spinGevGVCPExtendedStatusCodesSelectorEnums
```

< Selects the GigE Vision version to control extended status codes for.

Enumerator

GevGVCPExtendedStatusCodesSelector_Version1_1	Version 1 1
GevGVCPExtendedStatusCodesSelector_Version2_0	Version 2 0
NUM_GEVGVCPEXTENDEDSTATUSCODESSELECTOR	

13.8.1.104 spinGevGVSPExtendedIDModeEnums

```
enum spinGevGVSPExtendedIDModeEnums
```

< Enables the extended IDs mode.

Enumerator

GevGVSPExtendedIDMode_Off	Off
GevGVSPExtendedIDMode_On	On
NUM_GEVGVSPEXTENDEDIDMODE	

13.8.1.105 spinGevIEEE1588ClockAccuracyEnums

```
enum spinGevIEEE1588ClockAccuracyEnums
```

< Indicates the expected accuracy of the device clock when it is the grandmaster, or in the event it becomes the grandmaster.

Enumerator

GevIEEE1588ClockAccuracy_Unknown	Unknown Accuracy
NUM_GEVIEEE1588CLOCKACCURACY	

13.8.1.106 spinGevIEEE1588ModeEnums

```
enum spinGevIEEE1588ModeEnums
```

< Provides the mode of the IEEE 1588 clock.

Enumerator

GevIEEE1588Mode_Auto	Automatic
GevIEEE1588Mode_SlaveOnly	Slave Only
NUM_GEVIEEE1588MODE	

13.8.1.107 spinGevIEEE1588StatusEnums

```
enum spinGevIEEE1588StatusEnums
```

< Provides the status of the IEEE 1588 clock.

Enumerator

GevIEEE1588Status_Initializing	Initializing
GevIEEE1588Status_Faulty	Faulty
GevIEEE1588Status_Disabled	Disabled

Enumerator

GevIEEE1588Status_Listening	Listening
GevIEEE1588Status_PreMaster	Pre Master
GevIEEE1588Status_Master	Master
GevIEEE1588Status_Passive	Passive
GevIEEE1588Status_Uncalibrated	Uncalibrated
GevIEEE1588Status_Slave	Slave
NUM_GEVIEEE1588STATUS	

13.8.1.108 spinGevIPConfigurationStatusEnums

```
enum spinGevIPConfigurationStatusEnums
```

< Reports the current IP configuration status.

Enumerator

GevIPConfigurationStatus_None	None
GevIPConfigurationStatus_PersistentIP	Persistent IP
GevIPConfigurationStatus_DHCP	DHCP
GevIPConfigurationStatus_LLA	LLA
GevIPConfigurationStatus_ForceIP	Force IP
NUM_GEVIPCONFIGURATIONSTATUS	

13.8.1.109 spinGevPhysicalLinkConfigurationEnums

```
enum spinGevPhysicalLinkConfigurationEnums
```

< Controls the principal physical link configuration to use on next restart/power-up of the device.

Enumerator

GevPhysicalLinkConfiguration_SingleLink	Single Link
GevPhysicalLinkConfiguration_MultiLink	Multi Link
GevPhysicalLinkConfiguration_StaticLAG	Static LAG
GevPhysicalLinkConfiguration_DynamicLAG	Dynamic LAG
NUM_GEVPHYSICALLINKCONFIGURATION	

13.8.1.110 spinGevSupportedOptionSelectorEnums

```
enum spinGevSupportedOptionSelectorEnums
```

< Selects the GEV option to interrogate for existing support.

Enumerator

GevSupportedOptionSelector_UserDefinedName	
GevSupportedOptionSelector_SerialNumber	
GevSupportedOptionSelector_HeartbeatDisable	
GevSupportedOptionSelector_LinkSpeed	
GevSupportedOptionSelector_CCPApplicationSocket	
GevSupportedOptionSelector_ManifestTable	
GevSupportedOptionSelector_TestData	
GevSupportedOptionSelector_DiscoveryAckDelay	
GevSupportedOptionSelector_DiscoveryAckDelayWritable	
GevSupportedOptionSelector_ExtendedStatusCodes	
GevSupportedOptionSelector_Action	
GevSupportedOptionSelector_PendingAck	
GevSupportedOptionSelector_EventData	
GevSupportedOptionSelector_Event	
GevSupportedOptionSelector_PacketResend	
GevSupportedOptionSelector_WriteMem	
GevSupportedOptionSelector_CommandsConcatenation	
GevSupportedOptionSelector_IPConfigurationLLA	
GevSupportedOptionSelector_IPConfigurationDHCP	
GevSupportedOptionSelector_IPConfigurationPersistentIP	
GevSupportedOptionSelector_StreamChannelSourceSocket	
GevSupportedOptionSelector_MessageChannelSourceSocket	
NUM_GEVSUPPORTEDOPTIONSELECTOR	

13.8.1.111 spinImageComponentSelectorEnums

```
enum spinImageComponentSelectorEnums
```

< Selects a component to activate data streaming from.

Enumerator

ImageComponentSelector_Intensity	The acquisition of intensity of the reflected light is controlled.
ImageComponentSelector_Color	The acquisition of color of the reflected light is controlled
ImageComponentSelector_Infrared	The acquisition of non-visible infrared light is controlled.
ImageComponentSelector_Ultraviolet	The acquisition of non-visible ultraviolet light is controlled.
ImageComponentSelector_Range	The acquisition of range (distance) data is controlled. The data produced may be only range (2.5D) or a point cloud 3D coordinates depending on the Scan3dControl.

Enumerator

ImageComponentSelector_Disparity	The acquisition of stereo camera disparity data is controlled. Disparity is a more specific range format approximately inversely proportional to distance. Disparity is typically given in pixel units.
ImageComponentSelector_Confidence	The acquisition of confidence map of the acquired image is controlled. Confidence data may be binary (0 - invalid) or an integer where 0 is invalid and increasing value is increased confidence in the data in the corresponding pixel. If floating point representation is used the confidence image is normalized to the range [0,1], for integer representation the maximum possible integer represents maximum confidence.
ImageComponentSelector_Scatter	The acquisition of data measuring how much light is scattered around the reflected light. In processing this is used as an additional intensity image, often together with the standard intensity.
NUM_IMAGECOMPONENTSELECTOR	

13.8.1.112 spinImageCompressionJPEGFormatOptionEnums

```
enum spinImageCompressionJPEGFormatOptionEnums
```

< When JPEG is selected as the compression format, a device might optionally offer better control over JPEG-specific options through this feature.

Enumerator

ImageCompressionJPEGFormatOption_Lossless	Selects lossless JPEG compression based on a predictive coding model.
ImageCompressionJPEGFormatOption_Baseline↩ Standard	Indicates this is a baseline sequential (single-scan) DCT-based JPEG.
ImageCompressionJPEGFormatOption_Baseline↩ Optimized	Provides optimized color and slightly better compression than baseline standard by using custom Huffman tables optimized after statistical analysis of the image content.
ImageCompressionJPEGFormatOption_Progressive	Indicates this is a progressive (multi-scan) DCT-based JPEG.
NUM_↩ IMAGECOMPRESSIONJPEGFORMATOPTION	

13.8.1.113 spinImageCompressionModeEnums

```
enum spinImageCompressionModeEnums
```

<

Enumerator

ImageCompressionMode_Off	
ImageCompressionMode_Lossless	
NUM_IMAGECOMPRESSIONMODE	

13.8.1.114 spinImageCompressionRateOptionEnums

```
enum spinImageCompressionRateOptionEnums
```

< Two rate controlling options are offered: fixed bit rate or fixed quality. The exact implementation to achieve one or the other is vendor-specific.

Enumerator

ImageCompressionRateOption_FixBitrate	Output stream follows a constant bit rate. Allows easy bandwidth management on the link.
ImageCompressionRateOption_FixQuality	Output stream has a constant image quality. Can be used when image processing algorithms are sensitive to image degradation caused by excessive data compression.
NUM_IMAGECOMPRESSIONRATEOPTION	

13.8.1.115 spinLineFormatEnums

```
enum spinLineFormatEnums
```

< Displays the current electrical format of the selected physical input or output Line.

Enumerator

LineFormat_NoConnect	
LineFormat_TriState	
LineFormat_TTL	
LineFormat_LVDS	
LineFormat_RS422	
LineFormat_OptoCoupled	
LineFormat_OpenDrain	
NUM_LINEFORMAT	

13.8.1.116 spinLineInputFilterSelectorEnums

```
enum spinLineInputFilterSelectorEnums
```

< Selects the kind of input filter to configure: Deglitch or Debounce.

Enumerator

LineInputFilterSelector_Deglitch	
LineInputFilterSelector_Debounce	
NUM_LINEINPUTFILTERSELECTOR	

13.8.1.117 spinLineModeEnums

```
enum spinLineModeEnums
```

< Controls if the physical Line is used to Input or Output a signal.

Enumerator

LineMode_Input	
LineMode_Output	
NUM_LINEMODE	

13.8.1.118 spinLineSelectorEnums

```
enum spinLineSelectorEnums
```

< Selects the physical line (or pin) of the external device connector to configure

Enumerator

LineSelector_Line0	
LineSelector_Line1	
LineSelector_Line2	
LineSelector_Line3	
NUM_LINESELECTOR	

13.8.1.119 spinLineSourceEnums

```
enum spinLineSourceEnums
```

< Selects which internal acquisition or I/O source signal to output on the selected line. LineMode must be Output.

Enumerator

LineSource_Off	
LineSource_Line0	
LineSource_Line1	
LineSource_Line2	
LineSource_Line3	
LineSource_UserOutput0	
LineSource_UserOutput1	
LineSource_UserOutput2	
LineSource_UserOutput3	
LineSource_Counter0Active	
LineSource_Counter1Active	
LineSource_LogicBlock0	
LineSource_LogicBlock1	
LineSource_ExposureActive	
LineSource_FrameTriggerWait	
LineSource_SerialPort0	
LineSource_PPSSignal	
LineSource_AllPixel	
LineSource_AnyPixel	
NUM_LINESOURCE	

13.8.1.120 spinLogicBlockLUTInputActivationEnums

```
enum spinLogicBlockLUTInputActivationEnums
```

< Selects the activation mode of the Logic Input Source signal.

Enumerator

LogicBlockLUTInputActivation_LevelLow	
LogicBlockLUTInputActivation_LevelHigh	
LogicBlockLUTInputActivation_FallingEdge	
LogicBlockLUTInputActivation_RisingEdge	
LogicBlockLUTInputActivation_AnyEdge	
NUM_LOGICBLOCKLUTINPUTACTIVATION	

13.8.1.121 spinLogicBlockLUTInputSelectorEnums

```
enum spinLogicBlockLUTInputSelectorEnums
```

< Controls which LogicBlockLUT Input Source & Activation to access.

Enumerator

LogicBlockLUTInputSelector_Input0	
LogicBlockLUTInputSelector_Input1	
LogicBlockLUTInputSelector_Input2	
LogicBlockLUTInputSelector_Input3	
NUM_LOGICBLOCKLUTINPUTSELECTOR	

13.8.1.122 spinLogicBlockLUTInputSourceEnums

```
enum spinLogicBlockLUTInputSourceEnums
```

< Selects the source for the input into the Logic LUT.

Enumerator

LogicBlockLUTInputSource_Zero	Zero
LogicBlockLUTInputSource_Line0	Line0
LogicBlockLUTInputSource_Line1	Line1
LogicBlockLUTInputSource_Line2	Line2
LogicBlockLUTInputSource_Line3	Line3
LogicBlockLUTInputSource_UserOutput0	UserOutput0
LogicBlockLUTInputSource_UserOutput1	UserOutput1
LogicBlockLUTInputSource_UserOutput2	UserOutput2
LogicBlockLUTInputSource_UserOutput3	UserOutput3
LogicBlockLUTInputSource_Counter0Start	Counter0Start
LogicBlockLUTInputSource_Counter1Start	Counter1Start
LogicBlockLUTInputSource_Counter0End	Counter0End
LogicBlockLUTInputSource_Counter1End	Counter1End
LogicBlockLUTInputSource_LogicBlock0	LogicBlock0
LogicBlockLUTInputSource_LogicBlock1	LogicBlock1
LogicBlockLUTInputSource_ExposureStart	ExposureStart
LogicBlockLUTInputSource_ExposureEnd	ExposureEnd
LogicBlockLUTInputSource_FrameTriggerWait	FrameTriggerWait
LogicBlockLUTInputSource_AcquisitionActive	AcquisitionActive
NUM_LOGICBLOCKLUTINPUTSOURCE	

13.8.1.123 spinLogicBlockLUTSelectorEnums

```
enum spinLogicBlockLUTSelectorEnums
```

< Selects which LogicBlock LUT to configure

Enumerator

LogicBlockLUTSelector_Value	
LogicBlockLUTSelector_Enable	
NUM_LOGICBLOCKLUTSELECTOR	

13.8.1.124 spinLogicBlockSelectorEnums

```
enum spinLogicBlockSelectorEnums
```

< Selects which LogicBlock to configure

Enumerator

LogicBlockSelector_LogicBlock0	
LogicBlockSelector_LogicBlock1	
NUM_LOGICBLOCKSELECTOR	

13.8.1.125 spinLUTSelectorEnums

```
enum spinLUTSelectorEnums
```

The enum definitions for camera nodes.

< Selects which LUT to control.

Enumerator

LUTSelector_LUT1	This LUT is for re-mapping pixels of all formats (mono, Bayer, red, green and blue).
NUM_LUTSELECTOR	

13.8.1.126 spinPixelColorFilterEnums

```
enum spinPixelColorFilterEnums
```

< Type of color filter that is applied to the image. Only applies to Bayer pixel formats. All others have no color filter.

Enumerator

PixelColorFilter_None	No color filter.
PixelColorFilter_BayerRG	Bayer Red Green filter.

Enumerator

PixelColorFilter_BayerGB	Bayer Green Blue filter.
PixelColorFilter_BayerGR	Bayer Green Red filter.
PixelColorFilter_BayerBG	Bayer Blue Green filter.
NUM_PIXELCOLORFILTER	

13.8.1.127 spinPixelFormatEnums

```
enum spinPixelFormatEnums
```

< Format of the pixel provided by the camera.

Enumerator

PixelFormat_Mono8	
PixelFormat_Mono16	
PixelFormat_RGB8Packed	
PixelFormat_BayerGR8	
PixelFormat_BayerRG8	
PixelFormat_BayerGB8	
PixelFormat_BayerBG8	
PixelFormat_BayerGR16	
PixelFormat_BayerRG16	
PixelFormat_BayerGB16	
PixelFormat_BayerBG16	
PixelFormat_Mono12Packed	
PixelFormat_BayerGR12Packed	
PixelFormat_BayerRG12Packed	
PixelFormat_BayerGB12Packed	
PixelFormat_BayerBG12Packed	
PixelFormat_YUV411Packed	
PixelFormat_YUV422Packed	
PixelFormat_YUV444Packed	
PixelFormat_Mono12p	
PixelFormat_BayerGR12p	
PixelFormat_BayerRG12p	
PixelFormat_BayerGB12p	
PixelFormat_BayerBG12p	
PixelFormat_YCbCr8	
PixelFormat_YCbCr422_8	
PixelFormat_YCbCr411_8	
PixelFormat_BGR8	
PixelFormat_BGRa8	
PixelFormat_Mono10Packed	
PixelFormat_BayerGR10Packed	
PixelFormat_BayerRG10Packed	

Enumerator

PixelFormat_BayerGB10Packed	
PixelFormat_BayerBG10Packed	
PixelFormat_Mono10p	
PixelFormat_BayerGR10p	
PixelFormat_BayerRG10p	
PixelFormat_BayerGB10p	
PixelFormat_BayerBG10p	
PixelFormat_Mono1p	Monochrome 1-bit packed
PixelFormat_Mono2p	Monochrome 2-bit packed
PixelFormat_Mono4p	Monochrome 4-bit packed
PixelFormat_Mono8s	Monochrome 8-bit signed
PixelFormat_Mono10	Monochrome 10-bit unpacked
PixelFormat_Mono12	Monochrome 12-bit unpacked
PixelFormat_Mono14	Monochrome 14-bit unpacked
PixelFormat_Mono16s	Monochrome 16-bit signed
PixelFormat_Mono32f	Monochrome 32-bit float
PixelFormat_BayerBG10	Bayer Blue-Green 10-bit unpacked
PixelFormat_BayerBG12	Bayer Blue-Green 12-bit unpacked
PixelFormat_BayerGB10	Bayer Green-Blue 10-bit unpacked
PixelFormat_BayerGB12	Bayer Green-Blue 12-bit unpacked
PixelFormat_BayerGR10	Bayer Green-Red 10-bit unpacked
PixelFormat_BayerGR12	Bayer Green-Red 12-bit unpacked
PixelFormat_BayerRG10	Bayer Red-Green 10-bit unpacked
PixelFormat_BayerRG12	Bayer Red-Green 12-bit unpacked
PixelFormat_RGBa8	Red-Green-Blue-alpha 8-bit
PixelFormat_RGBa10	Red-Green-Blue-alpha 10-bit unpacked
PixelFormat_RGBa10p	Red-Green-Blue-alpha 10-bit packed
PixelFormat_RGBa12	Red-Green-Blue-alpha 12-bit unpacked
PixelFormat_RGBa12p	Red-Green-Blue-alpha 12-bit packed
PixelFormat_RGBa14	Red-Green-Blue-alpha 14-bit unpacked
PixelFormat_RGBa16	Red-Green-Blue-alpha 16-bit
PixelFormat_RGB8	Red-Green-Blue 8-bit
PixelFormat_RGB8_Planar	Red-Green-Blue 8-bit planar
PixelFormat_RGB10	Red-Green-Blue 10-bit unpacked
PixelFormat_RGB10_Planar	Red-Green-Blue 10-bit unpacked planar
PixelFormat_RGB10p	Red-Green-Blue 10-bit packed
PixelFormat_RGB10p32	Red-Green-Blue 10-bit packed into 32-bit
PixelFormat_RGB12	Red-Green-Blue 12-bit unpacked
PixelFormat_RGB12_Planar	Red-Green-Blue 12-bit unpacked planar
PixelFormat_RGB12p	Red-Green-Blue 12-bit packed
PixelFormat_RGB14	Red-Green-Blue 14-bit unpacked
PixelFormat_RGB16	Red-Green-Blue 16-bit
PixelFormat_RGB16s	Red-Green-Blue 16-bit signed
PixelFormat_RGB32f	Red-Green-Blue 32-bit float
PixelFormat_RGB16_Planar	Red-Green-Blue 16-bit planar

Enumerator

PixelFormat_RGB565p	Red-Green-Blue 5/6/5-bit packed
PixelFormat_BGRa10	Blue-Green-Red-alpha 10-bit unpacked
PixelFormat_BGRa10p	Blue-Green-Red-alpha 10-bit packed
PixelFormat_BGRa12	Blue-Green-Red-alpha 12-bit unpacked
PixelFormat_BGRa12p	Blue-Green-Red-alpha 12-bit packed
PixelFormat_BGRa14	Blue-Green-Red-alpha 14-bit unpacked
PixelFormat_BGRa16	Blue-Green-Red-alpha 16-bit
PixelFormat_RGBa32f	Red-Green-Blue-alpha 32-bit float
PixelFormat_BGR10	Blue-Green-Red 10-bit unpacked
PixelFormat_BGR10p	Blue-Green-Red 10-bit packed
PixelFormat_BGR12	Blue-Green-Red 12-bit unpacked
PixelFormat_BGR12p	Blue-Green-Red 12-bit packed
PixelFormat_BGR14	Blue-Green-Red 14-bit unpacked
PixelFormat_BGR16	Blue-Green-Red 16-bit
PixelFormat_BGR565p	Blue-Green-Red 5/6/5-bit packed
PixelFormat_R8	Red 8-bit
PixelFormat_R10	Red 10-bit
PixelFormat_R12	Red 12-bit
PixelFormat_R16	Red 16-bit
PixelFormat_G8	Green 8-bit
PixelFormat_G10	Green 10-bit
PixelFormat_G12	Green 12-bit
PixelFormat_G16	Green 16-bit
PixelFormat_B8	Blue 8-bit
PixelFormat_B10	Blue 10-bit
PixelFormat_B12	Blue 12-bit
PixelFormat_B16	Blue 16-bit
PixelFormat_Coord3D_ABC8	3D coordinate A-B-C 8-bit
PixelFormat_Coord3D_ABC8_Planar	3D coordinate A-B-C 8-bit planar
PixelFormat_Coord3D_ABC10p	3D coordinate A-B-C 10-bit packed
PixelFormat_Coord3D_ABC10p_Planar	3D coordinate A-B-C 10-bit packed planar
PixelFormat_Coord3D_ABC12p	3D coordinate A-B-C 12-bit packed
PixelFormat_Coord3D_ABC12p_Planar	3D coordinate A-B-C 12-bit packed planar
PixelFormat_Coord3D_ABC16	3D coordinate A-B-C 16-bit
PixelFormat_Coord3D_ABC16_Planar	3D coordinate A-B-C 16-bit planar
PixelFormat_Coord3D_ABC32f	3D coordinate A-B-C 32-bit floating point
PixelFormat_Coord3D_ABC32f_Planar	3D coordinate A-B-C 32-bit floating point planar
PixelFormat_Coord3D_AC8	3D coordinate A-C 8-bit
PixelFormat_Coord3D_AC8_Planar	3D coordinate A-C 8-bit planar
PixelFormat_Coord3D_AC10p	3D coordinate A-C 10-bit packed
PixelFormat_Coord3D_AC10p_Planar	3D coordinate A-C 10-bit packed planar
PixelFormat_Coord3D_AC12p	3D coordinate A-C 12-bit packed
PixelFormat_Coord3D_AC12p_Planar	3D coordinate A-C 12-bit packed planar
PixelFormat_Coord3D_AC16	3D coordinate A-C 16-bit
PixelFormat_Coord3D_AC16_Planar	3D coordinate A-C 16-bit planar
PixelFormat_Coord3D_AC32f	3D coordinate A-C 32-bit floating point
PixelFormat_Coord3D_AC32f_Planar	3D coordinate A-C 32-bit floating point planar

Enumerator

PixelFormat_Coord3D_A8	3D coordinate A 8-bit
PixelFormat_Coord3D_A10p	3D coordinate A 10-bit packed
PixelFormat_Coord3D_A12p	3D coordinate A 12-bit packed
PixelFormat_Coord3D_A16	3D coordinate A 16-bit
PixelFormat_Coord3D_A32f	3D coordinate A 32-bit floating point
PixelFormat_Coord3D_B8	3D coordinate B 8-bit
PixelFormat_Coord3D_B10p	3D coordinate B 10-bit packed
PixelFormat_Coord3D_B12p	3D coordinate B 12-bit packed
PixelFormat_Coord3D_B16	3D coordinate B 16-bit
PixelFormat_Coord3D_B32f	3D coordinate B 32-bit floating point
PixelFormat_Coord3D_C8	3D coordinate C 8-bit
PixelFormat_Coord3D_C10p	3D coordinate C 10-bit packed
PixelFormat_Coord3D_C12p	3D coordinate C 12-bit packed
PixelFormat_Coord3D_C16	3D coordinate C 16-bit
PixelFormat_Coord3D_C32f	3D coordinate C 32-bit floating point
PixelFormat_Confidence1	Confidence 1-bit unpacked
PixelFormat_Confidence1p	Confidence 1-bit packed
PixelFormat_Confidence8	Confidence 8-bit
PixelFormat_Confidence16	Confidence 16-bit
PixelFormat_Confidence32f	Confidence 32-bit floating point
PixelFormat_BiColorBGRG8	Bi-color Blue/Green - Red/Green 8-bit
PixelFormat_BiColorBGRG10	Bi-color Blue/Green - Red/Green 10-bit unpacked
PixelFormat_BiColorBGRG10p	Bi-color Blue/Green - Red/Green 10-bit packed
PixelFormat_BiColorBGRG12	Bi-color Blue/Green - Red/Green 12-bit unpacked
PixelFormat_BiColorBGRG12p	Bi-color Blue/Green - Red/Green 12-bit packed
PixelFormat_BiColorRGBG8	Bi-color Red/Green - Blue/Green 8-bit
PixelFormat_BiColorRGBG10	Bi-color Red/Green - Blue/Green 10-bit unpacked
PixelFormat_BiColorRGBG10p	Bi-color Red/Green - Blue/Green 10-bit packed
PixelFormat_BiColorRGBG12	Bi-color Red/Green - Blue/Green 12-bit unpacked
PixelFormat_BiColorRGBG12p	Bi-color Red/Green - Blue/Green 12-bit packed
PixelFormat_SCF1WBWG8	Sparse Color Filter #1 White-Blue-White-Green 8-bit
PixelFormat_SCF1WBWG10	Sparse Color Filter #1 White-Blue-White-Green 10-bit unpacked
PixelFormat_SCF1WBWG10p	Sparse Color Filter #1 White-Blue-White-Green 10-bit packed
PixelFormat_SCF1WBWG12	Sparse Color Filter #1 White-Blue-White-Green 12-bit unpacked
PixelFormat_SCF1WBWG12p	Sparse Color Filter #1 White-Blue-White-Green 12-bit packed
PixelFormat_SCF1WBWG14	Sparse Color Filter #1 White-Blue-White-Green 14-bit unpacked
PixelFormat_SCF1WBWG16	Sparse Color Filter #1 White-Blue-White-Green 16-bit unpacked
PixelFormat_SCF1WGWB8	Sparse Color Filter #1 White-Green-White-Blue 8-bit
PixelFormat_SCF1WGWB10	Sparse Color Filter #1 White-Green-White-Blue 10-bit unpacked
PixelFormat_SCF1WGWB10p	Sparse Color Filter #1 White-Green-White-Blue 10-bit packed
PixelFormat_SCF1WGWB12	Sparse Color Filter #1 White-Green-White-Blue 12-bit unpacked
PixelFormat_SCF1WGWB12p	Sparse Color Filter #1 White-Green-White-Blue 12-bit packed
PixelFormat_SCF1WGWB14	Sparse Color Filter #1 White-Green-White-Blue 14-bit unpacked
PixelFormat_SCF1WGWB16	Sparse Color Filter #1 White-Green-White-Blue 16-bit
PixelFormat_SCF1WGWR8	Sparse Color Filter #1 White-Green-White-Red 8-bit
PixelFormat_SCF1WGWR10	Sparse Color Filter #1 White-Green-White-Red 10-bit unpacked
PixelFormat_SCF1WGWR10p	Sparse Color Filter #1 White-Green-White-Red 10-bit packed

Enumerator

PixelFormat_SCF1WGWR12	Sparse Color Filter #1 White-Green-White-Red 12-bit unpacked
PixelFormat_SCF1WGWR12p	Sparse Color Filter #1 White-Green-White-Red 12-bit packed
PixelFormat_SCF1WGWR14	Sparse Color Filter #1 White-Green-White-Red 14-bit unpacked
PixelFormat_SCF1WGWR16	Sparse Color Filter #1 White-Green-White-Red 16-bit
PixelFormat_SCF1WRWG8	Sparse Color Filter #1 White-Red-White-Green 8-bit
PixelFormat_SCF1WRWG10	Sparse Color Filter #1 White-Red-White-Green 10-bit unpacked
PixelFormat_SCF1WRWG10p	Sparse Color Filter #1 White-Red-White-Green 10-bit packed
PixelFormat_SCF1WRWG12	Sparse Color Filter #1 White-Red-White-Green 12-bit unpacked
PixelFormat_SCF1WRWG12p	Sparse Color Filter #1 White-Red-White-Green 12-bit packed
PixelFormat_SCF1WRWG14	Sparse Color Filter #1 White-Red-White-Green 14-bit unpacked
PixelFormat_SCF1WRWG16	Sparse Color Filter #1 White-Red-White-Green 16-bit
PixelFormat_YCbCr8_CbYCr	YCbCr 4:4:4 8-bit
PixelFormat_YCbCr10_CbYCr	YCbCr 4:4:4 10-bit unpacked
PixelFormat_YCbCr10p_CbYCr	YCbCr 4:4:4 10-bit packed
PixelFormat_YCbCr12_CbYCr	YCbCr 4:4:4 12-bit unpacked
PixelFormat_YCbCr12p_CbYCr	YCbCr 4:4:4 12-bit packed
PixelFormat_YCbCr411_8_CbYYCrYY	YCbCr 4:1:1 8-bit
PixelFormat_YCbCr422_8_CbYCrY	YCbCr 4:2:2 8-bit
PixelFormat_YCbCr422_10	YCbCr 4:2:2 10-bit unpacked
PixelFormat_YCbCr422_10_CbYCrY	YCbCr 4:2:2 10-bit unpacked
PixelFormat_YCbCr422_10p	YCbCr 4:2:2 10-bit packed
PixelFormat_YCbCr422_10p_CbYCrY	YCbCr 4:2:2 10-bit packed
PixelFormat_YCbCr422_12	YCbCr 4:2:2 12-bit unpacked
PixelFormat_YCbCr422_12_CbYCrY	YCbCr 4:2:2 12-bit unpacked
PixelFormat_YCbCr422_12p	YCbCr 4:2:2 12-bit packed
PixelFormat_YCbCr422_12p_CbYCrY	YCbCr 4:2:2 12-bit packed
PixelFormat_YCbCr601_8_CbYCr	YCbCr 4:4:4 8-bit BT.601
PixelFormat_YCbCr601_10_CbYCr	YCbCr 4:4:4 10-bit unpacked BT.601
PixelFormat_YCbCr601_10p_CbYCr	YCbCr 4:4:4 10-bit packed BT.601
PixelFormat_YCbCr601_12_CbYCr	YCbCr 4:4:4 12-bit unpacked BT.601
PixelFormat_YCbCr601_12p_CbYCr	YCbCr 4:4:4 12-bit packed BT.601
PixelFormat_YCbCr601_411_8_CbYYCrYY	YCbCr 4:1:1 8-bit BT.601
PixelFormat_YCbCr601_422_8	YCbCr 4:2:2 8-bit BT.601
PixelFormat_YCbCr601_422_8_CbYCrY	YCbCr 4:2:2 8-bit BT.601
PixelFormat_YCbCr601_422_10	YCbCr 4:2:2 10-bit unpacked BT.601
PixelFormat_YCbCr601_422_10_CbYCrY	YCbCr 4:2:2 10-bit unpacked BT.601
PixelFormat_YCbCr601_422_10p	YCbCr 4:2:2 10-bit packed BT.601
PixelFormat_YCbCr601_422_10p_CbYCrY	YCbCr 4:2:2 10-bit packed BT.601
PixelFormat_YCbCr601_422_12	YCbCr 4:2:2 12-bit unpacked BT.601
PixelFormat_YCbCr601_422_12_CbYCrY	YCbCr 4:2:2 12-bit unpacked BT.601
PixelFormat_YCbCr601_422_12p	YCbCr 4:2:2 12-bit packed BT.601
PixelFormat_YCbCr601_422_12p_CbYCrY	YCbCr 4:2:2 12-bit packed BT.601
PixelFormat_YCbCr709_8_CbYCr	YCbCr 4:4:4 8-bit BT.709
PixelFormat_YCbCr709_10_CbYCr	YCbCr 4:4:4 10-bit unpacked BT.709
PixelFormat_YCbCr709_10p_CbYCr	YCbCr 4:4:4 10-bit packed BT.709
PixelFormat_YCbCr709_12_CbYCr	YCbCr 4:4:4 12-bit unpacked BT.709
PixelFormat_YCbCr709_12p_CbYCr	YCbCr 4:4:4 12-bit packed BT.709

Enumerator

PixelFormat_YCbCr709_411_8_CbYYCrYY	YCbCr 4:1:1 8-bit BT.709
PixelFormat_YCbCr709_422_8	YCbCr 4:2:2 8-bit BT.709
PixelFormat_YCbCr709_422_8_CbYCrY	YCbCr 4:2:2 8-bit BT.709
PixelFormat_YCbCr709_422_10	YCbCr 4:2:2 10-bit unpacked BT.709
PixelFormat_YCbCr709_422_10_CbYCrY	YCbCr 4:2:2 10-bit unpacked BT.709
PixelFormat_YCbCr709_422_10p	YCbCr 4:2:2 10-bit packed BT.709
PixelFormat_YCbCr709_422_10p_CbYCrY	YCbCr 4:2:2 10-bit packed BT.709
PixelFormat_YCbCr709_422_12	YCbCr 4:2:2 12-bit unpacked BT.709
PixelFormat_YCbCr709_422_12_CbYCrY	YCbCr 4:2:2 12-bit unpacked BT.709
PixelFormat_YCbCr709_422_12p	YCbCr 4:2:2 12-bit packed BT.709
PixelFormat_YCbCr709_422_12p_CbYCrY	YCbCr 4:2:2 12-bit packed BT.709
PixelFormat_YUV8_UYV	YUV 4:4:4 8-bit
PixelFormat_YUV411_8_UYYVYY	YUV 4:1:1 8-bit
PixelFormat_YUV422_8	YUV 4:2:2 8-bit
PixelFormat_YUV422_8_UYVY	YUV 4:2:2 8-bit
PixelFormat_Polarized8	Monochrome Polarized 8-bit
PixelFormat_Polarized10p	Monochrome Polarized 10-bit packed
PixelFormat_Polarized12p	Monochrome Polarized 12-bit packed
PixelFormat_Polarized16	Monochrome Polarized 16-bit
PixelFormat_BayerRGPolarized8	Polarized Bayer Red Green filter 8-bit
PixelFormat_BayerRGPolarized10p	Polarized Bayer Red Green filter 10-bit packed
PixelFormat_BayerRGPolarized12p	Polarized Bayer Red Green filter 12-bit packed
PixelFormat_BayerRGPolarized16	Polarized Bayer Red Green filter 16-bit
PixelFormat_LLCMono8	Lossless Compression Monochrome 8-bit
PixelFormat_LLCBayerRG8	Lossless Compression Bayer Red Green filter 8-bit
PixelFormat_JPEGMono8	JPEG Monochrome 8-bit
PixelFormat_JPEGColor8	JPEG Color 8-bit
PixelFormat_Raw16	Raw 16 bit.
PixelFormat_Raw8	Raw bit.
PixelFormat_R12_Jpeg	Red 12-bit JPEG.
PixelFormat_GR12_Jpeg	Green Red 12-bit JPEG.
PixelFormat_GB12_Jpeg	Green Blue 12-bit JPEG.
PixelFormat_B12_Jpeg	Blue 12-bit packed JPEG.
PixelFormat_GR12	Green-Red (single) channel from Bayer pattern 12-bit.
PixelFormat_GB12	Green-Blue (single) channel from Bayer pattern 12-bit.
UNKNOWN_PIXELFORMAT	
NUM_PIXELFORMAT	

13.8.1.128 spinPixelFormatInfoSelectorEnums

```
enum spinPixelFormatInfoSelectorEnums
```

< Select the pixel format for which the information will be returned.

Enumerator

PixelFormatInfoSelector_Mono1p	Monochrome 1-bit packed
PixelFormatInfoSelector_Mono2p	Monochrome 2-bit packed
PixelFormatInfoSelector_Mono4p	Monochrome 4-bit packed
PixelFormatInfoSelector_Mono8	Monochrome 8-bit
PixelFormatInfoSelector_Mono8s	Monochrome 8-bit signed
PixelFormatInfoSelector_Mono10	Monochrome 10-bit unpacked
PixelFormatInfoSelector_Mono10p	Monochrome 10-bit packed
PixelFormatInfoSelector_Mono12	Monochrome 12-bit unpacked
PixelFormatInfoSelector_Mono12p	Monochrome 12-bit packed
PixelFormatInfoSelector_Mono14	Monochrome 14-bit unpacked
PixelFormatInfoSelector_Mono16	Monochrome 16-bit
PixelFormatInfoSelector_Mono16s	Monochrome 16-bit signed
PixelFormatInfoSelector_Mono32f	Monochrome 32-bit float
PixelFormatInfoSelector_BayerBG8	Bayer Blue-Green 8-bit
PixelFormatInfoSelector_BayerBG10	Bayer Blue-Green 10-bit unpacked
PixelFormatInfoSelector_BayerBG10p	Bayer Blue-Green 10-bit packed
PixelFormatInfoSelector_BayerBG12	Bayer Blue-Green 12-bit unpacked
PixelFormatInfoSelector_BayerBG12p	Bayer Blue-Green 12-bit packed
PixelFormatInfoSelector_BayerBG16	Bayer Blue-Green 16-bit
PixelFormatInfoSelector_BayerGB8	Bayer Green-Blue 8-bit
PixelFormatInfoSelector_BayerGB10	Bayer Green-Blue 10-bit unpacked
PixelFormatInfoSelector_BayerGB10p	Bayer Green-Blue 10-bit packed
PixelFormatInfoSelector_BayerGB12	Bayer Green-Blue 12-bit unpacked
PixelFormatInfoSelector_BayerGB12p	Bayer Green-Blue 12-bit packed
PixelFormatInfoSelector_BayerGB16	Bayer Green-Blue 16-bit
PixelFormatInfoSelector_BayerGR8	Bayer Green-Red 8-bit
PixelFormatInfoSelector_BayerGR10	Bayer Green-Red 10-bit unpacked
PixelFormatInfoSelector_BayerGR10p	Bayer Green-Red 10-bit packed
PixelFormatInfoSelector_BayerGR12	Bayer Green-Red 12-bit unpacked
PixelFormatInfoSelector_BayerGR12p	Bayer Green-Red 12-bit packed
PixelFormatInfoSelector_BayerGR16	Bayer Green-Red 16-bit
PixelFormatInfoSelector_BayerRG8	Bayer Red-Green 8-bit
PixelFormatInfoSelector_BayerRG10	Bayer Red-Green 10-bit unpacked
PixelFormatInfoSelector_BayerRG10p	Bayer Red-Green 10-bit packed
PixelFormatInfoSelector_BayerRG12	Bayer Red-Green 12-bit unpacked
PixelFormatInfoSelector_BayerRG12p	Bayer Red-Green 12-bit packed
PixelFormatInfoSelector_BayerRG16	Bayer Red-Green 16-bit
PixelFormatInfoSelector_RGBa8	Red-Green-Blue-alpha 8-bit
PixelFormatInfoSelector_RGBa10	Red-Green-Blue-alpha 10-bit unpacked
PixelFormatInfoSelector_RGBa10p	Red-Green-Blue-alpha 10-bit packed
PixelFormatInfoSelector_RGBa12	Red-Green-Blue-alpha 12-bit unpacked
PixelFormatInfoSelector_RGBa12p	Red-Green-Blue-alpha 12-bit packed
PixelFormatInfoSelector_RGBa14	Red-Green-Blue-alpha 14-bit unpacked
PixelFormatInfoSelector_RGBa16	Red-Green-Blue-alpha 16-bit
PixelFormatInfoSelector_RGB8	Red-Green-Blue 8-bit
PixelFormatInfoSelector_RGB8_Planar	Red-Green-Blue 8-bit planar
PixelFormatInfoSelector_RGB10	Red-Green-Blue 10-bit unpacked

Enumerator

PixelFormatInfoSelector_RGB10_Planar	Red-Green-Blue 10-bit unpacked planar
PixelFormatInfoSelector_RGB10p	Red-Green-Blue 10-bit packed
PixelFormatInfoSelector_RGB10p32	Red-Green-Blue 10-bit packed into 32-bit
PixelFormatInfoSelector_RGB12	Red-Green-Blue 12-bit unpacked
PixelFormatInfoSelector_RGB12_Planar	Red-Green-Blue 12-bit unpacked planar
PixelFormatInfoSelector_RGB12p	Red-Green-Blue 12-bit packed
PixelFormatInfoSelector_RGB14	Red-Green-Blue 14-bit unpacked
PixelFormatInfoSelector_RGB16	Red-Green-Blue 16-bit
PixelFormatInfoSelector_RGB16s	Red-Green-Blue 16-bit signed
PixelFormatInfoSelector_RGB32f	Red-Green-Blue 32-bit float
PixelFormatInfoSelector_RGB16_Planar	Red-Green-Blue 16-bit planar
PixelFormatInfoSelector_RGB565p	Red-Green-Blue 5/6/5-bit packed
PixelFormatInfoSelector_BGRa8	Blue-Green-Red-alpha 8-bit
PixelFormatInfoSelector_BGRa10	Blue-Green-Red-alpha 10-bit unpacked
PixelFormatInfoSelector_BGRa10p	Blue-Green-Red-alpha 10-bit packed
PixelFormatInfoSelector_BGRa12	Blue-Green-Red-alpha 12-bit unpacked
PixelFormatInfoSelector_BGRa12p	Blue-Green-Red-alpha 12-bit packed
PixelFormatInfoSelector_BGRa14	Blue-Green-Red-alpha 14-bit unpacked
PixelFormatInfoSelector_BGRa16	Blue-Green-Red-alpha 16-bit
PixelFormatInfoSelector_RGBa32f	Red-Green-Blue-alpha 32-bit float
PixelFormatInfoSelector_BGR8	Blue-Green-Red 8-bit
PixelFormatInfoSelector_BGR10	Blue-Green-Red 10-bit unpacked
PixelFormatInfoSelector_BGR10p	Blue-Green-Red 10-bit packed
PixelFormatInfoSelector_BGR12	Blue-Green-Red 12-bit unpacked
PixelFormatInfoSelector_BGR12p	Blue-Green-Red 12-bit packed
PixelFormatInfoSelector_BGR14	Blue-Green-Red 14-bit unpacked
PixelFormatInfoSelector_BGR16	Blue-Green-Red 16-bit
PixelFormatInfoSelector_BGR565p	Blue-Green-Red 5/6/5-bit packed
PixelFormatInfoSelector_R8	Red 8-bit
PixelFormatInfoSelector_R10	Red 10-bit
PixelFormatInfoSelector_R12	Red 12-bit
PixelFormatInfoSelector_R16	Red 16-bit
PixelFormatInfoSelector_G8	Green 8-bit
PixelFormatInfoSelector_G10	Green 10-bit
PixelFormatInfoSelector_G12	Green 12-bit
PixelFormatInfoSelector_G16	Green 16-bit
PixelFormatInfoSelector_B8	Blue 8-bit
PixelFormatInfoSelector_B10	Blue 10-bit
PixelFormatInfoSelector_B12	Blue 12-bit
PixelFormatInfoSelector_B16	Blue 16-bit
PixelFormatInfoSelector_Coord3D_ABC8	3D coordinate A-B-C 8-bit
PixelFormatInfoSelector_Coord3D_ABC8_Planar	3D coordinate A-B-C 8-bit planar
PixelFormatInfoSelector_Coord3D_ABC10p	3D coordinate A-B-C 10-bit packed
PixelFormatInfoSelector_Coord3D_ABC10p_Planar	3D coordinate A-B-C 10-bit packed planar
PixelFormatInfoSelector_Coord3D_ABC12p	3D coordinate A-B-C 12-bit packed
PixelFormatInfoSelector_Coord3D_ABC12p_Planar	3D coordinate A-B-C 12-bit packed planar

Enumerator

PixelFormatInfoSelector_Coord3D_ABC16	3D coordinate A-B-C 16-bit
PixelFormatInfoSelector_Coord3D_ABC16_Planar	3D coordinate A-B-C 16-bit planar
PixelFormatInfoSelector_Coord3D_ABC32f	3D coordinate A-B-C 32-bit floating point
PixelFormatInfoSelector_Coord3D_ABC32f_Planar	3D coordinate A-B-C 32-bit floating point planar
PixelFormatInfoSelector_Coord3D_AC8	3D coordinate A-C 8-bit
PixelFormatInfoSelector_Coord3D_AC8_Planar	3D coordinate A-C 8-bit planar
PixelFormatInfoSelector_Coord3D_AC10p	3D coordinate A-C 10-bit packed
PixelFormatInfoSelector_Coord3D_AC10p_Planar	3D coordinate A-C 10-bit packed planar
PixelFormatInfoSelector_Coord3D_AC12p	3D coordinate A-C 12-bit packed
PixelFormatInfoSelector_Coord3D_AC12p_Planar	3D coordinate A-C 12-bit packed planar
PixelFormatInfoSelector_Coord3D_AC16	3D coordinate A-C 16-bit
PixelFormatInfoSelector_Coord3D_AC16_Planar	3D coordinate A-C 16-bit planar
PixelFormatInfoSelector_Coord3D_AC32f	3D coordinate A-C 32-bit floating point
PixelFormatInfoSelector_Coord3D_AC32f_Planar	3D coordinate A-C 32-bit floating point planar
PixelFormatInfoSelector_Coord3D_A8	3D coordinate A 8-bit
PixelFormatInfoSelector_Coord3D_A10p	3D coordinate A 10-bit packed
PixelFormatInfoSelector_Coord3D_A12p	3D coordinate A 12-bit packed
PixelFormatInfoSelector_Coord3D_A16	3D coordinate A 16-bit
PixelFormatInfoSelector_Coord3D_A32f	3D coordinate A 32-bit floating point
PixelFormatInfoSelector_Coord3D_B8	3D coordinate B 8-bit
PixelFormatInfoSelector_Coord3D_B10p	3D coordinate B 10-bit packed
PixelFormatInfoSelector_Coord3D_B12p	3D coordinate B 12-bit packed
PixelFormatInfoSelector_Coord3D_B16	3D coordinate B 16-bit
PixelFormatInfoSelector_Coord3D_B32f	3D coordinate B 32-bit floating point
PixelFormatInfoSelector_Coord3D_C8	3D coordinate C 8-bit
PixelFormatInfoSelector_Coord3D_C10p	3D coordinate C 10-bit packed
PixelFormatInfoSelector_Coord3D_C12p	3D coordinate C 12-bit packed
PixelFormatInfoSelector_Coord3D_C16	3D coordinate C 16-bit
PixelFormatInfoSelector_Coord3D_C32f	3D coordinate C 32-bit floating point
PixelFormatInfoSelector_Confidence1	Confidence 1-bit unpacked
PixelFormatInfoSelector_Confidence1p	Confidence 1-bit packed
PixelFormatInfoSelector_Confidence8	Confidence 8-bit
PixelFormatInfoSelector_Confidence16	Confidence 16-bit
PixelFormatInfoSelector_Confidence32f	Confidence 32-bit floating point
PixelFormatInfoSelector_BiColorBGRG8	Bi-color Blue/Green - Red/Green 8-bit
PixelFormatInfoSelector_BiColorBGRG10	Bi-color Blue/Green - Red/Green 10-bit unpacked
PixelFormatInfoSelector_BiColorBGRG10p	Bi-color Blue/Green - Red/Green 10-bit packed
PixelFormatInfoSelector_BiColorBGRG12	Bi-color Blue/Green - Red/Green 12-bit unpacked
PixelFormatInfoSelector_BiColorBGRG12p	Bi-color Blue/Green - Red/Green 12-bit packed
PixelFormatInfoSelector_BiColorRGBG8	Bi-color Red/Green - Blue/Green 8-bit
PixelFormatInfoSelector_BiColorRGBG10	Bi-color Red/Green - Blue/Green 10-bit unpacked
PixelFormatInfoSelector_BiColorRGBG10p	Bi-color Red/Green - Blue/Green 10-bit packed
PixelFormatInfoSelector_BiColorRGBG12	Bi-color Red/Green - Blue/Green 12-bit unpacked
PixelFormatInfoSelector_BiColorRGBG12p	Bi-color Red/Green - Blue/Green 12-bit packed
PixelFormatInfoSelector_SCF1WBWG8	Sparse Color Filter #1 White-Blue-White-Green 8-bit
PixelFormatInfoSelector_SCF1WBWG10	Sparse Color Filter #1 White-Blue-White-Green 10-bit unpacked

Enumerator

PixelFormatInfoSelector_SCF1WBWG10p	Sparse Color Filter #1 White-Blue-White-Green 10-bit packed
PixelFormatInfoSelector_SCF1WBWG12	Sparse Color Filter #1 White-Blue-White-Green 12-bit unpacked
PixelFormatInfoSelector_SCF1WBWG12p	Sparse Color Filter #1 White-Blue-White-Green 12-bit packed
PixelFormatInfoSelector_SCF1WBWG14	Sparse Color Filter #1 White-Blue-White-Green 14-bit unpacked
PixelFormatInfoSelector_SCF1WBWG16	Sparse Color Filter #1 White-Blue-White-Green 16-bit unpacked
PixelFormatInfoSelector_SCF1WGWB8	Sparse Color Filter #1 White-Green-White-Blue 8-bit
PixelFormatInfoSelector_SCF1WGWB10	Sparse Color Filter #1 White-Green-White-Blue 10-bit unpacked
PixelFormatInfoSelector_SCF1WGWB10p	Sparse Color Filter #1 White-Green-White-Blue 10-bit packed
PixelFormatInfoSelector_SCF1WGWB12	Sparse Color Filter #1 White-Green-White-Blue 12-bit unpacked
PixelFormatInfoSelector_SCF1WGWB12p	Sparse Color Filter #1 White-Green-White-Blue 12-bit packed
PixelFormatInfoSelector_SCF1WGWB14	Sparse Color Filter #1 White-Green-White-Blue 14-bit unpacked
PixelFormatInfoSelector_SCF1WGWB16	Sparse Color Filter #1 White-Green-White-Blue 16-bit
PixelFormatInfoSelector_SCF1WGWR8	Sparse Color Filter #1 White-Green-White-Red 8-bit
PixelFormatInfoSelector_SCF1WGWR10	Sparse Color Filter #1 White-Green-White-Red 10-bit unpacked
PixelFormatInfoSelector_SCF1WGWR10p	Sparse Color Filter #1 White-Green-White-Red 10-bit packed
PixelFormatInfoSelector_SCF1WGWR12	Sparse Color Filter #1 White-Green-White-Red 12-bit unpacked
PixelFormatInfoSelector_SCF1WGWR12p	Sparse Color Filter #1 White-Green-White-Red 12-bit packed
PixelFormatInfoSelector_SCF1WGWR14	Sparse Color Filter #1 White-Green-White-Red 14-bit unpacked
PixelFormatInfoSelector_SCF1WGWR16	Sparse Color Filter #1 White-Green-White-Red 16-bit
PixelFormatInfoSelector_SCF1WRWG8	Sparse Color Filter #1 White-Red-White-Green 8-bit
PixelFormatInfoSelector_SCF1WRWG10	Sparse Color Filter #1 White-Red-White-Green 10-bit unpacked
PixelFormatInfoSelector_SCF1WRWG10p	Sparse Color Filter #1 White-Red-White-Green 10-bit packed
PixelFormatInfoSelector_SCF1WRWG12	Sparse Color Filter #1 White-Red-White-Green 12-bit unpacked
PixelFormatInfoSelector_SCF1WRWG12p	Sparse Color Filter #1 White-Red-White-Green 12-bit packed
PixelFormatInfoSelector_SCF1WRWG14	Sparse Color Filter #1 White-Red-White-Green 14-bit unpacked
PixelFormatInfoSelector_SCF1WRWG16	Sparse Color Filter #1 White-Red-White-Green 16-bit
PixelFormatInfoSelector_YCbCr8	YCbCr 4:4:4 8-bit
PixelFormatInfoSelector_YCbCr8_CbYCr	YCbCr 4:4:4 8-bit
PixelFormatInfoSelector_YCbCr10_CbYCr	YCbCr 4:4:4 10-bit unpacked
PixelFormatInfoSelector_YCbCr10p_CbYCr	YCbCr 4:4:4 10-bit packed

Enumerator

PixelFormatInfoSelector_YCbCr12_CbYCr	YCbCr 4:4:4 12-bit unpacked
PixelFormatInfoSelector_YCbCr12p_CbYCr	YCbCr 4:4:4 12-bit packed
PixelFormatInfoSelector_YCbCr411_8	YCbCr 4:1:1 8-bit
PixelFormatInfoSelector_YCbCr411_8_CbYYCrYY	YCbCr 4:1:1 8-bit
PixelFormatInfoSelector_YCbCr422_8	YCbCr 4:2:2 8-bit
PixelFormatInfoSelector_YCbCr422_8_CbYCrY	YCbCr 4:2:2 8-bit
PixelFormatInfoSelector_YCbCr422_10	YCbCr 4:2:2 10-bit unpacked
PixelFormatInfoSelector_YCbCr422_10_CbYCrY	YCbCr 4:2:2 10-bit unpacked
PixelFormatInfoSelector_YCbCr422_10p	YCbCr 4:2:2 10-bit packed
PixelFormatInfoSelector_YCbCr422_10p_CbYCrY	YCbCr 4:2:2 10-bit packed
PixelFormatInfoSelector_YCbCr422_12	YCbCr 4:2:2 12-bit unpacked
PixelFormatInfoSelector_YCbCr422_12_CbYCrY	YCbCr 4:2:2 12-bit unpacked
PixelFormatInfoSelector_YCbCr422_12p	YCbCr 4:2:2 12-bit packed
PixelFormatInfoSelector_YCbCr422_12p_CbYCrY	YCbCr 4:2:2 12-bit packed
PixelFormatInfoSelector_YCbCr601_8_CbYCr	YCbCr 4:4:4 8-bit BT.601
PixelFormatInfoSelector_YCbCr601_10_CbYCr	YCbCr 4:4:4 10-bit unpacked BT.601
PixelFormatInfoSelector_YCbCr601_10p_CbYCr	YCbCr 4:4:4 10-bit packed BT.601
PixelFormatInfoSelector_YCbCr601_12_CbYCr	YCbCr 4:4:4 12-bit unpacked BT.601
PixelFormatInfoSelector_YCbCr601_12p_CbYCr	YCbCr 4:4:4 12-bit packed BT.601
PixelFormatInfoSelector_YCbCr601_411_8_Cb↩ YYCrYY	YCbCr 4:1:1 8-bit BT.601
PixelFormatInfoSelector_YCbCr601_422_8	YCbCr 4:2:2 8-bit BT.601
PixelFormatInfoSelector_YCbCr601_422_8_CbYCrY	YCbCr 4:2:2 8-bit BT.601
PixelFormatInfoSelector_YCbCr601_422_10	YCbCr 4:2:2 10-bit unpacked BT.601
PixelFormatInfoSelector_YCbCr601_422_10_Cb↩ YCrY	YCbCr 4:2:2 10-bit unpacked BT.601
PixelFormatInfoSelector_YCbCr601_422_10p	YCbCr 4:2:2 10-bit packed BT.601
PixelFormatInfoSelector_YCbCr601_422_10p_Cb↩ YCrY	YCbCr 4:2:2 10-bit packed BT.601
PixelFormatInfoSelector_YCbCr601_422_12	YCbCr 4:2:2 12-bit unpacked BT.601
PixelFormatInfoSelector_YCbCr601_422_12_Cb↩ YCrY	YCbCr 4:2:2 12-bit unpacked BT.601
PixelFormatInfoSelector_YCbCr601_422_12p	YCbCr 4:2:2 12-bit packed BT.601
PixelFormatInfoSelector_YCbCr601_422_12p_Cb↩ YCrY	YCbCr 4:2:2 12-bit packed BT.601
PixelFormatInfoSelector_YCbCr709_8_CbYCr	YCbCr 4:4:4 8-bit BT.709
PixelFormatInfoSelector_YCbCr709_10_CbYCr	YCbCr 4:4:4 10-bit unpacked BT.709
PixelFormatInfoSelector_YCbCr709_10p_CbYCr	YCbCr 4:4:4 10-bit packed BT.709
PixelFormatInfoSelector_YCbCr709_12_CbYCr	YCbCr 4:4:4 12-bit unpacked BT.709
PixelFormatInfoSelector_YCbCr709_12p_CbYCr	YCbCr 4:4:4 12-bit packed BT.709
PixelFormatInfoSelector_YCbCr709_411_8_Cb↩ YYCrYY	YCbCr 4:1:1 8-bit BT.709
PixelFormatInfoSelector_YCbCr709_422_8	YCbCr 4:2:2 8-bit BT.709
PixelFormatInfoSelector_YCbCr709_422_8_CbYCrY	YCbCr 4:2:2 8-bit BT.709
PixelFormatInfoSelector_YCbCr709_422_10	YCbCr 4:2:2 10-bit unpacked BT.709
PixelFormatInfoSelector_YCbCr709_422_10_Cb↩ YCrY	YCbCr 4:2:2 10-bit unpacked BT.709
PixelFormatInfoSelector_YCbCr709_422_10p	YCbCr 4:2:2 10-bit packed BT.709

Enumerator

PixelFormatInfoSelector_YCbCr709_422_10p_Cb↩YCrY	YCbCr 4:2:2 10-bit packed BT.709
PixelFormatInfoSelector_YCbCr709_422_12	YCbCr 4:2:2 12-bit unpacked BT.709
PixelFormatInfoSelector_YCbCr709_422_12_Cb↩YCrY	YCbCr 4:2:2 12-bit unpacked BT.709
PixelFormatInfoSelector_YCbCr709_422_12p	YCbCr 4:2:2 12-bit packed BT.709
PixelFormatInfoSelector_YCbCr709_422_12p_Cb↩YCrY	YCbCr 4:2:2 12-bit packed BT.709
PixelFormatInfoSelector_YUV8_UYV	YUV 4:4:4 8-bit
PixelFormatInfoSelector_YUV411_8_UYYVYY	YUV 4:1:1 8-bit
PixelFormatInfoSelector_YUV422_8	YUV 4:2:2 8-bit
PixelFormatInfoSelector_YUV422_8_UYVY	YUV 4:2:2 8-bit
PixelFormatInfoSelector_Polarized8	Monochrome Polarized 8-bit
PixelFormatInfoSelector_Polarized10p	Monochrome Polarized 10-bit packed
PixelFormatInfoSelector_Polarized12p	Monochrome Polarized 12-bit packed
PixelFormatInfoSelector_Polarized16	Monochrome Polarized 16-bit
PixelFormatInfoSelector_BayerRGPolarized8	Polarized Bayer Red Green filter 8-bit
PixelFormatInfoSelector_BayerRGPolarized10p	Polarized Bayer Red Green filter 10-bit packed
PixelFormatInfoSelector_BayerRGPolarized12p	Polarized Bayer Red Green filter 12-bit packed
PixelFormatInfoSelector_BayerRGPolarized16	Polarized Bayer Red Green filter 16-bit
PixelFormatInfoSelector_LLCMono8	Lossless Compression Monochrome 8-bit
PixelFormatInfoSelector_LLCBayerRG8	Lossless Compression Bayer Red Green filter 8-bit
PixelFormatInfoSelector_JPEGMono8	JPEG Monochrome 8-bit
PixelFormatInfoSelector_JPEGColor8	JPEG Color 8-bit
NUM_PIXELFORMATINFOSELECTOR	

13.8.1.129 spinPixelSizeEnums

```
enum spinPixelSizeEnums
```

< Total size in bits of a pixel of the image.

Enumerator

PixelSize_Bpp1	1 bit per pixel.
PixelSize_Bpp2	2 bits per pixel.
PixelSize_Bpp4	4 bits per pixel.
PixelSize_Bpp8	8 bits per pixel.
PixelSize_Bpp10	10 bits per pixel.
PixelSize_Bpp12	12 bits per pixel.
PixelSize_Bpp14	14 bits per pixel.
PixelSize_Bpp16	16 bits per pixel.
PixelSize_Bpp20	20 bits per pixel.
PixelSize_Bpp24	24 bits per pixel.
PixelSize_Bpp30	30 bits per pixel.
PixelSize_Bpp32	32 bits per pixel.

Enumerator

PixelSize_Bpp36	36 bits per pixel.
PixelSize_Bpp48	48 bits per pixel.
PixelSize_Bpp64	64 bits per pixel.
PixelSize_Bpp96	96 bits per pixel.
NUM_PIXELSIZE	

13.8.1.130 spinRegionDestinationEnums

```
enum spinRegionDestinationEnums
```

< Control the destination of the selected region.

Enumerator

RegionDestination_Stream0	The destination of the region is the data stream 0.
RegionDestination_Stream1	The destination of the region is the data stream 1.
RegionDestination_Stream2	The destination of the region is the data stream 2.
NUM_REGIONDESTINATION	

13.8.1.131 spinRegionModeEnums

```
enum spinRegionModeEnums
```

< Controls if the selected Region of interest is active and streaming.

Enumerator

RegionMode_Off	Disable the usage of the Region.
RegionMode_On	Enable the usage of the Region.
NUM_REGIONMODE	

13.8.1.132 spinRegionSelectorEnums

```
enum spinRegionSelectorEnums
```

< Selects the Region of interest to control. The RegionSelector feature allows devices that are able to extract multiple regions out of an image, to configure the features of those individual regions independently.

Enumerator

RegionSelector_Region0	Selected feature will control the region 0.
RegionSelector_Region1	Selected feature will control the region 1.
RegionSelector_Region2	Selected feature will control the region 2.
RegionSelector_All	Selected features will control all the regions at the same time.
NUM_REGIONSELECTOR	

13.8.1.133 spinRgbTransformLightSourceEnums

`enum spinRgbTransformLightSourceEnums`

< Used to select from a set of RGBtoRGB transform matricies calibrated for different light sources. Selecting a value also sets the white balance ratios (BalanceRatioRed and BalanceRatioBlue), but those can be overwritten through manual or auto white balance.

Enumerator

RgbTransformLightSource_General	Uses a matrix calibrated for a wide range of light sources.
RgbTransformLightSource_Tungsten2800K	Uses a matrix optimized for tungsten/incandescent light with color temperature 2800K.
RgbTransformLightSource_WarmFluorescent3000K	Uses a matrix optimized for a typical warm fluoresecent light with color temperature 3000K.
RgbTransformLightSource_CoolFluorescent4000K	Uses a matrix optimized for a typical cool fluoresecent light with color temperature 4000K.
RgbTransformLightSource_Daylight5000K	Uses a matrix optimized for noon Daylight with color temperature 5000K.
RgbTransformLightSource_Cloudy6500K	Uses a matrix optimized for a cloudy sky with color temperature 6500K.
RgbTransformLightSource_Shade8000K	Uses a matrix optimized for shade with color temperature 8000K.
RgbTransformLightSource_Custom	Uses a custom matrix set by the user through the ColorTransformationValueSelector and ColorTransformationValue controls.
NUM_RGBTRANSFORMLIGHTSOURCE	

13.8.1.134 spinScan3dCoordinateReferenceSelectorEnums

`enum spinScan3dCoordinateReferenceSelectorEnums`

< Sets the index to read a coordinate system reference value defining the transform of a point from the current (Anchor or Transformed) system to the reference system.

Enumerator

Scan3dCoordinateReferenceSelector_RotationX	Rotation around X axis.
Scan3dCoordinateReferenceSelector_RotationY	Rotation around Y axis.
Scan3dCoordinateReferenceSelector_RotationZ	Rotation around Z axis.
Scan3dCoordinateReferenceSelector_TranslationX	X axis translation.
Scan3dCoordinateReferenceSelector_TranslationY	Y axis translation.
Scan3dCoordinateReferenceSelector_TranslationZ	Z axis translation.
NUM_SCAN3DCOORDINATEREFERENCESELECTOR	

13.8.1.135 spinScan3dCoordinateSelectorEnums

```
enum spinScan3dCoordinateSelectorEnums
```

< Selects the individual coordinates in the vectors for 3D information/transformation.

Enumerator

Scan3dCoordinateSelector_CoordinateA	The first (X or Theta) coordinate
Scan3dCoordinateSelector_CoordinateB	The second (Y or Phi) coordinate
Scan3dCoordinateSelector_CoordinateC	The third (Z or Rho) coordinate.
NUM_SCAN3DCOORDINATESELECTOR	

13.8.1.136 spinScan3dCoordinateSystemEnums

```
enum spinScan3dCoordinateSystemEnums
```

< Specifies the Coordinate system to use for the device.

Enumerator

Scan3dCoordinateSystem_Cartesian	Default value. 3-axis orthogonal, right-hand X-Y-Z.
Scan3dCoordinateSystem_Spherical	A Theta-Phi-Rho coordinate system.
Scan3dCoordinateSystem_Cylindrical	A Theta-Y-Rho coordinate system.
NUM_SCAN3DCOORDINATESYSTEM	

13.8.1.137 spinScan3dCoordinateSystemReferenceEnums

```
enum spinScan3dCoordinateSystemReferenceEnums
```

< Defines coordinate system reference location.

Enumerator

Scan3dCoordinateSystemReference_Anchor	Default value. Original fixed reference. The coordinate system fixed relative the camera reference point marker is used.
Scan3dCoordinateSystemReference_Transformed	Transformed reference system. The transformed coordinate system is used according to the definition in the rotation and translation matrices.
NUM_SCAN3DCOORDINATESYSTEMREFERENCE	

13.8.1.138 spinScan3dCoordinateTransformSelectorEnums

```
enum spinScan3dCoordinateTransformSelectorEnums
```

< Sets the index to read/write a coordinate transform value.

Enumerator

Scan3dCoordinateTransformSelector_RotationX	Rotation around X axis.
Scan3dCoordinateTransformSelector_RotationY	Rotation around Y axis.
Scan3dCoordinateTransformSelector_RotationZ	Rotation around Z axis.
Scan3dCoordinateTransformSelector_TranslationX	Translation along X axis.
Scan3dCoordinateTransformSelector_TranslationY	Translation along Y axis.
Scan3dCoordinateTransformSelector_TranslationZ	Translation along Z axis.
NUM_SCAN3DCOORDINATETRANSFORMSELECTOR	

13.8.1.139 spinScan3dDistanceUnitEnums

```
enum spinScan3dDistanceUnitEnums
```

< Specifies the unit used when delivering calibrated distance data.

Enumerator

Scan3dDistanceUnit_Millimeter	Distance values are in millimeter units (default).
Scan3dDistanceUnit_Inch	Distance values are in inch units.
NUM_SCAN3DDISTANCEUNIT	

13.8.1.140 spinScan3dOutputModeEnums

```
enum spinScan3dOutputModeEnums
```

< Controls the Calibration and data organization of the device, naming the coordinates transmitted.

Enumerator

Scan3dOutputMode_UncalibratedC	Uncalibrated 2.5D Depth map. The distance data does not represent a physical unit and may be non-linear. The data is a 2.5D range map only.
Scan3dOutputMode_CalibratedABC_Grid	3 Coordinates in grid organization. The full 3 coordinate data with the grid array organization from the sensor kept.
Scan3dOutputMode_CalibratedABC_PointCloud	3 Coordinates without organization. The full 3 coordinate data without any organization of data points. Typically only valid points transmitted giving varying image size.
Scan3dOutputMode_CalibratedAC	2 Coordinates with fixed B sampling. The data is sent as a A and C coordinates (X,Z or Theta,Rho). The B (Y) axis uses the scale and offset parameters for the B axis.
Scan3dOutputMode_CalibratedAC_Linescan	2 Coordinates with varying sampling. The data is sent as a A and C coordinates (X,Z or Theta,Rho). The B (Y) axis comes from the encoder chunk value.
Scan3dOutputMode_CalibratedC	Calibrated 2.5D Depth map. The distance data is expressed in the chosen distance unit. The data is a 2.5D range map. No information on X-Y axes available.
Scan3dOutputMode_CalibratedC_Linescan	Depth Map with varying B sampling. The distance data is expressed in the chosen distance unit. The data is a 2.5D range map. The B (Y) axis comes from the encoder chunk value.
Scan3dOutputMode_RectifiedC	Rectified 2.5D Depth map. The distance data has been rectified to a uniform sampling pattern in the X and Y direction. The data is a 2.5D range map only. If a complete 3D point cloud is rectified but transmitted as explicit coordinates it should be transmitted as one of the "CalibratedABC" formats.
Scan3dOutputMode_RectifiedC_Linescan	Rectified 2.5D Depth map with varying B sampling. The data is sent as rectified 1D profiles using Coord3D_C pixels. The B (Y) axis comes from the encoder chunk value.
Scan3dOutputMode_DisparityC	Disparity 2.5D Depth map. The distance is inversely proportional to the pixel (disparity) value.
Scan3dOutputMode_DisparityC_Linescan	Disparity 2.5D Depth map with varying B sampling. The distance is inversely proportional to the pixel (disparity) value. The B (Y) axis comes from the encoder chunk value.
NUM_SCAN3DOUTPUTMODE	

13.8.1.141 spinSensorDigitizationTapsEnums

```
enum spinSensorDigitizationTapsEnums
```

< Number of digitized samples outputted simultaneously by the camera A/D conversion stage.

Enumerator

SensorDigitizationTaps_One	1 tap.
SensorDigitizationTaps_Two	2 taps.
SensorDigitizationTaps_Three	3 taps.

Enumerator

SensorDigitizationTaps_Four	4 taps.
SensorDigitizationTaps_Eight	8 taps.
SensorDigitizationTaps_Ten	10 taps.
NUM_SENSORDIGITIZATIONTAPS	

13.8.1.142 spinSensorShutterModeEnums

```
enum spinSensorShutterModeEnums
```

< Sets the shutter mode of the device.

Enumerator

SensorShutterMode_Global	The shutter opens and closes at the same time for all pixels. All the pixels are exposed for the same length of time at the same time.
SensorShutterMode_Rolling	The shutter opens and closes sequentially for groups (typically lines) of pixels. All the pixels are exposed for the same length of time but not at the same time.
SensorShutterMode_GlobalReset	The shutter opens at the same time for all pixels but ends in a sequential manner. The pixels are exposed for different lengths of time.
NUM_SENSORSHUTTERMODE	

13.8.1.143 spinSensorTapsEnums

```
enum spinSensorTapsEnums
```

< Number of taps of the camera sensor.

Enumerator

SensorTaps_One	1 tap.
SensorTaps_Two	2 taps.
SensorTaps_Three	3 taps.
SensorTaps_Four	4 taps.
SensorTaps_Eight	8 taps.
SensorTaps_Ten	10 taps.
NUM_SENSORTAPS	

13.8.1.144 spinSequencerConfigurationModeEnums

```
enum spinSequencerConfigurationModeEnums
```

< Controls whether or not a sequencer is in configuration mode.

Enumerator

SequencerConfigurationMode_Off	
SequencerConfigurationMode_On	
NUM_SEQUENCERCONFIGURATIONMODE	

13.8.1.145 spinSequencerConfigurationValidEnums

```
enum spinSequencerConfigurationValidEnums
```

< Display whether the current sequencer configuration is valid to run.

Enumerator

SequencerConfigurationValid_No	
SequencerConfigurationValid_Yes	
NUM_SEQUENCERCONFIGURATIONVALID	

13.8.1.146 spinSequencerModeEnums

```
enum spinSequencerModeEnums
```

< Controls whether or not a sequencer is active.

Enumerator

SequencerMode_Off	
SequencerMode_On	
NUM_SEQUENCERMODE	

13.8.1.147 spinSequencerSetValidEnums

```
enum spinSequencerSetValidEnums
```

< Displays whether the currently selected sequencer set's register contents are valid to use.

Enumerator

SequencerSetValid_No	
SequencerSetValid_Yes	
NUM_SEQUENCERSETVALID	

13.8.1.148 spinSequencerTriggerActivationEnums

`enum spinSequencerTriggerActivationEnums`

< Specifies the activation mode of the sequencer trigger.

Enumerator

SequencerTriggerActivation_RisingEdge	
SequencerTriggerActivation_FallingEdge	
SequencerTriggerActivation_AnyEdge	
SequencerTriggerActivation_LevelHigh	
SequencerTriggerActivation_LevelLow	
NUM_SEQUENCERTRIGGERACTIVATION	

13.8.1.149 spinSequencerTriggerSourceEnums

`enum spinSequencerTriggerSourceEnums`

< Specifies the internal signal or physical input line to use as the sequencer trigger source.

Enumerator

SequencerTriggerSource_Off	
SequencerTriggerSource_FrameStart	
NUM_SEQUENCERTRIGGERSOURCE	

13.8.1.150 spinSerialPortBaudRateEnums

`enum spinSerialPortBaudRateEnums`

< This feature controls the baud rate used by the selected serial port.

Enumerator

SerialPortBaudRate_Baud300	
SerialPortBaudRate_Baud600	
SerialPortBaudRate_Baud1200	
SerialPortBaudRate_Baud2400	
SerialPortBaudRate_Baud4800	
SerialPortBaudRate_Baud9600	
SerialPortBaudRate_Baud14400	
SerialPortBaudRate_Baud19200	
SerialPortBaudRate_Baud38400	
SerialPortBaudRate_Baud57600	
SerialPortBaudRate_Baud115200	
SerialPortBaudRate_Baud230400	
SerialPortBaudRate_Baud460800	
SerialPortBaudRate_Baud921600	
NUM_SERIALPORTBAUDRATE	

13.8.1.151 spinSerialPortParityEnums

```
enum spinSerialPortParityEnums
```

< This feature controls the parity used by the selected serial port.

Enumerator

SerialPortParity_None	
SerialPortParity_Odd	
SerialPortParity_Even	
SerialPortParity_Mark	
SerialPortParity_Space	
NUM_SERIALPORTPARITY	

13.8.1.152 spinSerialPortSelectorEnums

```
enum spinSerialPortSelectorEnums
```

< Selects which serial port of the device to control.

Enumerator

SerialPortSelector_SerialPort0	
NUM_SERIALPORTSELECTOR	

13.8.1.153 spinSerialPortSourceEnums

```
enum spinSerialPortSourceEnums
```

< Specifies the physical input Line on which to receive serial data.

Enumerator

SerialPortSource_Line0	
SerialPortSource_Line1	
SerialPortSource_Line2	
SerialPortSource_Line3	
SerialPortSource_Off	
NUM_SERIALPORTSOURCE	

13.8.1.154 spinSerialPortStopBitsEnums

```
enum spinSerialPortStopBitsEnums
```

< This feature controls the number of stop bits used by the selected serial port.

Enumerator

SerialPortStopBits_Bits1	
SerialPortStopBits_Bits1AndAHalf	
SerialPortStopBits_Bits2	
NUM_SERIALPORTSTOPBITS	

13.8.1.155 spinSoftwareSignalSelectorEnums

```
enum spinSoftwareSignalSelectorEnums
```

< Selects which Software Signal features to control.

Enumerator

SoftwareSignalSelector_SoftwareSignal0	Selects the software generated signal to control.
SoftwareSignalSelector_SoftwareSignal1	Selects the software generated signal to control.
SoftwareSignalSelector_SoftwareSignal2	Selects the software generated signal to control.
NUM_SOFTWARESIGNALSELECTOR	

13.8.1.156 spinSourceSelectorEnums

```
enum spinSourceSelectorEnums
```

< Selects the source to control.

Enumerator

SourceSelector_Source0	Selects the data source 0.
SourceSelector_Source1	Selects the data source 1.
SourceSelector_Source2	Selects the data source 2.
SourceSelector_All	Selects all the data sources.
NUM_SOURCESELECTOR	

13.8.1.157 spinTestPatternEnums

```
enum spinTestPatternEnums
```

< Selects the type of test pattern that is generated by the device as image source.

Enumerator

TestPattern_Off	Test pattern is disabled.
TestPattern_Increment	Pixel value increments by 1 for each pixel.
TestPattern_SensorTestPattern	A test pattern generated by the image sensor. The pattern varies for different sensor models.
NUM_TESTPATTERN	

13.8.1.158 spinTestPatternGeneratorSelectorEnums

```
enum spinTestPatternGeneratorSelectorEnums
```

< Selects which test pattern generator is controlled by the TestPattern feature.

Enumerator

TestPatternGeneratorSelector_Sensor	TestPattern feature controls the sensor`s test pattern generator.
TestPatternGeneratorSelector_PipelineStart	TestPattern feature controls the test pattern inserted at the start of the image pipeline.
NUM_TESTPATTERNGENERATORSELECTOR	

13.8.1.159 spinTimerSelectorEnums

```
enum spinTimerSelectorEnums
```

< Selects which Timer to configure.

Enumerator

TimerSelector_Timer0	Selects the Timer 0.
TimerSelector_Timer1	Selects the Timer 1.
TimerSelector_Timer2	Selects the Timer 2.
NUM_TIMERSELECTOR	

13.8.1.160 spinTimerStatusEnums

```
enum spinTimerStatusEnums
```

< Returns the current status of the Timer.

Enumerator

TimerStatus_TimerIdle	The Timer is idle.
TimerStatus_TimerTriggerWait	The Timer is waiting for a start trigger.
TimerStatus_TimerActive	The Timer is counting for the specified duration.
TimerStatus_TimerCompleted	The Timer reached the TimerDuration count.
NUM_TIMERSTATUS	

13.8.1.161 spinTimerTriggerActivationEnums

```
enum spinTimerTriggerActivationEnums
```

< Selects the activation mode of the trigger to start the Timer.

Enumerator

TimerTriggerActivation_RisingEdge	Starts counting on the Rising Edge of the selected trigger signal.
TimerTriggerActivation_FallingEdge	Starts counting on the Falling Edge of the selected trigger signal.
TimerTriggerActivation_AnyEdge	Starts counting on the Falling or Rising Edge of the selected trigger signal.
TimerTriggerActivation_LevelHigh	Counts as long as the selected trigger signal level is High.
TimerTriggerActivation_LevelLow	Counts as long as the selected trigger signal level is Low.
NUM_TIMERTRIGGERACTIVATION	

13.8.1.162 spinTimerTriggerSourceEnums

```
enum spinTimerTriggerSourceEnums
```

< Selects the source of the trigger to start the Timer.

Enumerator

TimerTriggerSource_Off	Disables the Timer trigger.
TimerTriggerSource_AcquisitionTrigger	Starts with the reception of the Acquisition Trigger.
TimerTriggerSource_AcquisitionStart	Starts with the reception of the Acquisition Start.
TimerTriggerSource_AcquisitionEnd	Starts with the reception of the Acquisition End.
TimerTriggerSource_FrameTrigger	Starts with the reception of the Frame Start Trigger.
TimerTriggerSource_FrameStart	Starts with the reception of the Frame Start.
TimerTriggerSource_FrameEnd	Starts with the reception of the Frame End.
TimerTriggerSource_FrameBurstStart	Starts with the reception of the Frame Burst Start.
TimerTriggerSource_FrameBurstEnd	Starts with the reception of the Frame Burst End.
TimerTriggerSource_LineTrigger	Starts with the reception of the Line Start Trigger.
TimerTriggerSource_LineStart	Starts with the reception of the Line Start.
TimerTriggerSource_LineEnd	Starts with the reception of the Line End.
TimerTriggerSource_ExposureStart	Starts with the reception of the Exposure Start.
TimerTriggerSource_ExposureEnd	Starts with the reception of the Exposure End.
TimerTriggerSource_Line0	Starts when the specidfied TimerTriggerActivation condition is met on the chosen I/O Line.
TimerTriggerSource_Line1	Starts when the specidfied TimerTriggerActivation condition is met on the chosen I/O Line.
TimerTriggerSource_Line2	Starts when the specidfied TimerTriggerActivation condition is met on the chosen I/O Line.
TimerTriggerSource_UserOutput0	Specifies which User Output bit signal to use as internal source for the trigger.
TimerTriggerSource_UserOutput1	Specifies which User Output bit signal to use as internal source for the trigger.
TimerTriggerSource_UserOutput2	Specifies which User Output bit signal to use as internal source for the trigger.
TimerTriggerSource_Counter0Start	Starts with the reception of the Counter Start.
TimerTriggerSource_Counter1Start	Starts with the reception of the Counter Start.
TimerTriggerSource_Counter2Start	Starts with the reception of the Counter Start.
TimerTriggerSource_Counter0End	Starts with the reception of the Counter End.
TimerTriggerSource_Counter1End	Starts with the reception of the Counter End.
TimerTriggerSource_Counter2End	Starts with the reception of the Counter End.
TimerTriggerSource_Timer0Start	Starts with the reception of the Timer Start.
TimerTriggerSource_Timer1Start	Starts with the reception of the Timer Start.
TimerTriggerSource_Timer2Start	Starts with the reception of the Timer Start.
TimerTriggerSource_Timer0End	Starts with the reception of the Timer End. Note that a timer can retrigger itself to achieve a free running Timer.
TimerTriggerSource_Timer1End	Starts with the reception of the Timer End. Note that a timer can retrigger itself to achieve a free running Timer.
TimerTriggerSource_Timer2End	Starts with the reception of the Timer End. Note that a timer can retrigger itself to achieve a free running Timer.

Enumerator

TimerTriggerSource_Encoder0	Starts with the reception of the Encoder output signal.
TimerTriggerSource_Encoder1	Starts with the reception of the Encoder output signal.
TimerTriggerSource_Encoder2	Starts with the reception of the Encoder output signal.
TimerTriggerSource_SoftwareSignal0	Starts on the reception of the Software Signal.
TimerTriggerSource_SoftwareSignal1	Starts on the reception of the Software Signal.
TimerTriggerSource_SoftwareSignal2	Starts on the reception of the Software Signal.
TimerTriggerSource_Action0	Starts with the assertion of the chosen action signal.
TimerTriggerSource_Action1	Starts with the assertion of the chosen action signal.
TimerTriggerSource_Action2	Starts with the assertion of the chosen action signal.
TimerTriggerSource_LinkTrigger0	Starts with the reception of the chosen Link Trigger.
TimerTriggerSource_LinkTrigger1	Starts with the reception of the chosen Link Trigger.
TimerTriggerSource_LinkTrigger2	Starts with the reception of the chosen Link Trigger.
NUM_TIMERTRIGGERSOURCE	

13.8.1.163 spinTransferComponentSelectorEnums

```
enum spinTransferComponentSelectorEnums
```

< Selects the color component for the control of the TransferStreamChannel feature.

Enumerator

TransferComponentSelector_Red	The TransferStreamChannel feature controls the index of the stream channel for the streaming of the red plane of the planar pixel formats.
TransferComponentSelector_Green	The TransferStreamChannel feature controls the index of the stream channel for the streaming of the green plane of the planar pixel formats.
TransferComponentSelector_Blue	The TransferStreamChannel feature controls the index of the stream channel for the streaming of blue plane of the planar pixel formats.
TransferComponentSelector_All	The TransferStreamChannel feature controls the index of the stream channel for the streaming of all the planes of the planar pixel formats simultaneously or non planar pixel formats.
NUM_TRANSFERCOMPONENTSELECTOR	

13.8.1.164 spinTransferControlModeEnums

```
enum spinTransferControlModeEnums
```

< Selects the control method for the transfers. Basic and Automatic start transmitting data as soon as there is enough data to fill a link layer packet. User Controlled allows you to directly control the transfer of blocks.

Enumerator

TransferControlMode_Basic	Basic
TransferControlMode_Automatic	Automatic
TransferControlMode_UserControlled	User Controlled
NUM_TRANSFERCONTROLMODE	

13.8.1.165 spinTransferOperationModeEnums

```
enum spinTransferOperationModeEnums
```

< Selects the operation mode of the transfer. Continuous is similar to Basic/Automatic but you can start/stop the transfer while acquisition runs independently. Multi Block transmits a specified number of blocks and then stops.

Enumerator

TransferOperationMode_Continuous	Continuous
TransferOperationMode_MultiBlock	Multi Block
NUM_TRANSFEROPERATIONMODE	

13.8.1.166 spinTransferQueueModeEnums

```
enum spinTransferQueueModeEnums
```

< Specifies the operation mode of the transfer queue.

Enumerator

TransferQueueMode_FirstInFirstOut	Blocks first In are transferred Out first.
NUM_TRANSFERQUEUEMODE	

13.8.1.167 spinTransferSelectorEnums

```
enum spinTransferSelectorEnums
```

< Selects which stream transfers are currently controlled by the selected Transfer features.

Enumerator

TransferSelector_Stream0	The transfer features control the data stream 0.
TransferSelector_Stream1	The transfer features control the data stream 1.
TransferSelector_Stream2	The transfer features control the data stream 2.
TransferSelector_All	The transfer features control all the data streams simulateneously.
NUM_TRANSFERSELECTOR	

13.8.1.168 spinTransferStatusSelectorEnums

```
enum spinTransferStatusSelectorEnums
```

< Selects which status of the transfer module to read.

Enumerator

TransferStatusSelector_Streaming	Data blocks are transmitted when enough data is available.
TransferStatusSelector_Paused	Data blocks transmission is suspended immediately.
TransferStatusSelector_Stopping	Data blocks transmission is stopping. The current block transmission will be completed and the transfer state will stop.
TransferStatusSelector_Stopped	Data blocks transmission is stopped.
TransferStatusSelector_QueueOverflow	Data blocks queue is in overflow state.
NUM_TRANSFERSTATUSSELECTOR	

13.8.1.169 spinTransferTriggerActivationEnums

```
enum spinTransferTriggerActivationEnums
```

< Specifies the activation mode of the transfer control trigger.

Enumerator

TransferTriggerActivation_RisingEdge	Specifies that the trigger is considered valid on the rising edge of the source signal.
TransferTriggerActivation_FallingEdge	Specifies that the trigger is considered valid on the falling edge of the source signal.
TransferTriggerActivation_AnyEdge	Specifies that the trigger is considered valid on the falling or rising edge of the source signal.
TransferTriggerActivation_LevelHigh	Specifies that the trigger is considered valid as long as the level of the source signal is high. This can apply to TransferActive and TransferPause trigger.
TransferTriggerActivation_LevelLow	Specifies that the trigger is considered valid as long as the level of the source signal is low. This can apply to TransferActive and TransferPause trigger.
NUM_TRANSFERTRIGGERACTIVATION	

13.8.1.170 spinTransferTriggerModeEnums

```
enum spinTransferTriggerModeEnums
```

< Controls if the selected trigger is active.

Enumerator

TransferTriggerMode_Off	Disables the selected trigger.
TransferTriggerMode_On	Enable the selected trigger.
NUM_TRANSFERTRIGGERMODE	

13.8.1.171 spinTransferTriggerSelectorEnums

```
enum spinTransferTriggerSelectorEnums
```

< Selects the type of transfer trigger to configure.

Enumerator

TransferTriggerSelector_TransferStart	Selects a trigger to start the transfers.
TransferTriggerSelector_TransferStop	Selects a trigger to stop the transfers.
TransferTriggerSelector_TransferAbort	Selects a trigger to abort the transfers.
TransferTriggerSelector_TransferPause	Selects a trigger to pause the transfers.
TransferTriggerSelector_TransferResume	Selects a trigger to Resume the transfers.
TransferTriggerSelector_TransferActive	Selects a trigger to Activate the transfers. This trigger type is used when TriggerActivation is set LevelHigh or levelLow.
TransferTriggerSelector_TransferBurstStart	Selects a trigger to start the transfer of a burst of frames specified by TransferBurstCount.
TransferTriggerSelector_TransferBurstStop	Selects a trigger to end the transfer of a burst of frames.
NUM_TRANSFERTRIGGERSELECTOR	

13.8.1.172 spinTransferTriggerSourceEnums

```
enum spinTransferTriggerSourceEnums
```

< Specifies the signal to use as the trigger source for transfers.

Enumerator

TransferTriggerSource_Line0	Specifies which physical line (or pin) and associated I/O control block to use as external source for the transfer control trigger signal.
TransferTriggerSource_Line1	Specifies which physical line (or pin) and associated I/O control block to use as external source for the transfer control trigger signal.
TransferTriggerSource_Line2	Specifies which physical line (or pin) and associated I/O control block to use as external source for the transfer control trigger signal.
TransferTriggerSource_Counter0Start	Specifies which of the Counter signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_Counter1Start	Specifies which of the Counter signal to use as internal source for the transfer control trigger signal.

Enumerator

TransferTriggerSource_Counter2Start	Specifies which of the Counter signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_Counter0End	Specifies which of the Counter signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_Counter1End	Specifies which of the Counter signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_Counter2End	Specifies which of the Counter signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_Timer0Start	Specifies which Timer signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_Timer1Start	Specifies which Timer signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_Timer2Start	Specifies which Timer signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_Timer0End	Specifies which Timer signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_Timer1End	Specifies which Timer signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_Timer2End	Specifies which Timer signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_SoftwareSignal0	Specifies which Software Signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_SoftwareSignal1	Specifies which Software Signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_SoftwareSignal2	Specifies which Software Signal to use as internal source for the transfer control trigger signal.
TransferTriggerSource_Action0	Specifies which Action command to use as internal source for the transfer control trigger signal.
TransferTriggerSource_Action1	Specifies which Action command to use as internal source for the transfer control trigger signal.
TransferTriggerSource_Action2	Specifies which Action command to use as internal source for the transfer control trigger signal.
NUM_TRANSFERTRIGGERSOURCE	

13.8.1.173 spinTriggerActivationEnums

```
enum spinTriggerActivationEnums
```

< Specifies the activation mode of the trigger.

Enumerator

TriggerActivation_LevelLow	
TriggerActivation_LevelHigh	
TriggerActivation_FallingEdge	
TriggerActivation_RisingEdge	
TriggerActivation_AnyEdge	
NUM_TRIGGERACTIVATION	

13.8.1.174 spinTriggerModeEnums

```
enum spinTriggerModeEnums
```

< Controls whether or not trigger is active.

Enumerator

TriggerMode_Off	
TriggerMode_On	
NUM_TRIGGERMODE	

13.8.1.175 spinTriggerOverlapEnums

```
enum spinTriggerOverlapEnums
```

< Specifies the overlap mode of the trigger.

Enumerator

TriggerOverlap_Off	
TriggerOverlap_ReadOut	
TriggerOverlap_PreviousFrame	
NUM_TRIGGEROVERLAP	

13.8.1.176 spinTriggerSelectorEnums

```
enum spinTriggerSelectorEnums
```

< Selects the type of trigger to configure.

Enumerator

TriggerSelector_AcquisitionStart	
TriggerSelector_FrameStart	
TriggerSelector_FrameBurstStart	
NUM_TRIGGERSELECTOR	

13.8.1.177 spinTriggerSourceEnums

```
enum spinTriggerSourceEnums
```

< Specifies the internal signal or physical input line to use as the trigger source.

Enumerator

TriggerSource_Software	
TriggerSource_Line0	
TriggerSource_Line1	
TriggerSource_Line2	
TriggerSource_Line3	
TriggerSource_UserOutput0	
TriggerSource_UserOutput1	
TriggerSource_UserOutput2	
TriggerSource_UserOutput3	
TriggerSource_Counter0Start	
TriggerSource_Counter1Start	
TriggerSource_Counter0End	
TriggerSource_Counter1End	
TriggerSource_LogicBlock0	
TriggerSource_LogicBlock1	
TriggerSource_Action0	
NUM_TRIGGERSOURCE	

13.8.1.178 spinUserOutputSelectorEnums

```
enum spinUserOutputSelectorEnums
```

< Selects which bit of the User Output register is set by UserOutputValue.

Enumerator

UserOutputSelector_UserOutput0	
UserOutputSelector_UserOutput1	
UserOutputSelector_UserOutput2	
UserOutputSelector_UserOutput3	
NUM_USEROUTPUTSELECTOR	

13.8.1.179 spinUserSetDefaultEnums

```
enum spinUserSetDefaultEnums
```

< Selects the feature User Set to load and make active by default when the device is restarted.

Enumerator

UserSetDefault_Default	Factory default set.
UserSetDefault_UserSet0	User configurable set 0.
UserSetDefault_UserSet1	User configurable set 1.
NUM_USERSETDEFAULT	

13.8.1.180 spinUserSetSelectorEnums

```
enum spinUserSetSelectorEnums
```

< Selects the feature User Set to load, save or configure.

Enumerator

UserSetSelector_Default	Factory default set.
UserSetSelector_UserSet0	User configurable set 0.
UserSetSelector_UserSet1	User configurable set 1.
NUM_USERSETSELECTOR	

13.8.1.181 spinWhiteClipSelectorEnums

```
enum spinWhiteClipSelectorEnums
```

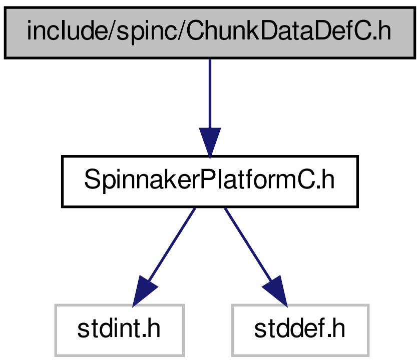
< Selects which White Clip to control.

Enumerator

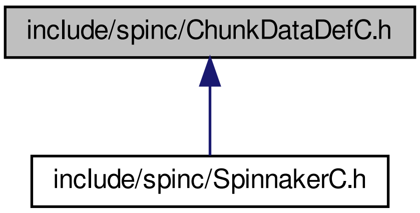
WhiteClipSelector_All	White Clip will be applied to all channels or taps.
WhiteClipSelector_Red	White Clip will be applied to the red channel.
WhiteClipSelector_Green	White Clip will be applied to the green channel.
WhiteClipSelector_Blue	White Clip will be applied to the blue channel.
WhiteClipSelector_Y	White Clip will be applied to Y channel.
WhiteClipSelector_U	White Clip will be applied to U channel.
WhiteClipSelector_V	White Clip will be applied to V channel.
WhiteClipSelector_Tap1	White Clip will be applied to Tap 1.
WhiteClipSelector_Tap2	White Clip will be applied to Tap 2.
NUM_WHITECLIPSELECTOR	

13.9 include/spinc/ChunkDataDefC.h File Reference

Include dependency graph for ChunkDataDefC.h:



This graph shows which files directly or indirectly include this file:



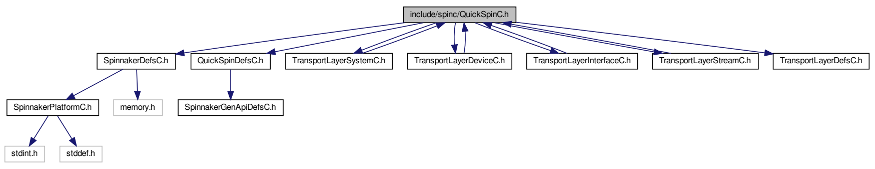
Data Structures

- struct spinChunkData

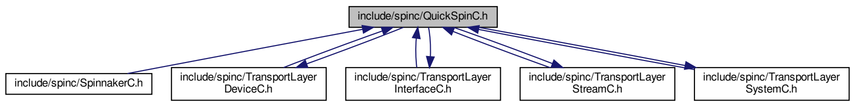
  The type of information that can be obtained from image chunk data.

13.10 include/spinc/QuickSpinC.h File Reference

Include dependency graph for QuickSpinC.h:



This graph shows which files directly or indirectly include this file:


include/spinc/QuickSpinC.h
include/spinc/SpinnakerC.h
include/spinc/TransportLayer DeviceC.h
include/spinc/TransportLayer InterfaceC.h
include/spinc/TransportLayer StreamC.h
include/spinc/TransportLayer SystemC.h

Functions

- SPINNAKERC_API quickSpinInit (spinCamera hCamera, quickSpin ∗pQuickSpin)
- SPINNAKERC_API quickSpinInitEx (spinCamera hCamera, quickSpin ∗pQuickSpin, quickSpinTLDevice ∗pQuickSpinTLDevice, quickSpinTLStream ∗pQuickSpinTLStream)
- SPINNAKERC_API quickSpinTLDeviceInit (spinCamera hCamera, quickSpinTLDevice ∗pQuickSpin←↩ TLDevice)
- SPINNAKERC_API quickSpinTLStreamInit (spinCamera hCamera, quickSpinTLStream ∗pQuickSpin←↩ TLStream)
- SPINNAKERC_API quickSpinTLInterfaceInit (spinInterface hInterface, quickSpinTLInterface ∗pQuickSpin←↩ TLInterface)
- SPINNAKERC_API quickSpinTLSystemInit (spinSystem hSystem, quickSpinTLSystem ∗pQuickSpin←↩ TLSystem)

13.10.1 Function Documentation

13.10.1.1 quickSpinInit()

```
SPINNAKERC_API quickSpinInit (
             spinCamera hCamera,
             quickSpin ∗ pQuickSpin )
```

13.10.1.2 quickSpinInitEx()

```
SPINNAKERC_API quickSpinInitEx (
            spinCamera hCamera,
            quickSpin * pQuickSpin,
            quickSpinTLDevice * pQuickSpinTLDevice,
            quickSpinTLStream * pQuickSpinTLStream )
```

13.10.1.3 quickSpinTLDeviceInit()

```
SPINNAKERC_API quickSpinTLDeviceInit (
            spinCamera hCamera,
            quickSpinTLDevice * pQuickSpinTLDevice )
```

13.10.1.4 quickSpinTLInterfaceInit()

```
SPINNAKERC_API quickSpinTLInterfaceInit (
            spinInterface hInterface,
            quickSpinTLInterface * pQuickSpinTLInterface )
```

13.10.1.5 quickSpinTLStreamInit()

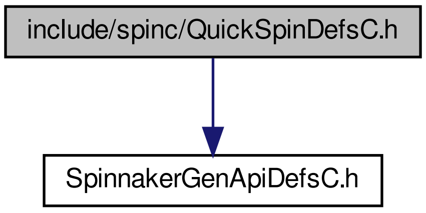
```
SPINNAKERC_API quickSpinTLStreamInit (
            spinCamera hCamera,
            quickSpinTLStream * pQuickSpinTLStream )
```

13.10.1.6 quickSpinTLSystemInit()

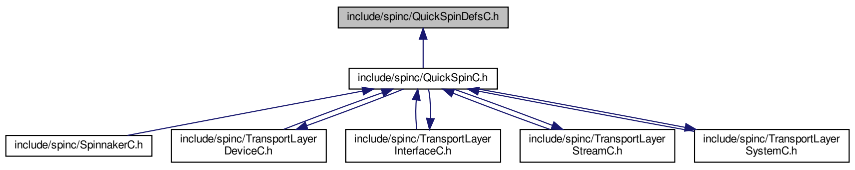
```
SPINNAKERC_API quickSpinTLSystemInit (
            spinSystem hSystem,
            quickSpinTLSystem * pQuickSpinTLSystem )
```

13.11 include/spinc/QuickSpinDefsC.h File Reference

Include dependency graph for QuickSpinDefsC.h:



This graph shows which files directly or indirectly include this file:


include/spinc/QuickSpinDefsC.h
include/spinc/QuickSpinC.h
include/spinc/SpinnakerC.h
include/spinc/TransportLayer DeviceC.h
include/spinc/TransportLayer InterfaceC.h
include/spinc/TransportLayer StreamC.h
include/spinc/TransportLayer SystemC.h

Data Structures

- struct quickSpin

Typedefs

- typedef spinNodeHandle quickSpinStringNode
- typedef spinNodeHandle quickSpinIntegerNode
- typedef spinNodeHandle quickSpinFloatNode
- typedef spinNodeHandle quickSpinBooleanNode
- typedef spinNodeHandle quickSpinEnumerationNode
- typedef spinNodeHandle quickSpinCommandNode
- typedef spinNodeHandle quickSpinRegisterNode

13.11.1 Typedef Documentation

13.11.1.1 quickSpinBooleanNode

```
typedef spinNodeHandle quickSpinBooleanNode
```

13.11.1.2 quickSpinCommandNode

```
typedef spinNodeHandle quickSpinCommandNode
```

13.11.1.3 quickSpinEnumerationNode

```
typedef spinNodeHandle quickSpinEnumerationNode
```

13.11.1.4 quickSpinFloatNode

```
typedef spinNodeHandle quickSpinFloatNode
```

13.11.1.5 quickSpinIntegerNode

```
typedef spinNodeHandle quickSpinIntegerNode
```

13.11.1.6 quickSpinRegisterNode

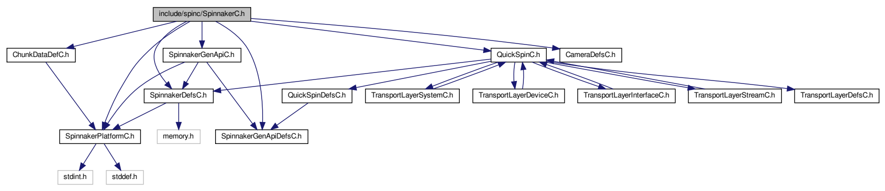
```
typedef spinNodeHandle quickSpinRegisterNode
```

13.11.1.7 quickSpinStringNode

```
typedef spinNodeHandle quickSpinStringNode
```

13.12 include/spinc/SpinnakerC.h File Reference

Include dependency graph for SpinnakerC.h:



Functions

- SPINNAKERC_API spinErrorGetLast (spinError ∗pError)

  Retrieves the error code of the last error.
- SPINNAKERC_API spinErrorGetLastMessage (char ∗pBuf, size_t ∗pBufLen)

  Retrieves the error message of the last error.
- SPINNAKERC_API spinErrorGetLastBuildDate (char ∗pBuf, size_t ∗pBufLen)

  Retrieves the build date of the last error.
- SPINNAKERC_API spinErrorGetLastBuildTime (char ∗pBuf, size_t ∗pBufLen)

  Retrieves the build time of the last error.
- SPINNAKERC_API spinErrorGetLastFileName (char ∗pBuf, size_t ∗pBufLen)

  Retrieves the filename of the last error.
- SPINNAKERC_API spinErrorGetLastFullMessage (char ∗pBuf, size_t ∗pBufLen)

  Retrieves the full error message of the last error.
- SPINNAKERC_API spinErrorGetLastFunctionName (char ∗pBuf, size_t ∗pBufLen)

  Retrieves the function name of the last error.
- SPINNAKERC_API spinErrorGetLastLineNumber (int64_t ∗pLineNum)

  Retrieves the line number of the last error.
- SPINNAKERC_API spinSystemGetInstance (spinSystem ∗phSystem)

  Retrieves an instance of the system object; the system is a singleton, so there will only ever be one instance; system instance must be destroyed by calling spinSystemReleaseInstance.
- SPINNAKERC_API spinSystemReleaseInstance (spinSystem hSystem)

  Releases the system; make sure handle is cleaned up properly by setting it to NULL after system is released; the handle can only be used again after calling spinSystemGetInstance.
- SPINNAKERC_API spinSystemGetInterfaces (spinSystem hSystem, spinInterfaceList hInterfaceList)

  Retrieves a list of detected (and enumerable) interfaces on the system; interface lists must be created and destroyed.
- SPINNAKERC_API spinSystemGetCameras (spinSystem hSystem, spinCameraList hCameraList)

  Retrieves a list of detected (and enumerable) cameras on the system; camera lists must be created and destroyed.
- SPINNAKERC_API spinSystemGetCamerasEx (spinSystem hSystem, bool8_t bUpdateInterfaces, bool8_t bUpdateCameras, spinCameraList hCameraList)

  Retrieves a list of detected (and enumerable) cameras on the system; manually set whether to update the current interface and camera lists; camera lists must be created and destroyed.
- SPINNAKERC_API spinSystemSetLoggingLevel (spinSystem hSystem, spinnakerLogLevel logLevel)

  Sets the logging level for all logging events on the system.
- SPINNAKERC_API spinSystemGetLoggingLevel (spinSystem hSystem, spinnakerLogLevel ∗pLogLevel)

  Retrieves the logging level for all logging events on the system.
- SPINNAKERC_API spinSystemRegisterLogEventHandler (spinSystem hSystem, spinLogEventHandler h↩ LogEventHandler)

Registers a logging event handler to the system (event handlers registered in this way must be unregistered)
- SPINNAKERC_API spinSystemUnregisterLogEventHandler (spinSystem hSystem, spinLogEventHandler hLogEventHandler)

  Unregisters a selected logging event handler from the system.
- SPINNAKERC_API spinSystemUnregisterAllLogEventHandlers (spinSystem hSystem)

  Unregisters all logging event handlers from the system.
- SPINNAKERC_API spinSystemIsInUse (spinSystem hSystem, bool8_t ∗pbIsInUse)

  Checks whether a system is currently in use.
- SPINNAKERC_API spinSystemRegisterDeviceArrivalEventHandler (spinSystem hSystem, spinDeviceArrivalEventHandler hDeviceArrivalEventHandler)

  Registers a device arrival event handler to every interface on the system (event handlers registered this way must be unregistered)
- SPINNAKERC_API spinSystemRegisterDeviceRemovalEventHandler (spinSystem hSystem, spinDeviceRemovalEventHandler hDeviceRemovalEventHandler)

  Registers a device removal event handler to the system to every interface on the system (event handlers registered this way must be unregistered)
- SPINNAKERC_API spinSystemUnregisterDeviceArrivalEventHandler (spinSystem hSystem, spinDeviceArrivalEventHandler hDeviceArrivalEventHandler)

  Unregisters a device arrival event handler from the system.
- SPINNAKERC_API spinSystemUnregisterDeviceRemovalEventHandler (spinSystem hSystem, spinDeviceRemovalEventHand hDeviceRemovalEventHandler)

  Unregisters a device removal event handler from the system.
- SPINNAKERC_API spinSystemRegisterInterfaceEventHandler (spinSystem hSystem, spinInterfaceEventHandler hInterfaceEventHandler)

  Registers an interface event handler (device arrival and device removal) to every interface on the system (interface events registered this way must be unregistered) If new interfaces are detected by the system after spinSystemRegisterInterfaceEventHandler() is called, those interfaces will be automatically registered with this event.
- SPINNAKERC_API spinSystemUnregisterInterfaceEventHandler (spinSystem hSystem, spinInterfaceEventHandler hInterfaceEventHandler)

  Unregisters an interface event handler from the system.
- SPINNAKERC_API spinSystemUpdateCameras (spinSystem hSystem, bool8_t ∗pbChanged)

  Updates the list of cameras on the system, informing whether there has been any changes.
- SPINNAKERC_API spinSystemUpdateCamerasEx (spinSystem hSystem, bool8_t bUpdateInterfaces, bool8_t ∗pbChanged)

  Updates the list of cameras on the system, informing whether there has been any changes; manually set whether to update the current interface lists.
- SPINNAKERC_API spinSystemSendActionCommand (spinSystem hSystem, size_t iDeviceKey, size_t i←GroupKey, size_t iGroupMask, size_t iActionTime, size_t ∗piResultSize, actionCommandResult results[ ])

  Broadcast an Action Command to all devices on system.
- SPINNAKERC_API spinSystemGetLibraryVersion (spinSystem hSystem, spinLibraryVersion ∗hLibrary←Version)

  Get current library version of Spinnaker.
- SPINNAKERC_API spinSystemGetTLNodeMap (spinSystem hSystem, spinNodeMapHandle ∗phNodeMap)

  Retrieves the transport layer nodemap from the system.
- SPINNAKERC_API spinInterfaceListCreateEmpty (spinInterfaceList ∗phInterfaceList)

  Creates an empty interface list (interface lists created this way must be destroyed)
- SPINNAKERC_API spinInterfaceListDestroy (spinInterfaceList hInterfaceList)

  Destroys an interface list.
- SPINNAKERC_API spinInterfaceListGetSize (spinInterfaceList hInterfaceList, size_t ∗pSize)

  Retrieves the number of interfaces in an interface list.
- SPINNAKERC_API spinInterfaceListGet (spinInterfaceList hInterfaceList, size_t index, spinInterface ∗ph←Interface)

  Retrieves an interface from an interface list using an index (interfaces retrieved this way must be released)

- SPINNAKERC_API spinInterfaceListClear (spinInterfaceList hInterfaceList)

    Clears an interface list.
- SPINNAKERC_API spinCameraListCreateEmpty (spinCameraList ∗phCameraList)

    Creates an empty camera list (camera lists created this way must be destroyed)
- SPINNAKERC_API spinCameraListDestroy (spinCameraList hCameraList)

    Destroys a camera list.
- SPINNAKERC_API spinCameraListGetSize (spinCameraList hCameraList, size_t ∗pSize)

    Retrieves the number of cameras on a camera list.
- SPINNAKERC_API spinCameraListGet (spinCameraList hCameraList, size_t index, spinCamera ∗ph←
Camera)

    Retrieves a camera from a camera list using an index.
- SPINNAKERC_API spinCameraListClear (spinCameraList hCameraList)

    Clears a camera list.
- SPINNAKERC_API spinCameraListRemove (spinCameraList hCameraList, size_t index)

    Removes a camera from a camera list using its index.
- SPINNAKERC_API spinCameraListAppend (spinCameraList hCameraListBase, spinCameraList hCamera←
ListToAppend)

    Appends all the cameras from one camera list to another.
- SPINNAKERC_API spinCameraListGetBySerial (spinCameraList hCameraList, const char ∗pSerial,
spinCamera ∗phCamera)

    Retrieves a camera from a camera list using its serial number.
- SPINNAKERC_API spinCameraListRemoveBySerial (spinCameraList hCameraList, const char ∗pSerial)

    Removes a camera from a camera list using its serial number.
- SPINNAKERC_API spinImageListCreateEmpty (spinImageList ∗phImageList)

    Creates an empty image list (image lists created this way must be destroyed)
- SPINNAKERC_API spinImageListDestroy (spinImageList hImageList)

    Destroys a image list.
- SPINNAKERC_API spinImageListGetSize (spinImageList hImageList, size_t ∗pSize)

    Retrieves the number of images in an image list.
- SPINNAKERC_API spinImageListGet (spinImageList hImageList, size_t index, spinImage ∗phImage)

    Retrieves a image from a image list using an index.
- SPINNAKERC_API spinImageListClear (spinImageList hImageList)

    Clears a image list.
- SPINNAKERC_API spinImageListRemove (spinImageList hImageList, size_t index)

    Removes a image from a image list using its index.
- SPINNAKERC_API spinImageListAppend (spinImageList hImageListBase, spinImageList hImageListTo←
Append)

    Appends all the images from one image list to another.
- SPINNAKERC_API spinImageListGetByPixelFormat (spinImageList hImageList, spinPixelFormatEnums
pixelFormat, spinImage ∗phImage)

    Retrieves a image from a image list given its pixel format.
- SPINNAKERC_API spinImageListRemoveByPixelFormat (spinImageList hImageList, spinPixelFormatEnums
pixelFormat)

    Removes a image from a image list using its pixel format.
- SPINNAKERC_API spinImageListRelease (spinImageList hImageList)
- SPINNAKERC_API spinImageListSave (spinImageList hImageList, const char ∗fileName)

    Saves an image list as an object to a file.
- SPINNAKERC_API spinImageListLoad (spinImageList ∗phImageList, const char ∗fileName)

    Creates an image list object from file.
- SPINNAKERC_API spinInterfaceUpdateCameras (spinInterface hInterface, bool8_t ∗pbChanged)

    Checks whether any cameras have been connected or disconnected on an interface.

- SPINNAKERC_API spinInterfaceGetCameras (spinInterface hInterface, spinCameraList hCameraList)

    Retrieves a camera list from an interface; camera lists must be created and destroy.
- SPINNAKERC_API spinInterfaceGetCamerasEx (spinInterface hInterface, bool8_t bUpdateCameras, spinCameraList hCameraList)

    Retrieves a camera list from an interface; manually set whether to update the cameras; camera lists must be created and destroyed.
- SPINNAKERC_API spinInterfaceGetTLNodeMap (spinInterface hInterface, spinNodeMapHandle ∗phNode↩ Map)

    Retrieves the transport layer nodemap from an interface.
- SPINNAKERC_API spinInterfaceRegisterDeviceArrivalEventHandler (spinInterface hInterface, spinDeviceArrivalEventHandler hDeviceArrivalEventHandler)

    Registers a device arrival event handler on an interface (event handlers registered in this way must be unregistered)
- SPINNAKERC_API spinInterfaceRegisterDeviceRemovalEventHandler (spinInterface hInterface, spinDeviceRemovalEventHan hDeviceRemovalEventHandler)

    Registers a device removal event handler on an interface (event handlers registered in this way must be unregistered)
- SPINNAKERC_API spinInterfaceUnregisterDeviceArrivalEventHandler (spinInterface hInterface, spinDeviceArrivalEventHandle hDeviceArrivalEventHandler)

    Unregisters a device arrival event handler from an interface.
- SPINNAKERC_API spinInterfaceUnregisterDeviceRemovalEventHandler (spinInterface hInterface, spinDeviceRemovalEventHa hDeviceRemovalEventHandler)

    Unregisters a device removal event handler from an interface.
- SPINNAKERC_API spinInterfaceRegisterInterfaceEventHandler (spinInterface hInterface, spinInterfaceEventHandler hInterfaceEventHandler)

    Registers an interface event handler (both device arrival and device removal) on an interface.
- SPINNAKERC_API spinInterfaceUnregisterInterfaceEventHandler (spinInterface hInterface, spinInterfaceEventHandler hInterfaceEventHandler)

    Unregisters an interface event handler from an interface.
- SPINNAKERC_API spinInterfaceRelease (spinInterface hInterface)

    Releases an interface.
- SPINNAKERC_API spinInterfaceIsInUse (spinInterface hInterface, bool8_t ∗pbIsInUse)

    Checks whether an interface is in use.
- SPINNAKERC_API spinInterfaceSendActionCommand (spinInterface hInterface, size_t iDeviceKey, size_↩ t iGroupKey, size_t iGroupMask, size_t iActionTime, size_t ∗piResultSize, actionCommandResult results[ ])

    Broadcast an Action Command to all devices on interface.
- SPINNAKERC_API spinCameraInit (spinCamera hCamera)

    Initializes a camera, allowing for much more interaction.
- SPINNAKERC_API spinCameraDeInit (spinCamera hCamera)

    Deinitializes a camera, greatly reducing functionality.
- SPINNAKERC_API spinCameraGetNodeMap (spinCamera hCamera, spinNodeMapHandle ∗phNodeMap)

    Retrieves the GenICam nodemap from a camera.
- SPINNAKERC_API spinCameraGetTLDeviceNodeMap (spinCamera hCamera, spinNodeMapHandle ∗ph↩ NodeMap)

    Retrieves the transport layer device nodemap from a camera.
- SPINNAKERC_API spinCameraGetTLStreamNodeMap (spinCamera hCamera, spinNodeMapHandle ∗ph↩ NodeMap)

    Retrieves the transport layer stream nodemap from a camera.
- SPINNAKERC_API spinCameraGetAccessMode (spinCamera hCamera, spinAccessMode ∗pAccessMode)

    Retrieves the access mode of a camera (as an enum, spinAccessMode)
- SPINNAKERC_API spinCameraReadPort (spinCamera hCamera, uint64_t iAddress, void ∗pBuffer, size_t iSize)
- SPINNAKERC_API spinCameraWritePort (spinCamera hCamera, uint64_t iAddress, void ∗pBuffer, size_t iSize)
- SPINNAKERC_API spinCameraBeginAcquisition (spinCamera hCamera)

Has a camera start acquiring images.
- SPINNAKERC_API spinCameraEndAcquisition (spinCamera hCamera)

  Has a camera stop acquiring images.
- SPINNAKERC_API spinCameraGetNextImage (spinCamera hCamera, spinImage ∗phImage)

  Retrieves an image from a camera.
- SPINNAKERC_API spinCameraGetNextImageEx (spinCamera hCamera, uint64_t grabTimeout, spinImage ∗phImage)

  Retrieves an image from a camera; manually set the timeout in milliseconds.
- SPINNAKERC_API spinCameraGetNextImageSync (spinCamera hCamera, uint64_t grabTimeout, spinImageList ∗phImageList)

  If a camera supports one or more streams, this function gets one image from each of the camera's streams, and returns the images in a list.
- SPINNAKERC_API spinCameraGetUniqueID (spinCamera hCamera, char ∗pBuf, size_t ∗pBufLen)

  Retrieves a unique identifier for a camera.
- SPINNAKERC_API spinCameraIsStreaming (spinCamera hCamera, bool8_t ∗pbIsStreaming)

  Checks whether a camera is currently acquiring images.
- SPINNAKERC_API spinCameraGetGuiXml (spinCamera hCamera, char ∗pBuf, size_t ∗pBufLen)

  Retrieves the GUI XML from a camera.
- SPINNAKERC_API spinCameraRegisterDeviceEventHandler (spinCamera hCamera, spinDeviceEventHandler hDeviceEventHandler)

  Registers a universal device event handler (every device event type) to a camera.
- SPINNAKERC_API spinCameraRegisterDeviceEventHandlerEx (spinCamera hCamera, spinDeviceEventHandler hDeviceEventHandler, const char ∗pName)

  Registers a specific device event handler (only one device event type) to a camera.
- SPINNAKERC_API spinCameraUnregisterDeviceEventHandler (spinCamera hCamera, spinDeviceEventHandler hDeviceEventHandler)

  Unregisters a device event handler from a camera.
- SPINNAKERC_API spinCameraRegisterImageEventHandler (spinCamera hCamera, spinImageEventHandler hImageEventHandler)

  Registers an image event handler to a camera.
- SPINNAKERC_API spinCameraRegisterImageEventHandlerEx (spinCamera hCamera, spinImageEventHandler hImageEventHandler, uint64_t streamIndex)

  Registers an image event handler to a camera Registers a specific stream handler for the camera given a stream index.
- SPINNAKERC_API spinCameraUnregisterImageEventHandler (spinCamera hCamera, spinImageEventHandler hImageEventHandler)

  Unregisters an image event handler from a camera.
- SPINNAKERC_API spinCameraRegisterImageListEventHandler (spinCamera hCamera, spinImageListEventHandler hImageListEventHandler)

  Registers an image list event handler to a camera.
- SPINNAKERC_API spinCameraUnregisterImageListEventHandler (spinCamera hCamera, spinImageListEventHandler hImageListEventHandler)

  Unregisters an image list event handler from a camera.
- SPINNAKERC_API spinCameraRelease (spinCamera hCamera)

  Releases a camera.
- SPINNAKERC_API spinCameraIsValid (spinCamera hCamera, bool8_t ∗pbValid)

  Checks whether a camera is still valid for use.
- SPINNAKERC_API spinCameraIsInitialized (spinCamera hCamera, bool8_t ∗pbInit)

  Checks whether a camera is currently initialized.
- SPINNAKERC_API spinCameraDiscoverMaxPacketSize (spinCamera hCamera, unsigned int ∗pMax←PacketSize)

  Returns the largest packet size that can be safely used on the interface that device is connected to.
- SPINNAKERC_API spinCameraForceIP ()

Forces the camera to be on the same subnet as its corresponding interface.

- SPINNAKERC_API spinImageCreateEmpty (spinImage ∗phImage)

  Creates an empty image; images created this way must be destroyed.
- SPINNAKERC_API spinImageCreate (spinImage hSrcImage, spinImage ∗phDestImage)

  Creates an image from another; images created this way must be destroyed.
- SPINNAKERC_API spinImageCreateEx (spinImage ∗phImage, size_t width, size_t height, size_t offsetX, size_t offsetY, spinPixelFormatEnums pixelFormat, void ∗pData)

  Creates an image with some set properties; images created this way must be destroyed.
- SPINNAKERC_API spinImageCreateEx2 (spinImage ∗phImage, size_t width, size_t height, size_t offsetX, size_t offsetY, spinPixelFormatEnums pixelFormat, void ∗pData, spinTLPayloadType dataPayloadType, size_t dataSize)

  Creates an image with some set properties; images created this way must be destroyed.
- SPINNAKERC_API spinImageDestroy (spinImage hImage)

  Destroys an image.
- SPINNAKERC_API spinImageGetColorProcessing (spinImage hImage, spinColorProcessingAlgorithm ∗p←Algorithm)

  Retrieves the color processing algorithm of a specific image.
- SPINNAKERC_API spinImageReset (spinImage hImage, size_t width, size_t height, size_t offsetX, size_t offsetY, spinPixelFormatEnums pixelFormat)

  Resets an image with some set properties.
- SPINNAKERC_API spinImageResetEx (spinImage hImage, size_t width, size_t height, size_t offsetX, size←_t offsetY, spinPixelFormatEnums pixelFormat, void ∗pData)

  Resets an image with some set properties and image data.
- SPINNAKERC_API spinImageGetID (spinImage hImage, uint64_t ∗pId)

  Retrieves the ID of an image.
- SPINNAKERC_API spinImageGetData (spinImage hImage, void ∗∗ppData)

  Retrieves the image data of an image.
- SPINNAKERC_API spinImageGetPrivateData (spinImage hImage, void ∗∗ppData)

  Retrieves the private data of an image.
- SPINNAKERC_API spinImageGetBufferSize (spinImage hImage, size_t ∗pSize)

  Retrieves the buffer size of an image.
- SPINNAKERC_API spinImageDeepCopy (spinImage hSrcImage, spinImage hDestImage)

  Creates a deep copy of an image (the destination image must be created as an empty image prior to the deep copy)
- SPINNAKERC_API spinImageGetWidth (spinImage hImage, size_t ∗pWidth)

  Retrieves the width of an image.
- SPINNAKERC_API spinImageGetHeight (spinImage hImage, size_t ∗pHeight)

  Retrieves the height of an image.
- SPINNAKERC_API spinImageGetOffsetX (spinImage hImage, size_t ∗pOffsetX)

  Retrieves the offset of an image along its X axis.
- SPINNAKERC_API spinImageGetOffsetY (spinImage hImage, size_t ∗pOffsetY)

  Retrieves the offset of an image along its Y axis.
- SPINNAKERC_API spinImageGetPaddingX (spinImage hImage, size_t ∗pPaddingX)

  Retrieves the padding of an image along its X axis.
- SPINNAKERC_API spinImageGetPaddingY (spinImage hImage, size_t ∗pPaddingY)

  Retrieves the padding of an image along its Y axis.
- SPINNAKERC_API spinImageGetFrameID (spinImage hImage, uint64_t ∗pFrameID)

  Retrieves the frame ID of an image.
- SPINNAKERC_API spinImageGetTimeStamp (spinImage hImage, uint64_t ∗pTimeStamp)

  Retrieves the timestamp of an image.
- SPINNAKERC_API spinImageGetPayloadType (spinImage hImage, size_t ∗pPayloadType)

  Retrieves the payload type of an image (as an enum, spinPayloadTypeInfoIds)

- SPINNAKERC_API spinImageGetTLPayloadType (spinImage hImage, spinTLPayloadType ∗pPayloadType)

    Retrieves the transport layer payload type of an image (as an enum, spinPayloadTypeInfoIds)
- SPINNAKERC_API spinImageGetPixelFormat (spinImage hImage, spinPixelFormatEnums ∗pPixelFormat)

    Retrieves the pixel format of an image (as an enum, spinPixelFormatEnums)
- SPINNAKERC_API spinImageGetTLPixelFormat (spinImage hImage, uint64_t ∗pPixelFormat)

    Retrieves the transport layer pixel format of an image (as an unsigned integer)
- SPINNAKERC_API spinImageGetTLPixelFormatNamespace (spinImage hImage, spinTLPixelFormatNamespace ∗pPixelFormatNamespace)

    Retrieves the transport layer pixel format namespace of an image (as an enum, spinPixelFormatNamespaceID)
- SPINNAKERC_API spinImageGetPixelFormatName (spinImage hImage, char ∗pBuf, size_t ∗pBufLen)

    Retrieves the pixel format of an image (as a symbolic)
- SPINNAKERC_API spinImageIsIncomplete (spinImage hImage, bool8_t ∗pbIsIncomplete)

    Checks whether an image is incomplete.
- SPINNAKERC_API spinImageGetValidPayloadSize (spinImage hImage, size_t ∗pSize)

    Retrieves the valid payload size of an image.
- SPINNAKERC_API spinImageSave (spinImage hImage, const char ∗pFilename, spinImageFileFormat format)

    Saves an image using a specified file format (using an enum, spinImageFileFormat)
- SPINNAKERC_API spinImageSaveFromExt (spinImage hImage, const char ∗pFilename)

    Saves an image using a specified file format (using the extension of the filename)
- SPINNAKERC_API spinImageSavePng (spinImage hImage, const char ∗pFilename, const spinPNGOption ∗pOption)

    Saves an image as a PNG image.
- SPINNAKERC_API spinImageSavePpm (spinImage hImage, const char ∗pFilename, const spinPPMOption ∗pOption)

    Saves an image as a PPM image.
- SPINNAKERC_API spinImageSavePgm (spinImage hImage, const char ∗pFilename, const spinPGMOption ∗pOption)

    Saves an image as an PGM image.
- SPINNAKERC_API spinImageSaveTiff (spinImage hImage, const char ∗pFilename, const spinTIFFOption ∗pOption)

    Saves an image as a TIFF image.
- SPINNAKERC_API spinImageSaveJpeg (spinImage hImage, const char ∗pFilename, const spinJPEGOption ∗pOption)

    Saves an image as a JPEG image.
- SPINNAKERC_API spinImageSaveJpg2 (spinImage hImage, const char ∗pFilename, const spinJPG2Option ∗pOption)

    Saves an image as a JPEG 2000 image.
- SPINNAKERC_API spinImageSaveBmp (spinImage hImage, const char ∗pFilename, const spinBMPOption ∗pOption)

    Saves an image as a BMP image.
- SPINNAKERC_API spinImageGetChunkLayoutID (spinImage hImage, uint64_t ∗pId)

    Retrieves the chunk layout ID of an image.
- SPINNAKERC_API spinImageCalculateStatistics (spinImage hImage, const spinImageStatistics hStatistics)

    Calculates the image statistics of an image.
- SPINNAKERC_API spinImageGetStatus (spinImage hImage, spinImageStatus ∗pStatus)

    Retrieves the image status of an image.
- SPINNAKERC_API spinImageGetStatusDescription (spinImageStatus status, char ∗pBuf, size_t ∗pBufLen)

    Retrieves the description of image status.
- SPINNAKERC_API spinImageRelease (spinImage hImage)

    Releases an image.
- SPINNAKERC_API spinImageHasCRC (spinImage hImage, bool8_t ∗pbHasCRC)

Checks whether an image has CRC.

- SPINNAKERC_API spinImageCheckCRC (spinImage hImage, bool8_t ∗pbCheckCRC)

  Checks whether the CRC of an image is correct.
- SPINNAKERC_API spinImageGetBitsPerPixel (spinImage hImage, size_t ∗pBitsPerPixel)

  Retrieves the number of bits per pixel of an image.
- SPINNAKERC_API spinImageGetSize (spinImage hImage, size_t ∗pImageSize)

  Retrieves the size of an image.
- SPINNAKERC_API spinImageGetStride (spinImage hImage, size_t ∗pStride)

  Retrieves the stride of an image.
- SPINNAKERC_API spinImageProcessorCreate (spinImageProcessor ∗phImageProcessor)

  Creates an image processor.
- SPINNAKERC_API spinImageProcessorDestroy (spinImageProcessor hImageProcessor)

  Destroys a image list.
- SPINNAKERC_API spinImageProcessorSetColorProcessing (spinImageProcessor hImageProcessor, spinColorProcessingAlgorithm colorAlgorithm)

  Sets the color processing algorithm used at the time of the spinImageProcessorConvert() call, therefore the most recent execution of this function will take precedence.
- SPINNAKERC_API spinImageProcessorGetColorProcessing (spinImageProcessor hImageProcessor, spinColorProcessingAlgorithm ∗pColorAlgorithm)

  Gets the default color processing algorithm.
- SPINNAKERC_API spinImageProcessorSetNumDecompressionThreads (spinImageProcessor hImage↩ Processor, unsigned int numThreads)

  Sets the default number of threads used for image decompression during spinImageProcessorConvert().
- SPINNAKERC_API spinImageProcessorGetNumDecompressionThreads (spinImageProcessor hImage↩ Processor, unsigned int ∗pNumThreads)

  Gets the number of threads used for image decompression during spinImageProcessorConvert().
- SPINNAKERC_API spinImageProcessorConvert (spinImageProcessor hImageProcessor, spinImage hSrc↩ Image, spinImage hDestImage, spinPixelFormatEnums destFormat)

  Converts the source image buffer to the specified destination pixel format and stores the result in the destination image.
- SPINNAKERC_API spinImageProcessorConvertImageList (spinImageProcessor hImageProcessor, spinImageList hSrcImageList, spinImage hDestImage, spinPixelFormatEnums destFormat)

  Converts the source list of image buffers to the specified output pixel format and returns the result in a new image.
- SPINNAKERC_API spinImageProcessorApplyGamma (spinImageProcessor hImageProcessor, spinImage hSrcImage, spinImage hDestImage, float gamma, bool8_t applyGammaInverse)

  Applies gamma correction to the source image and stores the result in the destination image.
- SPINNAKERC_API spinDeviceEventHandlerCreate (spinDeviceEventHandler ∗phDeviceEventHandler, spinDeviceEventFunction pFunction, void ∗pUserData)

  Creates a device event handler.
- SPINNAKERC_API spinDeviceEventHandlerDestroy (spinDeviceEventHandler hDeviceEventHandler)

  Destroys a device event handler.
- SPINNAKERC_API spinImageEventHandlerCreate (spinImageEventHandler ∗phImageEventHandler, spinImageEventFunction pFunction, void ∗pUserData)

  Creates an image event handler.
- SPINNAKERC_API spinImageEventHandlerDestroy (spinImageEventHandler hImageEventHandler)

  Destroys an image event handler.
- SPINNAKERC_API spinImageListEventHandlerCreate (spinImageListEventHandler ∗phImageEventHandler, spinImageListEventFunction pFunction, void ∗pUserData)

  Creates an image list event handler.
- SPINNAKERC_API spinImageListEventHandlerDestroy (spinImageListEventHandler hImageListEvent↩ Handler)

  Destroys an image list event handler.

- SPINNAKERC_API spinDeviceArrivalEventHandlerCreate (spinDeviceArrivalEventHandler ∗phDevice↩ArrivalEventHandler, spinArrivalEventFunction pFunction, void ∗pUserData)

  Creates a device arrival event handler.
- SPINNAKERC_API spinDeviceArrivalEventHandlerDestroy (spinDeviceArrivalEventHandler hDevice↩ArrivalEventHandler)

  Destroys a device arrival event handler.
- SPINNAKERC_API spinDeviceRemovalEventHandlerCreate (spinDeviceRemovalEventHandler ∗ph↩DeviceRemovalEventHandler, spinRemovalEventFunction pFunction, void ∗pUserData)

  Creates a device removal event handler.
- SPINNAKERC_API spinDeviceRemovalEventHandlerDestroy (spinDeviceRemovalEventHandler hDevice↩RemovalEventHandler)

  Destroys a device removal event handler.
- SPINNAKERC_API spinInterfaceEventHandlerCreate (spinInterfaceEventHandler ∗phInterfaceEvent↩Handler, spinArrivalEventFunction pArrivalFunction, spinRemovalEventFunction pRemovalFunction, void ∗pUserData)

  Creates an interface event handler (both device arrival and device removal)
- SPINNAKERC_API spinInterfaceEventHandlerDestroy (spinInterfaceEventHandler hInterfaceEventHandler)

  Destroys an interface event handler (both device arrival and device removal)
- SPINNAKERC_API spinLogEventHandlerCreate (spinLogEventHandler ∗phLogEventHandler, spinLogEventFunction pFunction, void ∗pUserData)

  Creates a log event handler.
- SPINNAKERC_API spinLogEventHandlerDestroy (spinLogEventHandler hLogEventHandler)

  Destroys a log event handler.
- SPINNAKERC_API spinImageStatisticsCreate (spinImageStatistics ∗phStatistics)

  Creates an image statistics context.
- SPINNAKERC_API spinImageStatisticsDestroy (spinImageStatistics hStatistics)

  Destroys an image statistics context.
- SPINNAKERC_API spinImageStatisticsEnableAll (spinImageStatistics hStatistics)

  Enables all channels of an image statistics context.
- SPINNAKERC_API spinImageStatisticsDisableAll (spinImageStatistics hStatistics)

  Disables all channels of an image statistics context.
- SPINNAKERC_API spinImageStatisticsEnableGreyOnly (spinImageStatistics hStatistics)

  Disables all channels of an image statistics context except grey-scale.
- SPINNAKERC_API spinImageStatisticsEnableRgbOnly (spinImageStatistics hStatistics)

  Disables all channels of an image statistics context except red, blue, and green.
- SPINNAKERC_API spinImageStatisticsEnableHslOnly (spinImageStatistics hStatistics)

  Disables all channels of an image statistics context except hue, saturation, and lightness.
- SPINNAKERC_API spinImageStatisticsGetChannelStatus (spinImageStatistics hStatistics, spinStatisticsChannel channel, bool8_t ∗pbEnabled)

  Checks whether an image statistics context is enabled.
- SPINNAKERC_API spinImageStatisticsSetChannelStatus (spinImageStatistics hStatistics, spinStatisticsChannel channel, bool8_t bEnable)

  Sets the status of an image statistics channel.
- SPINNAKERC_API spinImageStatisticsGetRange (spinImageStatistics hStatistics, spinStatisticsChannel channel, unsigned int ∗pMin, unsigned int ∗pMax)

  Retrieves the range of an image statistics channel.
- SPINNAKERC_API spinImageStatisticsGetPixelValueRange (spinImageStatistics hStatistics, spinStatisticsChannel channel, unsigned int ∗pMin, unsigned int ∗pMax)

  Retrieves the pixel value range of an image statistics channel.
- SPINNAKERC_API spinImageStatisticsGetNumPixelValues (spinImageStatistics hStatistics, spinStatisticsChannel channel, unsigned int ∗pNumValues)

  Retrieves the number of pixel values of an image statistics channel.

- SPINNAKERC_API spinImageStatisticsGetMean (spinImageStatistics hStatistics, spinStatisticsChannel channel, float ∗pMean)

  Retrieves the mean of pixel values of an image statistics channel.
- SPINNAKERC_API spinImageStatisticsGetHistogram (spinImageStatistics hStatistics, spinStatisticsChannel channel, int ∗∗ppHistogram)

  Retrieves a histogram of an image statistics channel.
- SPINNAKERC_API spinImageStatisticsGetAll (spinImageStatistics hStatistics, spinStatisticsChannel channel, unsigned int ∗pRangeMin, unsigned int ∗pRangeMax, unsigned int ∗pPixelValueMin, unsigned int ∗p←PixelValueMax, unsigned int ∗pNumPixelValues, float ∗pPixelValueMean, int ∗∗ppHistogram)

  Retrieves all available information of an image statistics channel.
- SPINNAKERC_API spinLogDataGetCategoryName (spinLogEventData hLogEventData, char ∗pBuf, size_t ∗pBufLen)

  Retrieves the category name of a log event.
- SPINNAKERC_API spinLogDataGetPriority (spinLogEventData hLogEventData, int64_t ∗pValue)

  Retrieves the priority of a log event.
- SPINNAKERC_API spinLogDataGetPriorityName (spinLogEventData hLogEventData, char ∗pBuf, size_←t ∗pBufLen)

  Retrieves the priority name of a log event.
- SPINNAKERC_API spinLogDataGetTimestamp (spinLogEventData hLogEventData, char ∗pBuf, size_t ∗p←BufLen)

  Retrieves the timestamp of a log event.
- SPINNAKERC_API spinLogDataGetNDC (spinLogEventData hLogEventData, char ∗pBuf, size_t ∗pBufLen)

  Retrieves the NDC of a log event.
- SPINNAKERC_API spinLogDataGetThreadName (spinLogEventData hLogEventData, char ∗pBuf, size_←t ∗pBufLen)

  Retrieves the thread name of a log event.
- SPINNAKERC_API spinLogDataGetLogMessage (spinLogEventData hLogEventData, char ∗pBuf, size_←t ∗pBufLen)

  Retrieves the log message of a log event.
- SPINNAKERC_API spinDeviceEventGetId (spinDeviceEventData hDeviceEventData, uint64_t ∗pEventId)

  Retrieves the event ID of a device event.
- SPINNAKERC_API spinDeviceEventGetPayloadData (spinDeviceEventData hDeviceEventData, const uint8_t ∗pBuf, size_t ∗pBufSize)

  Retrieves the payload data of a device event.
- SPINNAKERC_API spinDeviceEventGetPayloadDataSize (spinDeviceEventData hDeviceEventData, size_t ∗pBufSize)

  Retrieves the payload data size of a device event.
- SPINNAKERC_API spinDeviceEventGetName (spinDeviceEventData hDeviceEventData, char ∗pBuf, size←_t ∗pBufLen)

  Retrieves the event name of a device event.
- SPINNAKERC_API spinImageChunkDataGetIntValue (spinImage hImage, const char ∗pName, int64_t ∗p←Value)
- SPINNAKERC_API spinImageChunkDataGetFloatValue (spinImage hImage, const char ∗pName, double ∗pValue)

13.12.1 Function Documentation

13.12.1.1 spinCameraBeginAcquisition()

```
SPINNAKERC_API spinCameraBeginAcquisition (
            spinCamera hCamera )
```

Has a camera start acquiring images.

See also

spinError

Parameters

hCamera	The camera to begin acquiring images

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.2 spinCameraDeInit()

```
SPINNAKERC_API spinCameraDeInit (
            spinCamera hCamera )
```

Deinitializes a camera, greatly reducing functionality.

See also

spinError

Parameters

hCamera	The camera to deinitialize

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.3 spinCameraDiscoverMaxPacketSize()

```
SPINNAKERC_API spinCameraDiscoverMaxPacketSize (
            spinCamera hCamera,
            unsigned int * pMaxPacketSize )
```

Returns the largest packet size that can be safely used on the interface that device is connected to.

See also

spinError

Parameters

hCamera	The camera to check
pMaxPacketSize	The maximum packet size returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.4 spinCameraEndAcquisition()

```
SPINNAKERC_API spinCameraEndAcquisition (
            spinCamera hCamera )
```

Has a camera stop acquiring images.

See also

spinError

Parameters

hCamera	The camera to stop acquiring images

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.5 spinCameraForceIP()

```
SPINNAKERC_API spinCameraForceIP ( )
```

Forces the camera to be on the same subnet as its corresponding interface.

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.6 spinCameraGetAccessMode()

```
SPINNAKERC_API spinCameraGetAccessMode (
            spinCamera hCamera,
            spinAccessMode * pAccessMode )
```

Retrieves the access mode of a camera (as an enum, spinAccessMode)

See also

spinError

spinAccessMode

Parameters

hCamera	The camera of the access mode to retrieve
pAccessMode	The access mode enum pointer in which the access mode is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.7 spinCameraGetGuiXml()

```
SPINNAKERC_API spinCameraGetGuiXml (
            spinCamera hCamera,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the GUI XML from a camera.

See also

spinError

Parameters

hCamera	The camera of the GUI XML to retrieve
pBuf	The c-string character buffer in which the GUI XML is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.8 spinCameraGetNextImage()

```
SPINNAKERC_API spinCameraGetNextImage (
            spinCamera hCamera,
            spinImage * phImage )
```

Retrieves an image from a camera.

See also

spinError

Parameters

hCamera	The camera of the image to retrieve
phImage	The image handle pointer in which the image is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.9 spinCameraGetNextImageEx()

```
SPINNAKERC_API spinCameraGetNextImageEx (
            spinCamera hCamera,
            uint64_t grabTimeout,
            spinImage * phImage )
```

Retrieves an image from a camera; manually set the timeout in milliseconds.

See also

spinError

Parameters

hCamera	The camera of the image to retrieve
grabTimeout	A 64bit value that represents a timeout in milliseconds
phImage	The image handle pointer in which the image is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.10 spinCameraGetNextImageSync()

```
SPINNAKERC_API spinCameraGetNextImageSync (
            spinCamera hCamera,
            uint64_t grabTimeout,
            spinImageList * phImageList )
```

If a camera supports one or more streams, this function gets one image from each of the camera's streams, and returns the images in a list.

This function will block for the specified timeout period until an image arrives on all the streams.

See also

spinCameraInit()

spinCameraBeginAcquisition()

spinCameraEndAcquisition()

Parameters

hCamera	The camera of the image to retrieve
grabTimeout	A 64bit value that represents a timeout in milliseconds

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.11 spinCameraGetNodeMap()

```
SPINNAKERC_API spinCameraGetNodeMap (
            spinCamera hCamera,
            spinNodeMapHandle * phNodeMap )
```

Retrieves the GenICam nodemap from a camera.

See also

spinError

Parameters

hCamera	The camera from which the nodemap is retrieved
phNodeMap	The nodemap handle pointer in which the nodemap is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.12 spinCameraGetTLDeviceNodeMap()

```
SPINNAKERC_API spinCameraGetTLDeviceNodeMap (
            spinCamera hCamera,
            spinNodeMapHandle * phNodeMap )
```

Retrieves the transport layer device nodemap from a camera.

See also

spinError

Parameters

hCamera	The camera from which the transport layer device nodemap is retrieved
phNodeMap	The nodemap handle pointer in which the nodemap is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.13 spinCameraGetTLStreamNodeMap()

```
SPINNAKERC_API spinCameraGetTLStreamNodeMap (
            spinCamera hCamera,
            spinNodeMapHandle * phNodeMap )
```

Retrieves the transport layer stream nodemap from a camera.

See also

spinError

Parameters

hCamera	The camera from which the transport layer streaming nodemap is retrieved
phNodeMap	The nodemap handle pointer in which the nodemap is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.14 spinCameraGetUniqueID()

```
SPINNAKERC_API spinCameraGetUniqueID (
             spinCamera hCamera,
             char * pBuf,
             size_t * pBufLen )
```

Retrieves a unique identifier for a camera.

See also

spinError

Parameters

hCamera	The camera of the unique identifier
pBuf	The c-string character buffer in which the unique identifier is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.15 spinCameraInit()

```
SPINNAKERC_API spinCameraInit (
             spinCamera hCamera )
```

Initializes a camera, allowing for much more interaction.

See also

spinError

Parameters

hCamera	The camera to initialize

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.16 spinCameraIsInitialized()

```
SPINNAKERC_API spinCameraIsInitialized (
            spinCamera hCamera,
            bool8_t * pbInit )
```

Checks whether a camera is currently initialized.

See also

spinError

Parameters

hCamera	The camera to check
pbInit	The boolean pointer to return whether or not the camera is initialized

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.17 spinCameraIsStreaming()

```
SPINNAKERC_API spinCameraIsStreaming (
            spinCamera hCamera,
            bool8_t * pbIsStreaming )
```

Checks whether a camera is currently acquiring images.

See also

spinError

Parameters

hCamera	The camera to check
pbIsStreaming	The boolean pointer to return whether or not the camera is currently streaming

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.18 spinCameraIsValid()

```
SPINNAKERC_API spinCameraIsValid (
            spinCamera hCamera,
            bool8_t * pbValid )
```

Checks whether a camera is still valid for use.

See also

spinError

Parameters

hCamera	The camera to check
pbValid	The boolean pointer to return whether or not the camera is valid

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.19 spinCameraListAppend()

```
SPINNAKERC_API spinCameraListAppend (
            spinCameraList hCameraListBase,
            spinCameraList hCameraListToAppend )
```

Appends all the cameras from one camera list to another.

See also

spinError

Parameters

hCameraListBase	The camera list to receive the other
hCameraListToAppend	The camera list to add to the other

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.20 spinCameraListClear()

```
SPINNAKERC_API spinCameraListClear (
            spinCameraList hCameraList )
```

Clears a camera list.

See also

spinError

Parameters

hCameraList	The camera list to clear

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.21 spinCameraListCreateEmpty()

```
SPINNAKERC_API spinCameraListCreateEmpty (
            spinCameraList * phCameraList )
```

Creates an empty camera list (camera lists created this way must be destroyed)

See also

spinError

Parameters

phCameraList	The camera list handle pointer in which the empty camera list is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.22 spinCameraListDestroy()

```
SPINNAKERC_API spinCameraListDestroy (
            spinCameraList hCameraList )
```

Destroys a camera list.

See also

spinError

Parameters

hCameraList	The camera list to destroy

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.23 spinCameraListGet()

```
SPINNAKERC_API spinCameraListGet (
            spinCameraList hCameraList,
            size_t index,
            spinCamera * phCamera )
```

Retrieves a camera from a camera list using an index.

This function will return a SPINNAKER_ERR_INVALID_PARAMETER error if the input index is out of range.

See also

spinError

Parameters

hCameraList	The camera list of the camera to retrieve
index	The index of the camera
phCamera	The camera handle pointer in which the camera is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.24 spinCameraListGetBySerial()

```
SPINNAKERC_API spinCameraListGetBySerial (
            spinCameraList hCameraList,
            const char * pSerial,
            spinCamera * phCamera )
```

Retrieves a camera from a camera list using its serial number.

This function will return a NULL spinCamera pointer if no matching camera serial is found.

See also

spinError

Parameters

hCameraList	The camera list of the camera to retrieve
serial	The serial number of the camera to retrieve
phCamera	The camera handle pointer in which the camera is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.25 spinCameraListGetSize()

```
SPINNAKERC_API spinCameraListGetSize (
              spinCameraList hCameraList,
              size_t * pSize )
```

Retrieves the number of cameras on a camera list.

See also

spinError

Parameters

hCameraList	The camera list where the cameras to be counted are
pSize	The unsigned integer pointer in which the number of cameras is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.26 spinCameraListRemove()

```
SPINNAKERC_API spinCameraListRemove (
              spinCameraList hCameraList,
              size_t index )
```

Removes a camera from a camera list using its index.

See also

spinError

Parameters

hCameraList	The camera list of the camera to remove
index	The index of the camera to remove

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.27 spinCameraListRemoveBySerial()

```
SPINNAKERC_API spinCameraListRemoveBySerial (
            spinCameraList hCameraList,
            const char * pSerial )
```

Removes a camera from a camera list using its serial number.

See also

spinError

Parameters

hCameraList	The camera list of the camera to remove
pSerial	The serial number of the camera to remove

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.28 spinCameraReadPort()

```
SPINNAKERC_API spinCameraReadPort (
            spinCamera hCamera,
            uint64_t iAddress,
            void * pBuffer,
            size_t iSize )
```

13.12.1.29 spinCameraRegisterDeviceEventHandler()

```
SPINNAKERC_API spinCameraRegisterDeviceEventHandler (
            spinCamera hCamera,
            spinDeviceEventHandler hDeviceEventHandler )
```

Registers a universal device event handler (every device event type) to a camera.

See also

spinError

Parameters

hCamera	The camera on which to register the universal device event handler
hDeviceEventHandler	The device event handler to register

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.30 spinCameraRegisterDeviceEventHandlerEx()

```
SPINNAKERC_API spinCameraRegisterDeviceEventHandlerEx (
            spinCamera hCamera,
            spinDeviceEventHandler hDeviceEventHandler,
            const char * pName )
```

Registers a specific device event handler (only one device event type) to a camera.

See also

spinError

Parameters

hCamera	The camera on which to register the specific device event handler
hDeviceEventHandler	The device event handler to register
pName	The name of the device event handler to register

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.31 spinCameraRegisterImageEventHandler()

```
SPINNAKERC_API spinCameraRegisterImageEventHandler (
            spinCamera hCamera,
            spinImageEventHandler hImageEventHandler )
```

Registers an image event handler to a camera.

See also

spinError

Parameters

hCamera	The camera on which to register the image event handler
hImageEventHandler	The image event handler to register

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.32 spinCameraRegisterImageEventHandlerEx()

```
SPINNAKERC_API spinCameraRegisterImageEventHandlerEx (
            spinCamera hCamera,
            spinImageEventHandler hImageEventHandler,
            uint64_t streamIndex )
```

Registers an image event handler to a camera Registers a specific stream handler for the camera given a stream index.

The camera has to be initialized first with a call to spinCameraInit() before registering handlers for events.

See also

spinError

Parameters

hCamera	The camera on which to register the image event handler
hImageEventHandler	The image event handler to register
streamIndex	The index of the stream of where this handler will be registered to

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.33 spinCameraRegisterImageListEventHandler()

```
SPINNAKERC_API spinCameraRegisterImageListEventHandler (
            spinCamera hCamera,
            spinImageListEventHandler hImageListEventHandler )
```

Registers an image list event handler to a camera.

See also

spinError

Parameters

hCamera	The camera on which to register the image event handler
hImageListEventHandler	The image list event handler to register

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.34 spinCameraRelease()

```
SPINNAKERC_API spinCameraRelease (
            spinCamera hCamera )
```

Releases a camera.

See also

spinError

Parameters

hCamera	The camera to release

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.35 spinCameraUnregisterDeviceEventHandler()

```
SPINNAKERC_API spinCameraUnregisterDeviceEventHandler (
            spinCamera hCamera,
            spinDeviceEventHandler hDeviceEventHandler )
```

Unregisters a device event handler from a camera.

See also

spinError

Parameters

hCamera	The camera from which to unregister the device event handler
hDeviceEventHandler	The device event handler to unregister

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.36 spinCameraUnregisterImageEventHandler()

```
SPINNAKERC_API spinCameraUnregisterImageEventHandler (
            spinCamera hCamera,
            spinImageEventHandler hImageEventHandler )
```

Unregisters an image event handler from a camera.

See also

spinError

Parameters

hCamera	The camera from which to unregister the image event handler
hImageEventHandler	The image event handler to unregister

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.37 spinCameraUnregisterImageListEventHandler()

```
SPINNAKERC_API spinCameraUnregisterImageListEventHandler (
            spinCamera hCamera,
            spinImageListEventHandler hImageListEventHandler )
```

Unregisters an image list event handler from a camera.

See also

spinError

Parameters

hCamera	The camera from which to unregister the image event handler
hImageEventHandler	The image event handler to unregister

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.38 spinCameraWritePort()

```
SPINNAKERC_API spinCameraWritePort (
            spinCamera hCamera,
            uint64_t iAddress,
            void * pBuffer,
            size_t iSize )
```

13.12.1.39 spinDeviceArrivalEventHandlerCreate()

```
SPINNAKERC_API spinDeviceArrivalEventHandlerCreate (
            spinDeviceArrivalEventHandler * phDeviceArrivalEventHandler,
            spinArrivalEventFunction pFunction,
            void * pUserData )
```

Creates a device arrival event handler.

See also

spinError

Parameters

phDeviceArrivalEventHandler	The device arrival event handler pointer in which the device arrival event context is created
pFunction	The function to be called at device event occurrences; signature to match: void(<em>spinArrivalEventFunction)(void pUserData)
pUserData	Properties that can be passed into the event function

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.40 spinDeviceArrivalEventHandlerDestroy()

```
SPINNAKERC_API spinDeviceArrivalEventHandlerDestroy (
            spinDeviceArrivalEventHandler hDeviceArrivalEventHandler )
```

Destroys a device arrival event handler.

See also

spinError

Parameters

hDeviceArrivalEventHandler	The device arrival event handler to destroy

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.41 spinDeviceEventGetId()

```
SPINNAKERC_API spinDeviceEventGetId (
            spinDeviceEventData hDeviceEventData,
            uint64_t * pEventId )
```

Retrieves the event ID of a device event.

See also

spinError

Parameters

hDeviceEventData	The log event data received from the log event
pEventId	The unsigned integer pointer in which the event ID is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.42 spinDeviceEventGetName()

```
SPINNAKERC_API spinDeviceEventGetName (
            spinDeviceEventData hDeviceEventData,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the event name of a device event.

See also

spinError

Parameters

hDeviceEventData	The log event data received from the log event
pBuf	The c-string character buffer in which the name of the device event is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.43 spinDeviceEventGetPayloadData()

```
SPINNAKERC_API spinDeviceEventGetPayloadData (
            spinDeviceEventData hDeviceEventData,
            const uint8_t * pBuf,
            size_t * pBufSize )
```

Retrieves the payload data of a device event.

See also

spinError

Parameters

hDeviceEventData	The log event data received from the log event
pBuf	The unsigned integer pointer in which the event payload is returned
pBufSize	The unsigned integer pointer in which the size of the payload is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.44 spinDeviceEventGetPayloadDataSize()

```
SPINNAKERC_API spinDeviceEventGetPayloadDataSize (
            spinDeviceEventData hDeviceEventData,
            size_t * pBufSize )
```

Retrieves the payload data size of a device event.

See also

spinError

Parameters

hDeviceEventData	The log event data received from the log event
pBufSize	The unsigned integer pointer in which the size of the payload is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.45 spinDeviceEventHandlerCreate()

```
SPINNAKERC_API spinDeviceEventHandlerCreate (
            spinDeviceEventHandler * phDeviceEventHandler,
            spinDeviceEventFunction pFunction,
            void * pUserData )
```

Creates a device event handler.

See also

spinError

Parameters

phDeviceEventHandler	The device event handler pointer in which the device event context is created
pFunction	The function to be called at device event occurrences; signature to match: void(<em>spinDeviceEventFunction)(const spinDeviceEventData hEventData, const char pEventName, void∗ pUserData)
pUserData	Properties that can be passed into the event function

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.46 spinDeviceEventHandlerDestroy()

```
SPINNAKERC_API spinDeviceEventHandlerDestroy (
            spinDeviceEventHandler hDeviceEventHandler )
```

Destroys a device event handler.

See also

spinError

Parameters

hDeviceEventHandler	The device event handler to destroy

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.47 spinDeviceRemovalEventHandlerCreate()

```
SPINNAKERC_API spinDeviceRemovalEventHandlerCreate (
            spinDeviceRemovalEventHandler * phDeviceRemovalEventHandler,
            spinRemovalEventFunction pFunction,
            void * pUserData )
```

Creates a device removal event handler.

See also

spinError

Parameters

phDeviceRemovalEventHandler	The device removal event handler pointer in which the device removal event context is created
pFunction	The function to be called at device event occurrences; signature to match: void(<em>spinRemovalEventFunction)(uint64_t deviceSerialNumber, void pUserData)
pUserData	Properties that can be passed into the event function

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.48 spinDeviceRemovalEventHandlerDestroy()

```
SPINNAKERC_API spinDeviceRemovalEventHandlerDestroy (
            spinDeviceRemovalEventHandler hDeviceRemovalEventHandler )
```

Destroys a device removal event handler.

See also

spinError

Parameters

hDeviceRemovalEventHandler	The device removal event handler to destroy

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.49 spinErrorGetLast()

```
SPINNAKERC_API spinErrorGetLast (
            spinError * pError )
```

Retrieves the error code of the last error.

See also

spinError

Parameters

pError	The error enum pointer in which the error message is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.50 spinErrorGetLastBuildDate()

```
SPINNAKERC_API spinErrorGetLastBuildDate (
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the build date of the last error.

See also

spinError

Parameters

pBuf	The c-string character buffer in which the build date is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.51 spinErrorGetLastBuildTime()

```
SPINNAKERC_API spinErrorGetLastBuildTime (
             char * pBuf,
             size_t * pBufLen )
```

Retrieves the build time of the last error.

See also

spinError

Parameters

pBuf	The c-string character buffer in which the build time is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.52 spinErrorGetLastFileName()

```
SPINNAKERC_API spinErrorGetLastFileName (
             char * pBuf,
             size_t * pBufLen )
```

Retrieves the filename of the last error.

See also

spinError

Parameters

pBuf	The c-string character buffer in which the file name is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.53 spinErrorGetLastFullMessage()

```
SPINNAKERC_API spinErrorGetLastFullMessage (
             char * pBuf,
             size_t * pBufLen )
```

Retrieves the full error message of the last error.

See also

spinError

Parameters

pBuf	The c-string character buffer in which the full error message is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.54 spinErrorGetLastFunctionName()

```
SPINNAKERC_API spinErrorGetLastFunctionName (
             char * pBuf,
             size_t * pBufLen )
```

Retrieves the function name of the last error.

See also

spinError

Parameters

pBuf	The c-string character buffer in which the function name is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.55 spinErrorGetLastLineNumber()

```
SPINNAKERC_API spinErrorGetLastLineNumber (
            int64_t * pLineNum )
```

Retrieves the line number of the last error.

See also

spinError

Parameters

pBuf	The c-string character buffer in which the line number is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.56 spinErrorGetLastMessage()

```
SPINNAKERC_API spinErrorGetLastMessage (
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the error message of the last error.

See also

spinError

Parameters

pBuf	The c-string character buffer in which the error message is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.57 spinImageCalculateStatistics()

```
SPINNAKERC_API spinImageCalculateStatistics (
            spinImage hImage,
            const spinImageStatistics hStatistics )
```

Calculates the image statistics of an image.

See also

spinError

Parameters

hImage	The image to be saved
hStatistics	The image statistics context in which the calculated statistics are returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.58 spinImageCheckCRC()

```
SPINNAKERC_API spinImageCheckCRC (
            spinImage hImage,
            bool8_t * pbCheckCRC )
```

Checks whether the CRC of an image is correct.

See also

spinError

Parameters

hImage	The image to be saved
pbCheckCRC	The boolean pointer to return whether the image CRC passes

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.59 spinImageChunkDataGetFloatValue()

```
SPINNAKERC_API spinImageChunkDataGetFloatValue (
             spinImage hImage,
             const char * pName,
             double * pValue )
```

13.12.1.60 spinImageChunkDataGetIntValue()

```
SPINNAKERC_API spinImageChunkDataGetIntValue (
             spinImage hImage,
             const char * pName,
             int64_t * pValue )
```

13.12.1.61 spinImageCreate()

```
SPINNAKERC_API spinImageCreate (
             spinImage hSrcImage,
             spinImage * phDestImage )
```

Creates an image from another; images created this way must be destroyed.

See also

spinError

Parameters

hSrcImage	The image to be copied
phDestImage	The image handle pointer of the image to be created

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.62 spinImageCreateEmpty()

```
SPINNAKERC_API spinImageCreateEmpty (
            spinImage * phImage )
```

Creates an empty image; images created this way must be destroyed.

See also

spinError

Parameters

phImage	The image handle pointer in which the empty image is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.63 spinImageCreateEx()

```
SPINNAKERC_API spinImageCreateEx (
            spinImage * phImage,
            size_t width,
            size_t height,
            size_t offsetX,
            size_t offsetY,
            spinPixelFormatEnums pixelFormat,
            void * pData )
```

Creates an image with some set properties; images created this way must be destroyed.

See also

spinError

Parameters

phImage	The image handle pointer in which the image is returned
width	The width to set
height	The height to set
offsetX	The offset along the X axis to set
offsetY	The offset along the Y axis to set
pixelFormat	The pixel format to set
pData	The image data to set; can be set to null

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.64 spinImageCreateEx2()

```
SPINNAKERC_API spinImageCreateEx2 (
              spinImage * phImage,
              size_t width,
              size_t height,
              size_t offsetX,
              size_t offsetY,
              spinPixelFormatEnums pixelFormat,
              void * pData,
              spinTLPayloadType dataPayloadType,
              size_t dataSize )
```

Creates an image with some set properties; images created this way must be destroyed.

See also

spinError

spinImageGetTLPayloadType

Parameters

phImage	The image handle pointer in which the image is returned
width	The width to set
height	The height to set
offsetX	The offset along the X axis to set
offsetY	The offset along the Y axis to set
pixelFormat	The pixel format to set
pData	The image data to set; can be set to null
dataPayloadType	The payload type of the data. This value can be retrieved from an existing image by using the spinImageGetTLPayloadType() function call.
dataSize	The size of the provided data in bytes

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.65 spinImageDeepCopy()

```
SPINNAKERC_API spinImageDeepCopy (
              spinImage hSrcImage,
              spinImage hDestImage )
```

Creates a deep copy of an image (the destination image must be created as an empty image prior to the deep copy)

See also

spinError

Parameters

hSrcImage	The image to be copied
hDestImage	The image handle in which the image is copied

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.66 spinImageDestroy()

```
SPINNAKERC_API spinImageDestroy (
             spinImage hImage )
```

Destroys an image.

See also

spinError

Parameters

hImage	The image to destroy

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.67 spinImageEventHandlerCreate()

```
SPINNAKERC_API spinImageEventHandlerCreate (
             spinImageEventHandler * phImageEventHandler,
             spinImageEventFunction pFunction,
             void * pUserData )
```

Creates an image event handler.

See also

spinError

Parameters

phImageEventHandler	The image event handler pointer in which the image event context is created
pFunction	The function to be called at image event occurrences; signature to match: void(<em>spinImageEventFunction)(const spinImage hImage, void pUserData)
pUserData	Properties that can be passed into the event function

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.68 spinImageEventHandlerDestroy()

```
SPINNAKERC_API spinImageEventHandlerDestroy (
            spinImageEventHandler hImageEventHandler )
```

Destroys an image event handler.

See also

spinError

Parameters

hImageEventHandler	The image event handler to destroy

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.69 spinImageGetBitsPerPixel()

```
SPINNAKERC_API spinImageGetBitsPerPixel (
            spinImage hImage,
            size_t * pBitsPerPixel )
```

Retrieves the number of bits per pixel of an image.

See also

spinError

Parameters

hImage	The image to be saved
pBitsPerPixel	The unsigned integer pointer in which the number of bits per pixel is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.70 spinImageGetBufferSize()

```
SPINNAKERC_API spinImageGetBufferSize (
            spinImage hImage,
            size_t * pSize )
```

Retrieves the buffer size of an image.

See also

spinError

Parameters

hImage	The image of image data buffer to retrieve
pSize	The unsigned integer pointer in which the size of the image data if returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.71 spinImageGetChunkLayoutID()

```
SPINNAKERC_API spinImageGetChunkLayoutID (
            spinImage hImage,
            uint64_t * pId )
```

Retrieves the chunk layout ID of an image.

See also

spinError

Parameters

hImage	The image to be saved
pId	The unsigned integer pointer in which the chunk layout ID is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.72 spinImageGetColorProcessing()

```
SPINNAKERC_API spinImageGetColorProcessing (
            spinImage hImage,
            spinColorProcessingAlgorithm * pAlgorithm )
```

Retrieves the color processing algorithm of a specific image.

See also

spinError

Parameters

hImage	The image of the color processing algorithm to retrieve
pAlgorithm	The color processing algorithm pointer in which the color processing algorithm is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.73 spinImageGetData()

```
SPINNAKERC_API spinImageGetData (
            spinImage hImage,
            void ** ppData )
```

Retrieves the image data of an image.

See also

spinError

Parameters

hImage	The image of the image data to retrieve
ppData	The pointer to the void pointer in which the image data is retrieved

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.74 spinImageGetFrameID()

```
SPINNAKERC_API spinImageGetFrameID (
             spinImage hImage,
             uint64_t * pFrameID )
```

Retrieves the frame ID of an image.

See also

spinError

Parameters

hImage	The image of the frame ID to retrieve
pFrameID	The unsigned integer pointer in which the frame ID is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.75 spinImageGetHeight()

```
SPINNAKERC_API spinImageGetHeight (
             spinImage hImage,
             size_t * pHeight )
```

Retrieves the height of an image.

See also

spinError

Parameters

hImage	The image of the height to retrieve
pHeight	The unsigned integer pointer in which the height is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.76 spinImageGetID()

```
SPINNAKERC_API spinImageGetID (
            spinImage hImage,
            uint64_t * pId )
```

Retrieves the ID of an image.

See also

spinError

Parameters

hImage	The image of the ID to retrieve
pId	The unsigned integer pointer in which the ID is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.77 spinImageGetOffsetX()

```
SPINNAKERC_API spinImageGetOffsetX (
            spinImage hImage,
            size_t * pOffsetX )
```

Retrieves the offset of an image along its X axis.

See also

spinError

Parameters

hImage	The image of the offset along the X axis to retrieve
pOffsetX	The unsigned integer pointer in which the offset along the X axis is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.78 spinImageGetOffsetY()

```
SPINNAKERC_API spinImageGetOffsetY (
            spinImage hImage,
            size_t * pOffsetY )
```

Retrieves the offset of an image along its Y axis.

See also

spinError

Parameters

hImage	The image of the offset along the Y axis to retrieve
pOffsetY	The unsigned integer pointer in which the offset along the Y axis is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.79 spinImageGetPaddingX()

```
SPINNAKERC_API spinImageGetPaddingX (
            spinImage hImage,
            size_t * pPaddingX )
```

Retrieves the padding of an image along its X axis.

See also

spinError

Parameters

hImage	The image of the padding along the X axis to retrieve
pPaddingX	The unsigned integer pointer in which the padding along the X axis is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.80 spinImageGetPaddingY()

```
SPINNAKERC_API spinImageGetPaddingY (
            spinImage hImage,
            size_t * pPaddingY )
```

Retrieves the padding of an image along its Y axis.

See also

spinError

Parameters

hImage	The image of the padding along the Y axis to retrieve
pPaddingY	The unsigned integer pointer in which the padding along the Y axis is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.81 spinImageGetPayloadType()

```
SPINNAKERC_API spinImageGetPayloadType (
            spinImage hImage,
            size_t * pPayloadType )
```

Retrieves the payload type of an image (as an enum, spinPayloadTypeInfoIds)

See also

spinError

spinPayloadTypeInfoIds

Parameters

hImage	The image of the payload type to retrieve
pPayloadType	The payload type enum pointer in which the payload type is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.82 spinImageGetPixelFormat()

```
SPINNAKERC_API spinImageGetPixelFormat (
            spinImage hImage,
            spinPixelFormatEnums * pPixelFormat )
```

Retrieves the pixel format of an image (as an enum, spinPixelFormatEnums)

See also

spinError

spinPixelFormatEnums

Parameters

hImage	The image of the pixel format to retrieve
pPixelFormat	The pixel format enum pointer in which the pixel format is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.83 spinImageGetPixelFormatName()

```
SPINNAKERC_API spinImageGetPixelFormatName (
            spinImage hImage,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the pixel format of an image (as a symbolic)

See also

spinError

Parameters

hImage	The image of the pixel format to retrieve
pBuf	The c-string character buffer in which the pixel format symbolic is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.84 spinImageGetPrivateData()

```
SPINNAKERC_API spinImageGetPrivateData (
             spinImage hImage,
             void ** ppData )
```

Retrieves the private data of an image.

See also

spinError

Parameters

hImage	The image of the private image data to retrieve
ppData	The pointer to the void pointer in which the private image data is retrieved

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.85 spinImageGetSize()

```
SPINNAKERC_API spinImageGetSize (
             spinImage hImage,
             size_t * pImageSize )
```

Retrieves the size of an image.

See also

spinError

Parameters

hImage	The image to be saved
pImageSize	The unsigned integer pointer in which the size of the image is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.86 spinImageGetStatus()

```
SPINNAKERC_API spinImageGetStatus (
            spinImage hImage,
            spinImageStatus * pStatus )
```

Retrieves the image status of an image.

See also

spinError

Parameters

hImage	The image to be saved
pStatus	The status enum pointer in which the image status is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.87 spinImageGetStatusDescription()

```
SPINNAKERC_API spinImageGetStatusDescription (
            spinImageStatus status,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the description of image status.

See also

spinError

Parameters

status	The status enum
pBuf	The c-string character buffer in which the explanation of image status enum is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length; if pBuf is NULL, minimum length of string buffer is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.88 spinImageGetStride()

```
SPINNAKERC_API spinImageGetStride (
            spinImage hImage,
            size_t * pStride )
```

Retrieves the stride of an image.

See also

spinError

Parameters

hImage	The image to be saved
pStride	The unsigned integer pointer in which the stride is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.89 spinImageGetTimeStamp()

```
SPINNAKERC_API spinImageGetTimeStamp (
            spinImage hImage,
            uint64_t * pTimeStamp )
```

Retrieves the timestamp of an image.

See also

spinError

Parameters

hImage	The image of the timestamp to retrieve
pTimeStamp	The unsigned integer pointer om which the timestamp is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.90 spinImageGetTLPayloadType()

```
SPINNAKERC_API spinImageGetTLPayloadType (
            spinImage hImage,
            spinTLPayloadType * pPayloadType )
```

Retrieves the transport layer payload type of an image (as an enum, spinPayloadTypeInfoIds)

See also

spinError

spinPayloadTypeInfoIds

Parameters

hImage	The image of the TL payload type to retrieve
pPayloadType	The payload type enum pointer in which the TL payload type is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.91 spinImageGetTLPixelFormat()

```
SPINNAKERC_API spinImageGetTLPixelFormat (
            spinImage hImage,
            uint64_t * pPixelFormat )
```

Retrieves the transport layer pixel format of an image (as an unsigned integer)

See also

spinError

Parameters

hImage	The image of the TL pixel format to retrieve
pPixelFormat	The unsigned integer pointer in which the TL pixel format is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.92 spinImageGetTLPixelFormatNamespace()

```
SPINNAKERC_API spinImageGetTLPixelFormatNamespace (
            spinImage hImage,
            spinTLPixelFormatNamespace * pPixelFormatNamespace )
```

Retrieves the transport layer pixel format namespace of an image (as an enum, spinPixelFormatNamespaceID)

See also

spinError

spinPixelFormatNamespaceID

Parameters

hImage	The image of the TL pixel format namespace to retrieve
pPixelFormatNamespace	The pixel format namespace pointer in which the pixel format namespace is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.93 spinImageGetValidPayloadSize()

```
SPINNAKERC_API spinImageGetValidPayloadSize (
            spinImage hImage,
            size_t * pSize )
```

Retrieves the valid payload size of an image.

See also

spinError

Parameters

hImage	The image of the payload size to retrieve
pSize	The unsigned integer pointer in which the size of the valid payload is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.94 spinImageGetWidth()

```
SPINNAKERC_API spinImageGetWidth (
             spinImage hImage,
             size_t * pWidth )
```

Retrieves the width of an image.

See also

spinError

Parameters

hImage	The image of the width to retrieve
pWidth	The unsigned integer pointer in which the width is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.95 spinImageHasCRC()

```
SPINNAKERC_API spinImageHasCRC (
             spinImage hImage,
             bool8_t * pbHasCRC )
```

Checks whether an image has CRC.

See also

spinError

Parameters

hImage	The image to be saved
pbHasCRC	The boolean pointer to return whether the image has CRC available

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.96 spinImageIsIncomplete()

```
SPINNAKERC_API spinImageIsIncomplete (
            spinImage hImage,
            bool8_t * pbIsIncomplete )
```

Checks whether an image is incomplete.

See also

spinError

Parameters

hImage	The image to check
pbIsIncomplete	The boolean pointer to return whether or not the image is incomplete

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.97 spinImageListAppend()

```
SPINNAKERC_API spinImageListAppend (
            spinImageList hImageListBase,
            spinImageList hImageListToAppend )
```

Appends all the images from one image list to another.

See also

spinError

Parameters

hImageListBase	The image list to receive the other
hImageListToAppend	The image list to add to the other

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.98 spinImageListClear()

```
SPINNAKERC_API spinImageListClear (
            spinImageList hImageList )
```

Clears a image list.

See also

spinError

Parameters

hImageList	The image list to clear

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.99 spinImageListCreateEmpty()

```
SPINNAKERC_API spinImageListCreateEmpty (
            spinImageList * phImageList )
```

Creates an empty image list (image lists created this way must be destroyed)

See also

spinError

Parameters

phImageList	The image list handle pointer in which the empty image list is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.100 spinImageListDestroy()

```
SPINNAKERC_API spinImageListDestroy (
            spinImageList hImageList )
```

Destroys a image list.

See also

spinError

Parameters

hImageList	The image list to destroy

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.101 spinImageListEventHandlerCreate()

```
SPINNAKERC_API spinImageListEventHandlerCreate (
            spinImageListEventHandler * phImageEventHandler,
            spinImageListEventFunction pFunction,
            void * pUserData )
```

Creates an image list event handler.

See also

spinError

Parameters

phImageListEventHandler	The image list event handler pointer in which the image list event context is created
pFunction	The function to be called at image list event occurrences; signature to match: void(<em>spinImageListEventFunction)(const spinListImage hImage, void pUserData)
pUserData	Properties that can be passed into the event function

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.102 spinImageListEventHandlerDestroy()

```
SPINNAKERC_API spinImageListEventHandlerDestroy (
            spinImageListEventHandler hImageListEventHandler )
```

Destroys an image list event handler.

See also

spinError

Parameters

hImageListEventHandler	The image list event handler to destroy

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.103 spinImageListGet()

```
SPINNAKERC_API spinImageListGet (
            spinImageList hImageList,
            size_t index,
            spinImage * phImage )
```

Retrieves a image from a image list using an index.

This function will return a SPINNAKER_ERR_INVALID_PARAMETER error if the input index is out of range.

See also

spinError

Parameters

hImageList	The image list of the image to retrieve
index	The index of the image
phImage	The image handle pointer in which the image is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.104 spinImageListGetByPixelFormat()

```
SPINNAKERC_API spinImageListGetByPixelFormat (
            spinImageList hImageList,
            spinPixelFormatEnums pixelFormat,
            spinImage * phImage )
```

Retrieves a image from a image list given its pixel format.

This function will return a NULL spinImage pointer if no matching image pixel format is found.

See also

spinError

Parameters

hImageList	The image list of the image to retrieve
pixelFormat	The pixel format of the image to retrieve
phImage	The image handle pointer in which the image is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.105 spinImageListGetSize()

```
SPINNAKERC_API spinImageListGetSize (
            spinImageList hImageList,
            size_t * pSize )
```

Retrieves the number of images in an image list.

See also

spinError

Parameters

hImageList	The image list where the images to be counted are
pSize	The unsigned integer pointer in which the number of images is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.106 spinImageListLoad()

```
SPINNAKERC_API spinImageListLoad (
            spinImageList * phImageList,
            const char * fileName )
```

Creates an image list object from file.

See also

spinImageListSave()

spinError

Parameters

phImageList	The image list handle pointer in which the empty image list is returned
fileName	Name of the file to load an image object from.

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.107 spinImageListRelease()

```
SPINNAKERC_API spinImageListRelease (
            spinImageList hImageList )
```

13.12.1.108 spinImageListRemove()

```
SPINNAKERC_API spinImageListRemove (
            spinImageList hImageList,
            size_t index )
```

Removes a image from a image list using its index.

See also

spinError

Parameters

hImageList	The image list of the camera to remove
index	The index of the image to remove

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.109 spinImageListRemoveByPixelFormat()

```
SPINNAKERC_API spinImageListRemoveByPixelFormat (
            spinImageList hImageList,
            spinPixelFormatEnums pixelFormat )
```

Removes a image from a image list using its pixel format.

See also

spinError

Parameters

hImageList	The image list of the image to remove
pixelFormat	The pixel format of the image to remove

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.110 spinImageListSave()

```
SPINNAKERC_API spinImageListSave (
            spinImageList hImageList,
            const char * fileName )
```

Saves an image list as an object to a file.

See also

spinImageListLoad()
spinError

Parameters

hImageList	The image list of the image to remove
fileName	Name of the file to save the current image list object to. It is recommended to use the file extension 'sil'.

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.111 spinImageProcessorApplyGamma()

```
SPINNAKERC_API spinImageProcessorApplyGamma (
            spinImageProcessor hImageProcessor,
            spinImage hSrcImage,
            spinImage hDestImage,
            float gamma,
            bool8_t applyGammaInverse )
```

Applies gamma correction to the source image and stores the result in the destination image.

Parameters

hImageProcessor	The image processor context
hSrcImage	The source image from which to apply gamma on.
hDestImage	The destination image in which the gamma applied image data will be stored.
gamma	Gamma value to apply. A value between 0.5 and 4 is acceptable. (Default assuming image-to-screen)
applyGammaInverse	Converts a gamma corrected source image back to the original image using the inverse of the gamma value (used for applying screen-to-image gamma)

13.12.1.112 spinImageProcessorConvert()

```
SPINNAKERC_API spinImageProcessorConvert (
            spinImageProcessor hImageProcessor,
            spinImage hSrcImage,
            spinImage hDestImage,
            spinPixelFormatEnums destFormat )
```

Converts the source image buffer to the specified destination pixel format and stores the result in the destination image.

The destination image needs to be configured to have the correct buffer size before calling this function. See spinImageReset() to setup the correct buffer size according to specified pixel format.

Note that compressed images are decompressed before any further color processing or conversion during this call. Decompression is multi-threaded and defaults to utilizing one less than the number of concurrent threads supported by the system. The default number of decompression threads can be set with spinImageProcessorSetNumDecompressionThreads().

See also

spinPixelFormatEnums
spinImageReset
spinImageProcessorSetNumDecompressionThreads

Parameters

hImageProcessor	The image processor context
srcImage	The source image from which to convert the image from.
destImage	The destination image in which the converted image data will be stored.
destFormat	Output format of the converted image.

13.12.1.113 spinImageProcessorConvertImageList()

```
SPINNAKERC_API spinImageProcessorConvertImageList (
            spinImageProcessor hImageProcessor,
            spinImageList hSrcImageList,
            spinImage hDestImage,
            spinPixelFormatEnums destFormat )
```

Converts the source list of image buffers to the specified output pixel format and returns the result in a new image.

The conversion could encompasses decompression, interleaving and conversion of image data depending on the source pixel format of images in the source image list. The destination image needs to be configured to have the correct buffer size before calling this function. See spinImageReset() to setup the correct buffer size according to specified pixel format.

Note that compressed images are decompressed before any further color processing, interleaving or conversion is performed. Decompression is multi-threaded and defaults to utilizing one less than the number of concurrent threads supported by the system. The default number of decompression threads can be set with SetNumDecompression↩Threads().

Note not all the supported image pixel formats described in the class description are supported in this function.

List of supported image pixel formats for the source image list:

- PixelFormat_R12
- PixelFormat_GR12
- PixelFormat_GB12
- PixelFormat_B12
- PixelFormat_R12_Jpeg
- PixelFormat_GR12_Jpeg
- PixelFormat_GB12_Jpeg
- PixelFormat_B12_Jpeg

See also

spinPixelFormatEnums
spinImageReset
spinImageProcessorSetNumDecompressionThreads

Parameters

hImageProcessor	The image processor context
hSrcImageList	List of images from which to convert the images from.
hDestImage	The destination image in which the converted image data will be stored.
destFormat	Output format of the converted image.

13.12.1.114 spinImageProcessorCreate()

```
SPINNAKERC_API spinImageProcessorCreate (
            spinImageProcessor * phImageProcessor )
```

Creates an image processor.

See also

spinError

Parameters

phImageProcessor	The image processor handle pointer in which the image processor context is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.115 spinImageProcessorDestroy()

```
SPINNAKERC_API spinImageProcessorDestroy (
            spinImageProcessor hImageProcessor )
```

Destroys a image list.

See also

spinError

Parameters

hImageProcessor	The image processor context to destroy

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.116 spinImageProcessorGetColorProcessing()

```
SPINNAKERC_API spinImageProcessorGetColorProcessing (
            spinImageProcessor hImageProcessor,
            spinColorProcessingAlgorithm * pColorAlgorithm )
```

Gets the default color processing algorithm.

Parameters

hImageProcessor	The image processor context
pColorAlgorithm	The color processing algorithm pointer in which the color processing algorithm is returned

See also

spinImageProcessorSetColorProcessing()

13.12.1.117 spinImageProcessorGetNumDecompressionThreads()

```
SPINNAKERC_API spinImageProcessorGetNumDecompressionThreads (
            spinImageProcessor hImageProcessor,
            unsigned int * pNumThreads )
```

Gets the number of threads used for image decompression during spinImageProcessorConvert().

Parameters

hImageProcessor	The image processor context
pNumThreads	The unsigned integer pointer in which the number of parallel image decompression threads is returned

See also

spinImageProcessorSetNumDecompressionThreads()

13.12.1.118 spinImageProcessorSetColorProcessing()

```
SPINNAKERC_API spinImageProcessorSetColorProcessing (
            spinImageProcessor hImageProcessor,
            spinColorProcessingAlgorithm colorAlgorithm )
```

Sets the color processing algorithm used at the time of the spinImageProcessorConvert() call, therefore the most recent execution of this function will take precedence.

The DEFAULT algorithm is deprecated and should not be used in the ImageProcessor class.

Parameters

hImageProcessor	The image processor context
colorAlgorithm	The color processing algorithm to set.

See also

spinImageProcessorGetColorProcessing()

13.12.1.119 spinImageProcessorSetNumDecompressionThreads()

```
SPINNAKERC_API spinImageProcessorSetNumDecompressionThreads (
            spinImageProcessor hImageProcessor,
            unsigned int numThreads )
```

Sets the default number of threads used for image decompression during spinImageProcessorConvert().

The number of threads used is defaulted to be equal to one less than the number of concurrent threads supported by the system.

Parameters

hImageProcessor	The image processor context
numThreads	Number of parallel image decompression threads set to run

See also

spinImageProcessorConvert()

13.12.1.120 spinImageRelease()

```
SPINNAKERC_API spinImageRelease (
            spinImage hImage )
```

Releases an image.

See also

spinError

Parameters

hImage	The image to be saved

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.121 spinImageReset()

```
SPINNAKERC_API spinImageReset (
            spinImage hImage,
            size_t width,
            size_t height,
            size_t offsetX,
            size_t offsetY,
            spinPixelFormatEnums pixelFormat )
```

Resets an image with some set properties.

See also

spinError

Parameters

hImage	The image to be reset
width	The width to be reset to
height	The height to be reset to
offsetX	The offset to be reset to along the X axis
offsetY	The offset to be reset to along the Y axis
pixelFormat	The pixel format to be reset to

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.122 spinImageResetEx()

```
SPINNAKERC_API spinImageResetEx (
            spinImage hImage,
            size_t width,
            size_t height,
            size_t offsetX,
            size_t offsetY,
            spinPixelFormatEnums pixelFormat,
            void * pData )
```

Resets an image with some set properties and image data.

See also

spinError

Parameters

hImage	The image to reset
width	The width to be reset to
height	The height to be reset to
offsetX	The offset to be reset to along the X axis
offsetY	The offset to be reset to along the Y axis
pixelFormat	The pixel format to be reset to
pData	The image data to reset to

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.123 spinImageSave()

```
SPINNAKERC_API spinImageSave (
            spinImage hImage,
            const char * pFilename,
            spinImageFileFormat format )
```

Saves an image using a specified file format (using an enum, spinImageFileFormat)

See also

spinError

spinImageFileFormat

Parameters

hImage	The image to be saved
pFilename	The filename to use to save the image (with or without the appropriate file extension) @Param format The file format to use to save the image

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.124 spinImageSaveBmp()

```
SPINNAKERC_API spinImageSaveBmp (
            spinImage hImage,
            const char * pFilename,
            const spinBMPOption * pOption )
```

Saves an image as a BMP image.

See also

spinError

Parameters

hImage	The image to be saved
pFilename	The filename to use to save the image (with or without the appropriate file extension)
pOption	The image options related to saving as BMP; includes whether to save as indexed 8-bit

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.125 spinImageSaveFromExt()

```
SPINNAKERC_API spinImageSaveFromExt (
            spinImage hImage,
            const char * pFilename )
```

Saves an image using a specified file format (using the extension of the filename)

See also

spinError

Parameters

hImage	The image to be saved
pFilename	The filename to use to save the image (with or without the appropriate file extension)

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.126 spinImageSaveJpeg()

```
SPINNAKERC_API spinImageSaveJpeg (
            spinImage hImage,
            const char * pFilename,
            const spinJPEGOption * pOption )
```

Saves an image as a JPEG image.

See also

spinError

Parameters

hImage	The image to be saved
pFilename	The filename to use to save the image (with or without the appropriate file extension)
pOption	The image options related to saving as JPEG; includes quality and whether to save as progressive

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.127 spinImageSaveJpg2()

```
SPINNAKERC_API spinImageSaveJpg2 (
            spinImage hImage,
            const char * pFilename,
            const spinJPG2Option * pOption )
```

Saves an image as a JPEG 2000 image.

See also

spinError

Parameters

hImage	The image to be saved
pFilename	The filename to use to save the image (with or without the appropriate file extension)
pOption	The image options related to saving as JPEG 2000; includes quality

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.128 spinImageSavePgm()

```
SPINNAKERC_API spinImageSavePgm (
            spinImage hImage,
            const char * pFilename,
            const spinPGMOption * pOption )
```

Saves an image as an PGM image.

See also

spinError

Parameters

hImage	The image to be saved
pFilename	The filename to use to save the image (with or without the appropriate file extension)
pOption	The image options related to saving as PGM; includes whether to save as binary

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.129 spinImageSavePng()

```
SPINNAKERC_API spinImageSavePng (
            spinImage hImage,
            const char * pFilename,
            const spinPNGOption * pOption )
```

Saves an image as a PNG image.

See also

spinError

Parameters

hImage	The image to be saved
pFilename	The filename to use to save the image (with or without the appropriate file extension)
pOption	The image options related to saving as PNG; includes compression level and whether to save as interlaced

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.130 spinImageSavePpm()

```
SPINNAKERC_API spinImageSavePpm (
            spinImage hImage,
            const char * pFilename,
            const spinPPMOption * pOption )
```

Saves an image as a PPM image.

See also

spinError

Parameters

hImage	The image to be saved
pFilename	The filename to use to save the image (with or without the appropriate file extension)
pOption	The image options related to saving as PPM; includes whether to save as binary

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.131 spinImageSaveTiff()

```
SPINNAKERC_API spinImageSaveTiff (
            spinImage hImage,
            const char * pFilename,
            const spinTIFFOption * pOption )
```

Saves an image as a TIFF image.

See also

spinError

Parameters

hImage	The image to be saved
pFilename	The filename to use to save the image (with or without the appropriate file extension)
pOption	The image options related to saving as TIFF; includes compression method

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.132 spinImageStatisticsCreate()

```
SPINNAKERC_API spinImageStatisticsCreate (
            spinImageStatistics * phStatistics )
```

Creates an image statistics context.

Parameters

phStatistics	The statistics handle pointer in which the image statistics context is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.133 spinImageStatisticsDestroy()

```
SPINNAKERC_API spinImageStatisticsDestroy (
            spinImageStatistics hStatistics )
```

Destroys an image statistics context.

See also

spinError

Parameters

hStatistics	The image statistics context to destroy

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.134 spinImageStatisticsDisableAll()

```
SPINNAKERC_API spinImageStatisticsDisableAll (
            spinImageStatistics hStatistics )
```

Disables all channels of an image statistics context.

See also

spinError

Parameters

hStatistics	The image statistics context to disable all channels

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.135 spinImageStatisticsEnableAll()

```
SPINNAKERC_API spinImageStatisticsEnableAll (
            spinImageStatistics hStatistics )
```

Enables all channels of an image statistics context.

See also

spinError

Parameters

hStatistics	The image statistics context to enable all channels

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.136 spinImageStatisticsEnableGreyOnly()

```
SPINNAKERC_API spinImageStatisticsEnableGreyOnly (
            spinImageStatistics hStatistics )
```

Disables all channels of an image statistics context except grey-scale.

See also

spinError

Parameters

hStatistics	The image statistics context to enable only grey

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.137 spinImageStatisticsEnableHslOnly()

```
SPINNAKERC_API spinImageStatisticsEnableHslOnly (
            spinImageStatistics hStatistics )
```

Disables all channels of an image statistics context except hue, saturation, and lightness.

See also

spinError

Parameters

hStatistics	The image statistics context to enable only HSL

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.138 spinImageStatisticsEnableRgbOnly()

```
SPINNAKERC_API spinImageStatisticsEnableRgbOnly (
            spinImageStatistics hStatistics )
```

Disables all channels of an image statistics context except red, blue, and green.

See also

spinError

Parameters

hStatistics	The image statistics context to enable only RGB

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.139 spinImageStatisticsGetAll()

```
SPINNAKERC_API spinImageStatisticsGetAll (
            spinImageStatistics hStatistics,
            spinStatisticsChannel channel,
            unsigned int * pRangeMin,
            unsigned int * pRangeMax,
            unsigned int * pPixelValueMin,
            unsigned int * pPixelValueMax,
            unsigned int * pNumPixelValues,
            float * pPixelValueMean,
            int ** ppHistogram )
```

Retrieves all available information of an image statistics channel.

See also

spinError

Parameters

hStatistics	The image statistics context of the channel
channel	The channel of the information to retrieve
pRangeMin	The unsigned integer pointer in which the minimum value of the range is returned
pRangeMax	The unsigned integer pointer in which the maximum value of the range is returned
pPixelValueMin	The unsigned integer pointer in which the minimum pixel value of the range is returned
pPixelValueMax	The unsigned integer pointer in which the maximum pixel value of the range is returned
pNumPixelValues	The unsigned integer pointer in which the number of pixel values is returned
pPixelValueMean	The float pointer in which the mean pixel value is returned
ppiHistogram	The pointer to the pointer in which the histogram data is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.140 spinImageStatisticsGetChannelStatus()

```
SPINNAKERC_API spinImageStatisticsGetChannelStatus (
            spinImageStatistics hStatistics,
            spinStatisticsChannel channel,
            bool8_t * pbEnabled )
```

Checks whether an image statistics context is enabled.

See also

spinError

Parameters

hStatistics	The image statistics context of the channel
channel	The channel to check
pbEnabled	The boolean pointer to return whether or not the channel is enabled

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.141 spinImageStatisticsGetHistogram()

```
SPINNAKERC_API spinImageStatisticsGetHistogram (
             spinImageStatistics hStatistics,
             spinStatisticsChannel channel,
             int ** ppHistogram )
```

Retrieves a histogram of an image statistics channel.

See also

spinError

Parameters

hStatistics	The image statistics context of the channel
channel	The channel of the histogram to be returned
pHistogram	The pointer to the integer pointer in which the histogram data is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.142 spinImageStatisticsGetMean()

```
SPINNAKERC_API spinImageStatisticsGetMean (
             spinImageStatistics hStatistics,
             spinStatisticsChannel channel,
             float * pMean )
```

Retrieves the mean of pixel values of an image statistics channel.

See also

spinError

Parameters

hStatistics	The image statistics context of the channel
channel	The channel of the mean pixel value to be retrieved
pMean	The float pointer in which the mean pixel value is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.143 spinImageStatisticsGetNumPixelValues()

```
SPINNAKERC_API spinImageStatisticsGetNumPixelValues (
             spinImageStatistics hStatistics,
             spinStatisticsChannel channel,
             unsigned int * pNumValues )
```

Retrieves the number of pixel values of an image statistics channel.

See also

spinError

Parameters

hStatistics	The image statistics context of the channel
channel	The channel where the pixel values to be counted are
iNumValues	The unsigned integer pointer in which the number of pixel values is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.144 spinImageStatisticsGetPixelValueRange()

```
SPINNAKERC_API spinImageStatisticsGetPixelValueRange (
             spinImageStatistics hStatistics,
             spinStatisticsChannel channel,
             unsigned int * pMin,
             unsigned int * pMax )
```

Retrieves the pixel value range of an image statistics channel.

See also

spinError

Parameters

hStatistics	The image statistics context of the channel
channel	The channel of the pixel value range to retrieve
pMin	The unsigned integer pointer in which the minimum value of the pixel value range is returned
pMax	The unsigned integer pointer in which the maximum value of the pixel value range is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.145 spinImageStatisticsGetRange()

```
SPINNAKERC_API spinImageStatisticsGetRange (
            spinImageStatistics hStatistics,
            spinStatisticsChannel channel,
            unsigned int * pMin,
            unsigned int * pMax )
```

Retrieves the range of an image statistics channel.

See also

spinError

Parameters

hStatistics	The image statistics context of the channel
channel	The channel of the range to retrieve
pMin	The unsigned integer pointer in which the minimum value of the range is returned
pMax	The unsigned integer pointer in which the maximum value of the range is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.146 spinImageStatisticsSetChannelStatus()

```
SPINNAKERC_API spinImageStatisticsSetChannelStatus (
            spinImageStatistics hStatistics,
            spinStatisticsChannel channel,
            bool8_t bEnable )
```

Sets the status of an image statistics channel.

See also

spinError

Parameters

hStatistics	The image statistics context of the channel
channel	The channel to enable/disable
bEnable	The boolean value to set; true enables, false disables

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.147 spinInterfaceEventHandlerCreate()

```
SPINNAKERC_API spinInterfaceEventHandlerCreate (
            spinInterfaceEventHandler * phInterfaceEventHandler,
            spinArrivalEventFunction pArrivalFunction,
            spinRemovalEventFunction pRemovalFunction,
            void * pUserData )
```

Creates an interface event handler (both device arrival and device removal)

See also

spinError

Parameters

phInterfaceEventHandler	The interface event handler pointer in which the interface event context is created
pArrivalFunction	The function to be called at arrival event occurrences; signature to match: void(<em>spinArrivalEventFunction)(void pUserData)
hRemovalFunction	The function to be called at removal event occurrences; signature to match: void(<em>spinRemovalEventFunction)(uint64_t deviceSerialNumber, void pUserData)
pUserData	Properties that can be passed into the event function

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.148 spinInterfaceEventHandlerDestroy()

```
SPINNAKERC_API spinInterfaceEventHandlerDestroy (
            spinInterfaceEventHandler hInterfaceEventHandler )
```

Destroys an interface event handler (both device arrival and device removal)

See also

spinError

Parameters

hInterfaceEventHandler	The interface event handler to destroy

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.149 spinInterfaceGetCameras()

```
SPINNAKERC_API spinInterfaceGetCameras (
            spinInterface hInterface,
            spinCameraList hCameraList )
```

Retrieves a camera list from an interface; camera lists must be created and destroy.

See also

spinCameraListCreateEmpty()

spinCameraListDestroy()

spinError

Parameters

hInterface	The interface of the camera list to retrieve
hCameraList	The camera list to house the cameras from the interface

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.150 spinInterfaceGetCamerasEx()

```
SPINNAKERC_API spinInterfaceGetCamerasEx (
            spinInterface hInterface,
            bool8_t bUpdateCameras,
            spinCameraList hCameraList )
```

Retrieves a camera list from an interface; manually set whether to update the cameras; camera lists must be created and destroyed.

See also

spinCameraListCreateEmpty()

spinCameraListDestroy()

spinError

Parameters

hInterface	The interface of the camera list to retrieve
bUpdateCameras	The boolean of whether or not to update the cameras
hCameraList	The camera list to house the cameras from the interface

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.151 spinInterfaceGetTLNodeMap()

```
SPINNAKERC_API spinInterfaceGetTLNodeMap (
            spinInterface hInterface,
            spinNodeMapHandle * phNodeMap )
```

Retrieves the transport layer nodemap from an interface.

See also

spinError

Parameters

hInterface	The interface of the nodemap to retrieve
phNodeMap	The nodemap handle pointer in which the transport layer interface nodemap is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.152 spinInterfaceIsInUse()

```
SPINNAKERC_API spinInterfaceIsInUse (
            spinInterface hInterface,
            bool8_t * pbIsInUse )
```

Checks whether an interface is in use.

See also

spinError

Parameters

hInterface	The interface to check
pbIsInUse	The boolean pointer to return whether or not the interface is in use

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.153 spinInterfaceListClear()

```
SPINNAKERC_API spinInterfaceListClear (
            spinInterfaceList hInterfaceList )
```

Clears an interface list.

See also

spinError

Parameters

hInterfaceList	The interface list to clear

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.154 spinInterfaceListCreateEmpty()

```
SPINNAKERC_API spinInterfaceListCreateEmpty (
            spinInterfaceList * phInterfaceList )
```

Creates an empty interface list (interface lists created this way must be destroyed)

See also

spinError

Parameters

phInterfaceList	The interface list handle pointer in which the empty interface list is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.155 spinInterfaceListDestroy()

```
SPINNAKERC_API spinInterfaceListDestroy (
            spinInterfaceList hInterfaceList )
```

Destroys an interface list.

See also

spinError

Parameters

hInterfaceList	The interface list to destroy

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.156 spinInterfaceListGet()

```
SPINNAKERC_API spinInterfaceListGet (
            spinInterfaceList hInterfaceList,
            size_t index,
            spinInterface * phInterface )
```

Retrieves an interface from an interface list using an index (interfaces retrieved this way must be released)

See also

spinError

Parameters

hInterfaceList	The interface list of the interface to be retrieved
index	The index of the interface
phInterface	The interface handle pointer in which the interface is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.157 spinInterfaceListGetSize()

```
SPINNAKERC_API spinInterfaceListGetSize (
            spinInterfaceList hInterfaceList,
            size_t * pSize )
```

Retrieves the number of interfaces in an interface list.

See also

spinError

Parameters

hInterfaceList	The interface list where the interfaces to be counted are
pSize	The unsigned integer pointer in which the number of interfaces is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

See also

spinError

13.12.1.158 spinInterfaceRegisterDeviceArrivalEventHandler()

```
SPINNAKERC_API spinInterfaceRegisterDeviceArrivalEventHandler (
            spinInterface hInterface,
            spinDeviceArrivalEventHandler hDeviceArrivalEventHandler )
```

Registers a device arrival event handler on an interface (event handlers registered in this way must be unregistered)

See also

spinError

Parameters

hInterface	The interface on which to register the device arrival event handler
hDeviceArrivalEventHandler	The device arrival event handler to register

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.159 spinInterfaceRegisterDeviceRemovalEventHandler()

```
SPINNAKERC_API spinInterfaceRegisterDeviceRemovalEventHandler (
            spinInterface hInterface,
            spinDeviceRemovalEventHandler hDeviceRemovalEventHandler )
```

Registers a device removal event handler on an interface (event handlers registered in this way must be unregistered)

See also

spinError

Parameters

hInterface	the Interface on which to register the device removal event handler
hDeviceRemovalEventHandler	The device removal event handler to register

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.160 spinInterfaceRegisterInterfaceEventHandler()

```
SPINNAKERC_API spinInterfaceRegisterInterfaceEventHandler (
            spinInterface hInterface,
            spinInterfaceEventHandler hInterfaceEventHandler )
```

Registers an interface event handler (both device arrival and device removal) on an interface.

See also

spinError

Parameters

hInterface	The interface on which to register the interface event handler
hInterfaceEventHandler	The interface event handler to register

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.161 spinInterfaceRelease()

```
SPINNAKERC_API spinInterfaceRelease (
            spinInterface hInterface )
```

Releases an interface.

See also

spinError

Parameters

hInterface	The interface to release

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.162 spinInterfaceSendActionCommand()

```
SPINNAKERC_API spinInterfaceSendActionCommand (
            spinInterface hInterface,
            size_t iDeviceKey,
            size_t iGroupKey,
            size_t iGroupMask,
            size_t iActionTime,
            size_t * piResultSize,
            actionCommandResult results[] )
```

Broadcast an Action Command to all devices on interface.

See also

spinError

Parameters

iDeviceKey	The Action Command's device key
iGroupKey	The Action Command's group key
iGroupMask	The Action Command's group mask
iActionTime	(Optional) Time when to assert a future action. Zero means immediate action.

Parameters

piResultSize	(Optional) The number of results in the results array. The value passed should be equal to the expected number of devices that acknowledge the command. Returns the number of received results.
results	(Optional) An Array with ∗piResultSize elements to hold the action command result status. The buffer is filled starting from index 0. If received results are less than expected number of devices that acknowledge the command, remaining results are not changed. If received results are more than expected number of devices that acknowledge the command, extra results are ignored and not appended to array. This parameter is ignored if piResultSize is 0. Thus this parameter can be NULL if pResultSize is 0 or NULL.

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.163 spinInterfaceUnregisterDeviceArrivalEventHandler()

```
SPINNAKERC_API spinInterfaceUnregisterDeviceArrivalEventHandler (
            spinInterface hInterface,
            spinDeviceArrivalEventHandler hDeviceArrivalEventHandler )
```

Unregisters a device arrival event handler from an interface.

See also

spinError

Parameters

hInterface	The interface from which to unregister the device arrival event handler
hDeviceArrivalEventHandler	The device arrival event handler to unregister

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.164 spinInterfaceUnregisterDeviceRemovalEventHandler()

```
SPINNAKERC_API spinInterfaceUnregisterDeviceRemovalEventHandler (
            spinInterface hInterface,
            spinDeviceRemovalEventHandler hDeviceRemovalEventHandler )
```

Unregisters a device removal event handler from an interface.

See also

spinError

Parameters

hInterface	The interface from which to unregister the device removal event handler
hDeviceRemovalEventHandler	The device removal event handler to unregister

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.165 spinInterfaceUnregisterInterfaceEventHandler()

```
SPINNAKERC_API spinInterfaceUnregisterInterfaceEventHandler (
            spinInterface hInterface,
            spinInterfaceEventHandler hInterfaceEventHandler )
```

Unregisters an interface event handler from an interface.

See also

spinError

Parameters

hInterface	The interface from which to unregister the interface event handler
hInterfaceEventHandler	The interface event handler to unregister

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.166 spinInterfaceUpdateCameras()

```
SPINNAKERC_API spinInterfaceUpdateCameras (
            spinInterface hInterface,
            bool8_t * pbChanged )
```

Checks whether any cameras have been connected or disconnected on an interface.

See also

spinError

Parameters

hInterface	The interface of the list of attached cameras to update
pbChanged	The boolean pointer to return whether or not the cameras have changed

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.167 spinLogDataGetCategoryName()

```
SPINNAKERC_API spinLogDataGetCategoryName (
            spinLogEventData hLogEventData,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the category name of a log event.

See also

spinError

Parameters

hLogEventData	The log event data received from the log event
pBuf	The c-string character buffer in which the category name of the log event is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.168 spinLogDataGetLogMessage()

```
SPINNAKERC_API spinLogDataGetLogMessage (
            spinLogEventData hLogEventData,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the log message of a log event.

See also

spinError

Parameters

hLogEventData	The log event data received from the log event
pBuf	The c-string character buffer in which the log message of the log event is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.169 spinLogDataGetNDC()

```
SPINNAKERC_API spinLogDataGetNDC (
            spinLogEventData hLogEventData,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the NDC of a log event.

See also

spinError

Parameters

hLogEventData	The log event data received from the log event
pBuf	The c-string character buffer in which the NDC of the log event is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.170 spinLogDataGetPriority()

```
SPINNAKERC_API spinLogDataGetPriority (
            spinLogEventData hLogEventData,
            int64_t * pValue )
```

Retrieves the priority of a log event.

See also

spinError

Parameters

hLogEventData	The log event data received from the log event
pValue	The integer pointer in which the priority value is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.171 spinLogDataGetPriorityName()

```
SPINNAKERC_API spinLogDataGetPriorityName (
            spinLogEventData hLogEventData,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the priority name of a log event.

See also

spinError

Parameters

hLogEventData	The log event data received from the log event
pBuf	The c-string character buffer in which the priority name of the log event is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.172 spinLogDataGetThreadName()

```
SPINNAKERC_API spinLogDataGetThreadName (
            spinLogEventData hLogEventData,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the thread name of a log event.

See also

spinError

Parameters

hLogEventData	The log event data received from the log event
pBuf	The c-string character buffer in which the thread name of the log event is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.173 spinLogDataGetTimestamp()

```
SPINNAKERC_API spinLogDataGetTimestamp (
             spinLogEventData hLogEventData,
             char * pBuf,
             size_t * pBufLen )
```

Retrieves the timestamp of a log event.

See also

spinError

Parameters

hLogEventData	The log event data received from the log event
pBuf	The c-string character buffer in which the timestamp of the log event is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.174 spinLogEventHandlerCreate()

```
SPINNAKERC_API spinLogEventHandlerCreate (
             spinLogEventHandler * phLogEventHandler,
             spinLogEventFunction pFunction,
             void * pUserData )
```

Creates a log event handler.

See also

spinError

Parameters

phLogEventHandler	The log event handler pointer in which the log event context is created
pFunction	The function to be called at device event occurrences; signature to match: void(<em>spinLogEventFunction)(const spinLogEventData hEventData, void pUserData)
pUserData	Properties that can be passed into the event function

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.175 spinLogEventHandlerDestroy()

```
SPINNAKERC_API spinLogEventHandlerDestroy (
            spinLogEventHandler hLogEventHandler )
```

Destroys a log event handler.

See also

spinError

Parameters

hLogEventHandler	The log event handler to destroy

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.176 spinSystemGetCameras()

```
SPINNAKERC_API spinSystemGetCameras (
            spinSystem hSystem,
            spinCameraList hCameraList )
```

Retrieves a list of detected (and enumerable) cameras on the system; camera lists must be created and destroyed.

See also

spinCameraListCreateEmpty()
spinCameraListDestroy()
spinError

Parameters

hSystem	The system, from which the camera list is retrieved
hCameraList	The camera list to house the cameras from the system

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.177 spinSystemGetCamerasEx()

```
SPINNAKERC_API spinSystemGetCamerasEx (
            spinSystem hSystem,
            bool8_t bUpdateInterfaces,
            bool8_t bUpdateCameras,
            spinCameraList hCameraList )
```

Retrieves a list of detected (and enumerable) cameras on the system; manually set whether to update the current interface and camera lists; camera lists must be created and destroyed.

See also

spinCameraListCreateEmpty()

spinCameraListDestroy()

spinError

Parameters

hSystem	The system, from which the camera list is retrieved
bUpdateInterfaces	The boolean of whether to update the interface list
bUpdateCameras	The boolean of whether to update the camera list
hCameraList	The camera list to house the cameras from the system

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.178 spinSystemGetInstance()

```
SPINNAKERC_API spinSystemGetInstance (
            spinSystem * phSystem )
```

Retrieves an instance of the system object; the system is a singleton, so there will only ever be one instance; system instance must be destroyed by calling spinSystemReleaseInstance.

See also

spinSystemReleaseInstance

spinError

Parameters

phSystem	The system handle pointer in which the system instance is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.179 spinSystemGetInterfaces()

```
SPINNAKERC_API spinSystemGetInterfaces (
            spinSystem hSystem,
            spinInterfaceList hInterfaceList )
```

Retrieves a list of detected (and enumerable) interfaces on the system; interface lists must be created and destroyed.

See also

spinInterfaceListCreateEmpty()

spinInterfaceListDestroy()

spinError

Parameters

hSystem	The system, from which the interface list is retrieved
hInterfaceList	The interface list to house the interfaces from the system

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.180 spinSystemGetLibraryVersion()

```
SPINNAKERC_API spinSystemGetLibraryVersion (
            spinSystem hSystem,
            spinLibraryVersion * hLibraryVersion )
```

Get current library version of Spinnaker.

Returns

A struct containing the current version of Spinnaker(major, minor, type, build).

13.12.1.181 spinSystemGetLoggingLevel()

```
SPINNAKERC_API spinSystemGetLoggingLevel (
            spinSystem hSystem,
            spinnakerLogLevel * pLogLevel )
```

Retrieves the logging level for all logging events on the system.

See also

spinError

Parameters

hSystem	The system, from which the logging level is retrieved
logLevel	The logging level enum pointer in which the current logging level is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.182 spinSystemGetTLNodeMap()

```
SPINNAKERC_API spinSystemGetTLNodeMap (
            spinSystem hSystem,
            spinNodeMapHandle * phNodeMap )
```

Retrieves the transport layer nodemap from the system.

See also

spinError

Parameters

hSystem	The system handle.
phNodeMap	The nodemap handle pointer in which the transport layer system nodemap is returned.

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.183 spinSystemIsInUse()

```
SPINNAKERC_API spinSystemIsInUse (
            spinSystem hSystem,
            bool8_t * pbIsInUse )
```

Checks whether a system is currently in use.

See also

spinError

Parameters

hSystem	The system to check
pbIsInUse	The boolean pointer to return whether the system is currently in use

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.184 spinSystemRegisterDeviceArrivalEventHandler()

```
SPINNAKERC_API spinSystemRegisterDeviceArrivalEventHandler (
            spinSystem hSystem,
            spinDeviceArrivalEventHandler hDeviceArrivalEventHandler )
```

Registers a device arrival event handler to every interface on the system (event handlers registered this way must be unregistered)

See also

spinError

Parameters

hSystem	The system, on which the device arrival event handler is registered
hDeviceArrivalEventHandler	The device arrival event handler to register on the system

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.185 spinSystemRegisterDeviceRemovalEventHandler()

```
SPINNAKERC_API spinSystemRegisterDeviceRemovalEventHandler (
            spinSystem hSystem,
            spinDeviceRemovalEventHandler hDeviceRemovalEventHandler )
```

Registers a device removal event handler to the system to every interface on the system (event handlers registered this way must be unregistered)

See also

spinError

Parameters

hSystem	The system, on which the device removal event handler is registered
hDeviceRemovalEventHandler	The device removal event handler to register on the system

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.186 spinSystemRegisterInterfaceEventHandler()

```
SPINNAKERC_API spinSystemRegisterInterfaceEventHandler (
            spinSystem hSystem,
            spinInterfaceEventHandler hInterfaceEventHandler )
```

Registers an interface event handler (device arrival and device removal) to every interface on the system (interface events registered this way must be unregistered) If new interfaces are detected by the system after spinSystemRegisterInterfaceEventHandler() is called, those interfaces will be automatically registered with this event.

See also

spinError

spinInterfaceEventHandler

Parameters

hSystem	The system, on which the interface event handler is registered
hInterfaceEventHandler	The interface event handler (device arrival and device removal) to register on the system

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.187 spinSystemRegisterLogEventHandler()

```
SPINNAKERC_API spinSystemRegisterLogEventHandler (
            spinSystem hSystem,
            spinLogEventHandler hLogEventHandler )
```

Registers a logging event handler to the system (event handlers registered in this way must be unregistered)

See also

spinError

Parameters

hSystem	The system, on which the logging event handler is registered
hLogEventHandler	The logging event handler to register on the system

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.188 spinSystemReleaseInstance()

```
SPINNAKERC_API spinSystemReleaseInstance (
            spinSystem hSystem )
```

Releases the system; make sure handle is cleaned up properly by setting it to NULL after system is released; the handle can only be used again after calling spinSystemGetInstance.

See also

spinSystemGetInstance

spinError

Parameters

hSystem	The system handle

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.189 spinSystemSendActionCommand()

```
SPINNAKERC_API spinSystemSendActionCommand (
            spinSystem hSystem,
            size_t iDeviceKey,
            size_t iGroupKey,
            size_t iGroupMask,
            size_t iActionTime,
            size_t * piResultSize,
            actionCommandResult results[] )
```

Broadcast an Action Command to all devices on system.

See also

spinError

Parameters

hSystem	The system on which to send the action command to all devices.
iDeviceKey	The Action Command's device key
iGroupKey	The Action Command's group key
iGroupMask	The Action Command's group mask
iActionTime	(Optional) Time when to assert a future action. Zero means immediate action.
piResultSize	(Optional) The number of results in the results array. The value passed should be equal to the expected number of devices that acknowledge the command. Returns the number of received results.
results	(Optional) An Array with ∗piResultSize elements to hold the action command result status. The buffer is filled starting from index 0. If received results are less than expected number of devices that acknowledge the command, remaining results are not changed. If received results are more than expected number of devices that acknowledge the command, extra results are ignored and not appended to array. This parameter is ignored if piResultSize is 0. Thus this parameter can be NULL if pResultSize is 0 or NULL.

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.190 spinSystemSetLoggingLevel()

```
SPINNAKERC_API spinSystemSetLoggingLevel (
            spinSystem hSystem,
            spinnakerLogLevel logLevel )
```

Sets the logging level for all logging events on the system.

See also

spinError

Parameters

hSystem	The system, on which the logging level is set
logLevel	The logging level to set

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.191 spinSystemUnregisterAllLogEventHandlers()

```
SPINNAKERC_API spinSystemUnregisterAllLogEventHandlers (
            spinSystem hSystem )
```

Unregisters all logging event handlers from the system.

See also

spinError

Parameters

hSystem	The system, from which all logging event handlers are unregistered

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.192 spinSystemUnregisterDeviceArrivalEventHandler()

```
SPINNAKERC_API spinSystemUnregisterDeviceArrivalEventHandler (
            spinSystem hSystem,
            spinDeviceArrivalEventHandler hDeviceArrivalEventHandler )
```

Unregisters a device arrival event handler from the system.

See also

spinError

spinDeviceArrivalEventHandler

Parameters

hSystem	The system, from which the device arrival event handler is unregistered
hDeviceArrivalEventHandler	The device arrival event handler to unregister from the system

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.193 spinSystemUnregisterDeviceRemovalEventHandler()

```
SPINNAKERC_API spinSystemUnregisterDeviceRemovalEventHandler (
            spinSystem hSystem,
            spinDeviceRemovalEventHandler hDeviceRemovalEventHandler )
```

Unregisters a device removal event handler from the system.

See also

spinError

spinDeviceRemovalEventHandler

Parameters

hSystem	The system, from which the device removal event handler is unregistered
hDeviceRemovalEventHandler	The device removal event handler to unregister from the system

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.194 spinSystemUnregisterInterfaceEventHandler()

```
SPINNAKERC_API spinSystemUnregisterInterfaceEventHandler (
            spinSystem hSystem,
            spinInterfaceEventHandler hInterfaceEventHandler )
```

Unregisters an interface event handler from the system.

See also

spinError

spinInterfaceEventHandler

Parameters

hSystem	The system, from which the interface event handler is unregistered
hInterfaceEventHandler	The interface event handler (device arrival and device removal) to unregister from the system

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.195 spinSystemUnregisterLogEventHandler()

```
SPINNAKERC_API spinSystemUnregisterLogEventHandler (
            spinSystem hSystem,
            spinLogEventHandler hLogEventHandler )
```

Unregisters a selected logging event handler from the system.

See also

spinError

Parameters

hSystem	The system, from which the logging event handler is unregistered
hLogEventHandler	The logging event handler to unregister from the system

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.196 spinSystemUpdateCameras()

```
SPINNAKERC_API spinSystemUpdateCameras (
            spinSystem hSystem,
            bool8_t * pbChanged )
```

Updates the list of cameras on the system, informing whether there has been any changes.

See also

spinError

Parameters

hSystem	The system, on which the list of attached cameras is updated
pbChanged	The boolean pointer to return whether cameras have arrived on or been removed from the system

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.12.1.197 spinSystemUpdateCamerasEx()

```
SPINNAKERC_API spinSystemUpdateCamerasEx (
            spinSystem hSystem,
            bool8_t bUpdateInterfaces,
            bool8_t * pbChanged )
```

Updates the list of cameras on the system, informing whether there has been any changes; manually set whether to update the current interface lists.

See also

spinError

Parameters

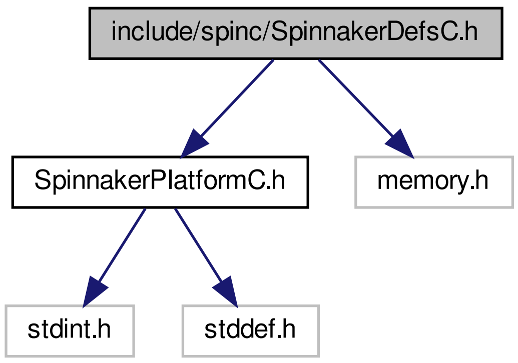
hSystem	The system, on which the list of attached cameras is updated
bUpdateInterfaces	The boolean of whether to update the interface list
pbChanged	The boolean pointer to return whether cameras have arrived or been removed from the system

Returns

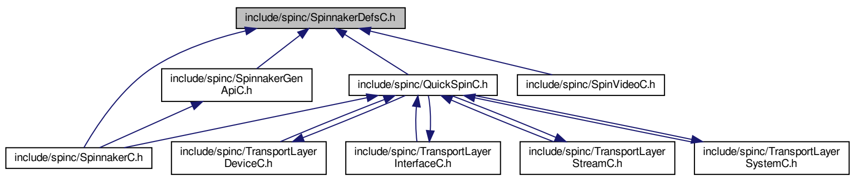
spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.13 include/spinc/SpinnakerDefsC.h File Reference

Include dependency graph for SpinnakerDefsC.h:



This graph shows which files directly or indirectly include this file:


include/spinc/SpinnakerDefsC.h
include/spinc/SpinnakerGen ApiC.h
include/spinc/QuickSpinC.h
include/spinc/SpinVideoC.h
include/spinc/SpinnakerC.h
include/spinc/TransportLayer DeviceC.h
include/spinc/TransportLayer InterfaceC.h
include/spinc/TransportLayer StreamC.h
include/spinc/TransportLayer SystemC.h

Data Structures

- struct spinPNGOption

  Options for saving PNG images.
- struct spinPPMOption

  Options for saving PPM images.
- struct spinPGMOption

  Options for saving PGM images.
- struct spinTIFFOption

  Options for saving TIFF images.
- struct spinJPEGOption

  Options for saving JPEG images.
- struct spinJPG2Option

  Options for saving JPEG 2000 images.
- struct spinBMPOption

  Options for saving BMP images.
- struct spinMJPGOption

  Options for saving MJPG videos.
- struct spinH264Option

  Options for saving H264 videos.
- struct spinAVIOption

  Options for saving uncompressed videos.
- struct spinLibraryVersion

  Provides easier access to the current version of Spinnaker.
- struct actionCommandResult

  Action Command Result.

Typedefs

- typedef uint8_t bool8_t
- typedef void $*$ spinSystem

  Handle for system functionality.
- typedef void $*$ spinInterfaceList

  Handle for interface list functionality.
- typedef void $*$ spinInterface

  Handle for interface functionality.
- typedef void $*$ spinCameraList

  Handle for interface functionality.
- typedef void $*$ spinCamera

Handle for camera functionality.
- typedef void ∗ spinImage

  Handle for image functionality.
- typedef void ∗ spinImageList

  Handle for image list functionality.
- typedef void ∗ spinImageProcessor

  Handle for image processor functionality.
- typedef void ∗ spinImageStatistics

  Handle for image statistics functionality.
- typedef void ∗ spinDeviceEventHandler

  Handle for device event handler functionality.
- typedef void ∗ spinImageEventHandler

  Handle for image event handler functionality.
- typedef void ∗ spinImageListEventHandler

  Handle for image list event handler functionality.
- typedef void ∗ spinDeviceArrivalEventHandler

  Handle for arrival event handler functionality.
- typedef void ∗ spinDeviceRemovalEventHandler

  Handle for removal event handler functionality.
- typedef void ∗ spinInterfaceEventHandler

  Handle for interface event handler functionality.
- typedef void ∗ spinLogEventHandler

  Handle for logging event handler functionality.
- typedef void ∗ spinLogEventData

  Handle for logging event data functionality.
- typedef void ∗ spinDeviceEventData

  Handle for device event data functionality.
- typedef void ∗ spinVideo

  Handle for video recording functionality.
- typedef void(∗ spinDeviceEventFunction) (const spinDeviceEventData hEventData, const char ∗pEvent←Name, void ∗pUserData)

  Function signatures are used to create and trigger callbacks and events.
- typedef void(∗ spinImageEventFunction) (const spinImage hImage, void ∗pUserData)
- typedef void(∗ spinImageListEventFunction) (const spinImageList hImage, void ∗pUserData)
- typedef void(∗ spinArrivalEventFunction) (const spinCamera hCamera, void ∗pUserData)
- typedef void(∗ spinRemovalEventFunction) (const spinCamera hCamera, void ∗pUserData)
- typedef void(∗ spinLogEventFunction) (const spinLogEventData hEventData, void ∗pUserData)

Enumerations

- enum spinError {
  SPINNAKER_ERR_SUCCESS = 0 ,
  SPINNAKER_ERR_ERROR = -1001 ,
  SPINNAKER_ERR_NOT_INITIALIZED = -1002 ,
  SPINNAKER_ERR_NOT_IMPLEMENTED = -1003 ,
  SPINNAKER_ERR_RESOURCE_IN_USE = -1004 ,
  SPINNAKER_ERR_ACCESS_DENIED = -1005 ,
  SPINNAKER_ERR_INVALID_HANDLE = -1006 ,
  SPINNAKER_ERR_INVALID_ID = -1007 ,
  SPINNAKER_ERR_NO_DATA = -1008 ,
  SPINNAKER_ERR_INVALID_PARAMETER = -1009 ,
  SPINNAKER_ERR_IO = -1010 ,

SPINNAKER_ERR_TIMEOUT = -1011 ,
SPINNAKER_ERR_ABORT = -1012 ,
SPINNAKER_ERR_INVALID_BUFFER = -1013 ,
SPINNAKER_ERR_NOT_AVAILABLE = -1014 ,
SPINNAKER_ERR_INVALID_ADDRESS = -1015 ,
SPINNAKER_ERR_BUFFER_TOO_SMALL = -1016 ,
SPINNAKER_ERR_INVALID_INDEX = -1017 ,
SPINNAKER_ERR_PARSING_CHUNK_DATA = -1018 ,
SPINNAKER_ERR_INVALID_VALUE = -1019 ,
SPINNAKER_ERR_RESOURCE_EXHAUSTED = -1020 ,
SPINNAKER_ERR_OUT_OF_MEMORY = -1021 ,
SPINNAKER_ERR_BUSY = -1022 ,
SPINNAKER_ERR_GENICAM_INVALID_ARGUMENT = -2001 ,
SPINNAKER_ERR_GENICAM_OUT_OF_RANGE = -2002 ,
SPINNAKER_ERR_GENICAM_PROPERTY = -2003 ,
SPINNAKER_ERR_GENICAM_RUN_TIME = -2004 ,
SPINNAKER_ERR_GENICAM_LOGICAL = -2005 ,
SPINNAKER_ERR_GENICAM_ACCESS = -2006 ,
SPINNAKER_ERR_GENICAM_TIMEOUT = -2007 ,
SPINNAKER_ERR_GENICAM_DYNAMIC_CAST = -2008 ,
SPINNAKER_ERR_GENICAM_GENERIC = -2009 ,
SPINNAKER_ERR_GENICAM_BAD_ALLOCATION = -2010 ,
SPINNAKER_ERR_IM_CONVERT = -3001 ,
SPINNAKER_ERR_IM_COPY = -3002 ,
SPINNAKER_ERR_IM_MALLOC = -3003 ,
SPINNAKER_ERR_IM_NOT_SUPPORTED = -3004 ,
SPINNAKER_ERR_IM_HISTOGRAM_RANGE = -3005 ,
SPINNAKER_ERR_IM_HISTOGRAM_MEAN = -3006 ,
SPINNAKER_ERR_IM_MIN_MAX = -3007 ,
SPINNAKER_ERR_IM_COLOR_CONVERSION = -3008 ,
SPINNAKER_ERR_CUSTOM_ID = -10000 }

The error codes used in Spinnaker C.

- enum spinColorProcessingAlgorithm {
SPINNAKER_COLOR_PROCESSING_ALGORITHM_NONE ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_NEAREST_NEIGHBOR ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_NEAREST_NEIGHBOR_AVG ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_BILINEAR ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_EDGE_SENSING ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_HQ_LINEAR ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_IPP ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_DIRECTIONAL_FILTER ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_RIGOROUS ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_WEIGHTED_DIRECTIONAL_FILTER }

Color processing algorithms.

- enum spinStatisticsChannel {
SPINNAKER_STATISTICS_CHANNEL_GREY ,
SPINNAKER_STATISTICS_CHANNEL_RED ,
SPINNAKER_STATISTICS_CHANNEL_GREEN ,
SPINNAKER_STATISTICS_CHANNEL_BLUE ,
SPINNAKER_STATISTICS_CHANNEL_HUE ,
SPINNAKER_STATISTICS_CHANNEL_SATURATION ,
SPINNAKER_STATISTICS_CHANNEL_LIGHTNESS ,
SPINNAKER_STATISTICS_CHANNEL_NUM_CHANNELS }

Channels that allow statistics to be calculated.

- enum spinImageFileFormat {
SPINNAKER_IMAGE_FILE_FORMAT_FROM_FILE_EXT = -1 ,
SPINNAKER_IMAGE_FILE_FORMAT_PGM ,
SPINNAKER_IMAGE_FILE_FORMAT_PPM ,

SPINNAKER_IMAGE_FILE_FORMAT_BMP ,
SPINNAKER_IMAGE_FILE_FORMAT_JPEG ,
SPINNAKER_IMAGE_FILE_FORMAT_JPEG2000 ,
SPINNAKER_IMAGE_FILE_FORMAT_TIFF ,
SPINNAKER_IMAGE_FILE_FORMAT_PNG ,
SPINNAKER_IMAGE_FILE_FORMAT_RAW ,
SPINNAKER_IMAGE_FILE_FORMAT_FORCE_32BITS = 0x7FFFFFFF }

File formats to be used for saving images to disk.

- enum spinTLPixelFormatNamespace {
SPINNAKER_TLPIXELFORMAT_NAMESPACE_UNKNOWN = 0 ,
SPINNAKER_TLPIXELFORMAT_NAMESPACE_GEV = 1 ,
SPINNAKER_TLPIXELFORMAT_NAMESPACE_IIDC = 2 ,
SPINNAKER_TLPIXELFORMAT_NAMESPACE_PFNC_16BIT = 3 ,
SPINNAKER_TLPIXELFORMAT_NAMESPACE_PFNC_32BIT = 4 ,
SPINNAKER_PIXELFORMAT_NAMESPACE_CUSTOM_ID = 1000 }

This enum represents the namespace in which the TL specific pixel format resides.

- enum spinImageStatus {
SPINNAKER_IMAGE_STATUS_UNKNOWN_ERROR = -1 ,
SPINNAKER_IMAGE_STATUS_NO_ERROR = 0 ,
SPINNAKER_IMAGE_STATUS_CRC_CHECK_FAILED = 1 ,
SPINNAKER_IMAGE_STATUS_DATA_OVERFLOW = 2 ,
SPINNAKER_IMAGE_STATUS_MISSING_PACKETS ,
SPINNAKER_IMAGE_STATUS_LEADER_BUFFER_SIZE_INCONSISTENT ,
SPINNAKER_IMAGE_STATUS_TRAILER_BUFFER_SIZE_INCONSISTENT ,
SPINNAKER_IMAGE_STATUS_PACKETID_INCONSISTENT ,
SPINNAKER_IMAGE_STATUS_MISSING_LEADER = 7 ,
SPINNAKER_IMAGE_STATUS_MISSING_TRAILER = 8 ,
SPINNAKER_IMAGE_STATUS_DATA_INCOMPLETE = 9 ,
SPINNAKER_IMAGE_STATUS_INFO_INCONSISTENT = 10 ,
SPINNAKER_IMAGE_STATUS_CHUNK_DATA_INVALID = 11 ,
SPINNAKER_IMAGE_STATUS_NO_SYSTEM_RESOURCES = 12 }

Status of images returned from spinImageGetStatus() call.

- enum spinnakerLogLevel {
SPINNAKER_LOG_LEVEL_OFF = -1 ,
SPINNAKER_LOG_LEVEL_FATAL = 0 ,
SPINNAKER_LOG_LEVEL_ALERT = 100 ,
SPINNAKER_LOG_LEVEL_CRIT = 200 ,
SPINNAKER_LOG_LEVEL_ERROR = 300 ,
SPINNAKER_LOG_LEVEL_WARN = 400 ,
SPINNAKER_LOG_LEVEL_NOTICE = 500 ,
SPINNAKER_LOG_LEVEL_INFO = 600 ,
SPINNAKER_LOG_LEVEL_DEBUG = 700 ,
SPINNAKER_LOG_LEVEL_NOTSET = 800 }

log levels

- enum spinTLPayloadType {
SPINNAKER_TLPAYLOAD_TYPE_UNKNOWN = 0 ,
SPINNAKER_TLPAYLOAD_TYPE_IMAGE = 1 ,
SPINNAKER_TLPAYLOAD_TYPE_RAW_DATA = 2 ,
SPINNAKER_TLPAYLOAD_TYPE_FILE = 3 ,
SPINNAKER_TLPAYLOAD_TYPE_CHUNK_DATA = 4 ,
SPINNAKER_TLPAYLOAD_TYPE_JPEG = 5 ,
SPINNAKER_TLPAYLOAD_TYPE_JPEG2000 = 6 ,
SPINNAKER_TLPAYLOAD_TYPE_H264 = 7 ,
SPINNAKER_TLPAYLOAD_TYPE_CHUNK_ONLY = 8 ,
SPINNAKER_TLPAYLOAD_TYPE_DEVICE_SPECIFIC = 9 ,
SPINNAKER_TLPAYLOAD_TYPE_MULTI_PART = 10 ,
SPINNAKER_TLPAYLOAD_TYPE_CUSTOM_ID = 1000 ,

SPINNAKER_TLPAYLOAD_TYPE_LOSSLESS_COMPRESSED = SPINNAKER_TLPAYLOAD_TYPE_↩
CUSTOM_ID + 1 ,
SPINNAKER_TLPAYLOAD_TYPE_LOSSY_COMPRESSED = SPINNAKER_TLPAYLOAD_TYPE_↩
CUSTOM_ID + 2 ,
SPINNAKER_TLPAYLOAD_TYPE_JPEG_LOSSLESS_COMPRESSED = SPINNAKER_TLPAYLOAD_↩
TYPE_CUSTOM_ID + 3 }
- enum spinTIFFCompressionMethod {
SPINNAKER_TIFF_COMPRESS_METHOD_NONE = 1 ,
SPINNAKER_TIFF_COMPRESS_METHOD_PACKBITS ,
SPINNAKER_TIFF_COMPRESS_METHOD_DEFLATE ,
SPINNAKER_TIFF_COMPRESS_METHOD_ADOBE_DEFLATE ,
SPINNAKER_TIFF_COMPRESS_METHOD_CCITTFAX3 ,
SPINNAKER_TIFF_COMPRESS_METHOD_CCITTFAX4 ,
SPINNAKER_TIFF_COMPRESS_METHOD_LZW ,
SPINNAKER_TIFF_COMPRESS_METHOD_JPG }

  Compression method to use for encoding TIFF images.
- enum spinActionCommandStatus {
SPINNAKER_ACTION_COMMAND_STATUS_OK = 0 ,
SPINNAKER_ACTION_COMMAND_STATUS_NO_REF_TIME = 0x8013 ,
SPINNAKER_ACTION_COMMAND_STATUS_OVERFLOW = 0x8015 ,
SPINNAKER_ACTION_COMMAND_STATUS_ACTION_LATE = 0x8016 ,
SPINNAKER_ACTION_COMMAND_STATUS_ERROR = 0x8FFF }

  Possible Status Codes Returned from Action Command.

Variables

- static const bool8_t False = 0
- static const bool8_t True = 1

13.13.1 Typedef Documentation

13.13.1.1 bool8_t

```
typedef uint8_t bool8_t
```

13.13.1.2 spinArrivalEventFunction

```
typedef void(* spinArrivalEventFunction) (const spinCamera hCamera, void *pUserData)
```

13.13.1.3 spinCamera

```
typedef void* spinCamera
```

Handle for camera functionality.

Created by calling spinCameraListGet(), which requires a call to spinCameraRelease() to release.

13.13.1.4 spinCameraList

```
typedef void* spinCameraList
```

Handle for interface functionality.

Created by calling spinSystemGetCameras() or spinInterfaceGetCameras(), which require a call to spinCameraListClear() to clear, or spinCameraListCreateEmpty(), which requires a call to spinCameraListDestroy() to destroy.

13.13.1.5 spinDeviceArrivalEventHandler

```
typedef void* spinDeviceArrivalEventHandler
```

Handle for arrival event handler functionality.

Created by calling spinArrivalEventCreate(), which requires a call to spinDeviceArrivalEventHandlerDestroy() to destroy.

13.13.1.6 spinDeviceEventData

```
typedef void* spinDeviceEventData
```

Handle for device event data functionality.

Received in device event function. No need to release, clear, or destroy.

13.13.1.7 spinDeviceEventFunction

```
typedef void(* spinDeviceEventFunction) (const spinDeviceEventData hEventData, const char *p↩
EventName, void *pUserData)
```

Function signatures are used to create and trigger callbacks and events.

13.13.1.8 spinDeviceEventHandler

```
typedef void* spinDeviceEventHandler
```

Handle for device event handler functionality.

Created by calling spinDeviceEventHandlerCreate(), which requires a call to spinDeviceEventHandlerDestroy() to destroy.

13.13.1.9 spinDeviceRemovalEventHandler

```
typedef void* spinDeviceRemovalEventHandler
```

Handle for removal event handler functionality.

Created by calling spinDeviceRemovalEventHandlerCreate(), which requires a call to spinDeviceRemovalEventHandlerDestroy() to destroy.

13.13.1.10 spinImage

```
typedef void* spinImage
```

Handle for image functionality.

Created by calling spinCameraGetNextImage() or spinCameraGetNextImageEx(), which require a call to spinImageRelease() to remove from buffer, or spinImageCreateEmpty(), spinImageCreateEx(), or spinImageCreate(), which require a call to spinImageDestroy() to destroy.

13.13.1.11 spinImageEventFunction

```
typedef void(* spinImageEventFunction) (const spinImage hImage, void *pUserData)
```

13.13.1.12 spinImageEventHandler

```
typedef void* spinImageEventHandler
```

Handle for image event handler functionality.

Created by calling spinImageEventHandlerCreate(), which requires a call to spinImageEventHandlerDestroy() to destroy.

13.13.1.13 spinImageList

```
typedef void* spinImageList
```

Handle for image list functionality.

Created by calling spinCameraGetNextImageSync(), which require a call to spinImageRelease() to remove from buffer, or spinImageCreateEmpty(), spinImageCreateEx(), or spinImageCreate(), which require a call to spinImageDestroy() to destroy.

13.13.1.14 spinImageListEventFunction

```
typedef void(* spinImageListEventFunction) (const spinImageList hImage, void *pUserData)
```

13.13.1.15 spinImageListEventHandler

```
typedef void* spinImageListEventHandler
```

Handle for image list event handler functionality.

Created by calling spinImageListEventHandlerCreate(), which requires a call to spinImageListEventHandlerDestroy() to destroy.

13.13.1.16 spinImageProcessor

```
typedef void* spinImageProcessor
```

Handle for image processor functionality.

Created by calling spinImageProcessorCreate(), which requires a call to spinImageProcessorDestroy() to destroy.

13.13.1.17 spinImageStatistics

```
typedef void* spinImageStatistics
```

Handle for image statistics functionality.

Created by calling spinImageStatisticsCreate(), which requires a call to spinImageStatisticsDestroy() to destroy.

13.13.1.18 spinInterface

```
typedef void* spinInterface
```

Handle for interface functionality.

Created by calling spinInterfaceListGet(), which requires a call to spinInterfaceRelease() to release.

13.13.1.19 spinInterfaceEventHandler

```
typedef void* spinInterfaceEventHandler
```

Handle for interface event handler functionality.

Created by calling spinInterfaceEventHandlerCreate(), which requires a call to spinInterfaceEventHandlerDestroy() to destroy.

13.13.1.20 spinInterfaceList

```
typedef void* spinInterfaceList
```

Handle for interface list functionality.

Created by calling spinSystemGetInterfaces(), which requires a call to spinInterfaceListClear() to clear, or spinInterfaceListCreateEmpty(), which requires a call to spinInterfaceListDestroy() to destroy.

13.13.1.21 spinLogEventData

```
typedef void* spinLogEventData
```

Handle for logging event data functionality.

Received in log event function. No need to release, clear, or destroy.

13.13.1.22 spinLogEventFunction

```
typedef void(* spinLogEventFunction) (const spinLogEventData hEventData, void *pUserData)
```

13.13.1.23 spinLogEventHandler

```
typedef void* spinLogEventHandler
```

Handle for logging event handler functionality.

Created by calling spinLogEventHandlerCreate(), which requires a call to spinLogEventHandlerDestroy() to destroy.

13.13.1.24 spinRemovalEventFunction

```
typedef void(* spinRemovalEventFunction) (const spinCamera hCamera, void *pUserData)
```

13.13.1.25 spinSystem

```
typedef void* spinSystem
```

Handle for system functionality.

Created by calling spinSystemGetInstance(), which requires a call to spinSystemReleaseInstance() to release.

13.13.1.26 spinVideo

```
typedef void* spinVideo
```

Handle for video recording functionality.

Created by calling spinVideoOpenUncompressed(), spinVideoOpenMJPG(), and spinVideoOpenH264(), which require a call to spinVideoClose() to destroy.

13.13.2 Enumeration Type Documentation

13.13.2.1 spinActionCommandStatus

```
enum spinActionCommandStatus
```

Possible Status Codes Returned from Action Command.

Enumerator

SPINNAKER_ACTION_COMMAND_STATUS_OK	The device acknowledged the command.
SPINNAKER_ACTION_COMMAND_STATUS_NO_REF_TIME	
SPINNAKER_ACTION_COMMAND_STATUS_OVERFLOW	
SPINNAKER_ACTION_COMMAND_STATUS_ACTION_LATE	
SPINNAKER_ACTION_COMMAND_STATUS_ERROR	

13.13.2.2 spinColorProcessingAlgorithm

```
enum spinColorProcessingAlgorithm
```

Color processing algorithms.

Please refer to our knowledge base at article at https://www.flir.com/support-center/iis/machine-vision/k
for complete details for each algorithm.

Enumerator

SPINNAKER_COLOR_PROCESSING_↩ ALGORITHM_NONE	No color processing.
SPINNAKER_COLOR_PROCESSING_↩ ALGORITHM_NEAREST_NEIGHBOR	Fastest but lowest quality. Equivalent to FLYCAPTURE_NEAREST_NEIGHBOR_FAST in FlyCapture.
SPINNAKER_COLOR_PROCESSING_↩ ALGORITHM_NEAREST_NEIGHBOR_AVG	Nearest Neighbor with averaged green pixels. Higher quality but slower compared to nearest neighbor without averaging.
SPINNAKER_COLOR_PROCESSING_↩ ALGORITHM_BILINEAR	Weighted average of surrounding 4 pixels in a 2x2 neighborhood.
SPINNAKER_COLOR_PROCESSING_↩ ALGORITHM_EDGE_SENSING	Weights surrounding pixels based on localized edge orientation.
SPINNAKER_COLOR_PROCESSING_↩ ALGORITHM_HQ_LINEAR	Well-balanced speed and quality.
SPINNAKER_COLOR_PROCESSING_↩ ALGORITHM_IPP	Multi-threaded with similar results to edge sensing.
SPINNAKER_COLOR_PROCESSING_↩ ALGORITHM_DIRECTIONAL_FILTER	Best quality but much faster than rigorous.
SPINNAKER_COLOR_PROCESSING_↩ ALGORITHM_RIGOROUS	Slowest but produces good results.
SPINNAKER_COLOR_PROCESSING_↩ ALGORITHM_WEIGHTED_DIRECTIONAL_FILTER	Weighted pixel average from different directions.

13.13.2.3 spinError

```
enum spinError
```

The error codes used in Spinnaker C.

These codes are returned from every function in Spinnaker C. The error codes in the range of -2000 to -2999 are reserved for GenICam related errors. The error codes in the range of -3000 to -3999 are reserved for image processing related errors.

Enumerator

SPINNAKER_ERR_SUCCESS	An error code of 0 means that the function has run without error.
SPINNAKER_ERR_ERROR	The error codes in the range of -1000 to -1999 are reserved for Spinnaker exceptions.
SPINNAKER_ERR_NOT_INITIALIZED	
SPINNAKER_ERR_NOT_IMPLEMENTED	
SPINNAKER_ERR_RESOURCE_IN_USE	
SPINNAKER_ERR_ACCESS_DENIED	
SPINNAKER_ERR_INVALID_HANDLE	
SPINNAKER_ERR_INVALID_ID	
SPINNAKER_ERR_NO_DATA	
SPINNAKER_ERR_INVALID_PARAMETER	
SPINNAKER_ERR_IO	
SPINNAKER_ERR_TIMEOUT	
SPINNAKER_ERR_ABORT	
SPINNAKER_ERR_INVALID_BUFFER	
SPINNAKER_ERR_NOT_AVAILABLE	
SPINNAKER_ERR_INVALID_ADDRESS	
SPINNAKER_ERR_BUFFER_TOO_SMALL	
SPINNAKER_ERR_INVALID_INDEX	
SPINNAKER_ERR_PARSING_CHUNK_DATA	
SPINNAKER_ERR_INVALID_VALUE	
SPINNAKER_ERR_RESOURCE_EXHAUSTED	
SPINNAKER_ERR_OUT_OF_MEMORY	
SPINNAKER_ERR_BUSY	
SPINNAKER_ERR_GENICAM_INVALID_↩ ARGUMENT	The error codes in the range of -2000 to -2999 are reserved for Gen API related errors.
SPINNAKER_ERR_GENICAM_OUT_OF_RANGE	
SPINNAKER_ERR_GENICAM_PROPERTY	
SPINNAKER_ERR_GENICAM_RUN_TIME	
SPINNAKER_ERR_GENICAM_LOGICAL	
SPINNAKER_ERR_GENICAM_ACCESS	
SPINNAKER_ERR_GENICAM_TIMEOUT	
SPINNAKER_ERR_GENICAM_DYNAMIC_CAST	
SPINNAKER_ERR_GENICAM_GENERIC	
SPINNAKER_ERR_GENICAM_BAD_ALLOCATION	
SPINNAKER_ERR_IM_CONVERT	The error codes in the range of -3000 to -3999 are reserved for image processing related errors.
SPINNAKER_ERR_IM_COPY	
SPINNAKER_ERR_IM_MALLOC	
SPINNAKER_ERR_IM_NOT_SUPPORTED	
SPINNAKER_ERR_IM_HISTOGRAM_RANGE	
SPINNAKER_ERR_IM_HISTOGRAM_MEAN	
SPINNAKER_ERR_IM_MIN_MAX	
SPINNAKER_ERR_IM_COLOR_CONVERSION	

Enumerator

SPINNAKER_ERR_CUSTOM_ID	Error codes less than -10000 are reserved for user-defined custom errors.

13.13.2.4 spinImageFileFormat

```
enum spinImageFileFormat
```

File formats to be used for saving images to disk.

Enumerator

SPINNAKER_IMAGE_FILE_FORMAT_FROM_FILE_EXT	Determine file format from file extension.
SPINNAKER_IMAGE_FILE_FORMAT_PGM	Portable gray map.
SPINNAKER_IMAGE_FILE_FORMAT_PPM	Portable pixmap.
SPINNAKER_IMAGE_FILE_FORMAT_BMP	Bitmap.
SPINNAKER_IMAGE_FILE_FORMAT_JPEG	JPEG.
SPINNAKER_IMAGE_FILE_FORMAT_JPEG2000	JPEG 2000.
SPINNAKER_IMAGE_FILE_FORMAT_TIFF	Tagged image file format.
SPINNAKER_IMAGE_FILE_FORMAT_PNG	Portable network graphics.
SPINNAKER_IMAGE_FILE_FORMAT_RAW	Raw data.
SPINNAKER_IMAGE_FILE_FORMAT_FORCE_32BITS	

13.13.2.5 spinImageStatus

```
enum spinImageStatus
```

Status of images returned from spinImageGetStatus() call.

Enumerator

SPINNAKER_IMAGE_STATUS_UNKNOWN_ERROR	Image has an unknown error.
SPINNAKER_IMAGE_STATUS_NO_ERROR	Image is returned from GetNextImage() call without any errors.
SPINNAKER_IMAGE_STATUS_CRC_CHECK_↩ FAILED	Image failed CRC check.
SPINNAKER_IMAGE_STATUS_DATA_OVERFLOW	Received more data than the size of the image.
SPINNAKER_IMAGE_STATUS_MISSING_PACKETS	Image has missing packets. Potential fixes include enabling jumbo packets and adjusting packet size/delay. For more information see `https://www.flir.↩ com/support-center/iis/machine-vision/application`
SPINNAKER_IMAGE_STATUS_LEADER_↩ BUFFER_SIZE_INCONSISTENT	Image leader is incomplete. Could be caused by missing packet(s). See link above.

Enumerator

SPINNAKER_IMAGE_STATUS_TRAILER_↩ BUFFER_SIZE_INCONSISTENT	Image trailer is incomplete. Could be caused by missing packet(s). See link above.
SPINNAKER_IMAGE_STATUS_PACKETID_↩ INCONSISTENT	Image has an inconsistent packet id. Could be caused by missing packet(s). See link above.
SPINNAKER_IMAGE_STATUS_MISSING_LEADER	Image leader is missing. Could be caused by missing packet(s). See link above.
SPINNAKER_IMAGE_STATUS_MISSING_TRAILER	Image trailer is missing. Could be caused by missing packet(s). See link above.
SPINNAKER_IMAGE_STATUS_DATA_INCOMPLETE	Image data is incomplete. Could be caused by missing packet(s). See link above.
SPINNAKER_IMAGE_STATUS_INFO_↩ INCONSISTENT	Image info is corrupted. Could be caused by missing packet(s). See link above.
SPINNAKER_IMAGE_STATUS_CHUNK_DATA_↩ INVALID	Image chunk data is invalid.
SPINNAKER_IMAGE_STATUS_NO_SYSTEM_↩ RESOURCES	Image cannot be processed due to lack of system resources.

13.13.2.6 spinnakerLogLevel

```
enum spinnakerLogLevel
```

log levels

Enumerator

SPINNAKER_LOG_LEVEL_OFF	
SPINNAKER_LOG_LEVEL_FATAL	
SPINNAKER_LOG_LEVEL_ALERT	
SPINNAKER_LOG_LEVEL_CRIT	
SPINNAKER_LOG_LEVEL_ERROR	
SPINNAKER_LOG_LEVEL_WARN	
SPINNAKER_LOG_LEVEL_NOTICE	
SPINNAKER_LOG_LEVEL_INFO	
SPINNAKER_LOG_LEVEL_DEBUG	
SPINNAKER_LOG_LEVEL_NOTSET	

13.13.2.7 spinStatisticsChannel

```
enum spinStatisticsChannel
```

Channels that allow statistics to be calculated.

Enumerator

SPINNAKER_STATISTICS_CHANNEL_GREY	
SPINNAKER_STATISTICS_CHANNEL_RED	
SPINNAKER_STATISTICS_CHANNEL_GREEN	
SPINNAKER_STATISTICS_CHANNEL_BLUE	
SPINNAKER_STATISTICS_CHANNEL_HUE	
SPINNAKER_STATISTICS_CHANNEL_SATURATION	
SPINNAKER_STATISTICS_CHANNEL_LIGHTNESS	
SPINNAKER_STATISTICS_CHANNEL_NUM_CHANNELS	

13.13.2.8 spinTIFFCompressionMethod

```
enum spinTIFFCompressionMethod
```

Compression method to use for encoding TIFF images.

Enumerator

SPINNAKER_TIFF_COMPRESS_METHOD_NONE	
SPINNAKER_TIFF_COMPRESS_METHOD_PACKBITS	
SPINNAKER_TIFF_COMPRESS_METHOD_DEFLATE	
SPINNAKER_TIFF_COMPRESS_METHOD_ADOBE_DEFLATE	
SPINNAKER_TIFF_COMPRESS_METHOD_CCITTFAX3	
SPINNAKER_TIFF_COMPRESS_METHOD_CCITTFAX4	
SPINNAKER_TIFF_COMPRESS_METHOD_LZW	
SPINNAKER_TIFF_COMPRESS_METHOD_JPG	

13.13.2.9 spinTLPayloadType

```
enum spinTLPayloadType
```

Enumerator

SPINNAKER_TLPAYLOAD_TYPE_UNKNOWN	
SPINNAKER_TLPAYLOAD_TYPE_IMAGE	
SPINNAKER_TLPAYLOAD_TYPE_RAW_DATA	
SPINNAKER_TLPAYLOAD_TYPE_FILE	
SPINNAKER_TLPAYLOAD_TYPE_CHUNK_DATA	
SPINNAKER_TLPAYLOAD_TYPE_JPEG	
SPINNAKER_TLPAYLOAD_TYPE_JPEG2000	
SPINNAKER_TLPAYLOAD_TYPE_H264	
SPINNAKER_TLPAYLOAD_TYPE_CHUNK_ONLY	
SPINNAKER_TLPAYLOAD_TYPE_DEVICE_SPECIFIC	
SPINNAKER_TLPAYLOAD_TYPE_MULTI_PART	

Enumerator

SPINNAKER_TLPAYLOAD_TYPE_CUSTOM_ID	
SPINNAKER_TLPAYLOAD_TYPE_LOSSLESS_COMPRESSED	
SPINNAKER_TLPAYLOAD_TYPE_LOSSY_COMPRESSED	
SPINNAKER_TLPAYLOAD_TYPE_JPEG_LOSSLESS_COMPRESSED	

13.13.2.10 spinTLPixelFormatNamespace

```
enum spinTLPixelFormatNamespace
```

This enum represents the namespace in which the TL specific pixel format resides.

This enum is returned from a captured image when calling spinImageGetTLPixelFormatNamespace(). It can be used to interpret the raw pixel format returned from spinImageGetTLPixelFormat().

See also

spinImageGetTLPixelFormat()
spinImageGetTLPixelFormatNamespace()

Enumerator

SPINNAKER_TLPIXELFORMAT_NAMESPACE_UNKNOWN	
SPINNAKER_TLPIXELFORMAT_NAMESPACE_GEV	
SPINNAKER_TLPIXELFORMAT_NAMESPACE_IIDC	
SPINNAKER_TLPIXELFORMAT_NAMESPACE_PFNC_16BIT	
SPINNAKER_TLPIXELFORMAT_NAMESPACE_PFNC_32BIT	
SPINNAKER_PIXELFORMAT_NAMESPACE_CUSTOM_ID	

13.13.3 Variable Documentation

13.13.3.1 False

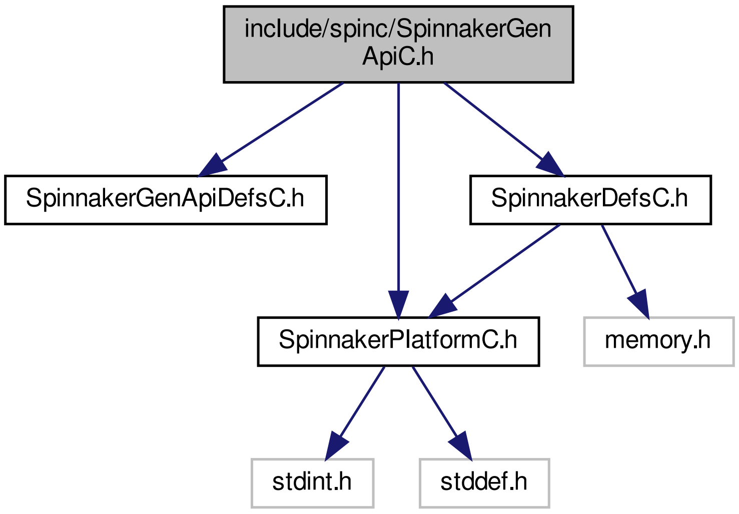
```
const bool8_t False = 0  [static]
```

13.13.3.2 True

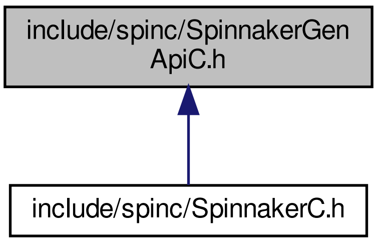
```
const bool8_t True = 1  [static]
```

13.14 include/spinc/SpinnakerGenApiC.h File Reference

Include dependency graph for SpinnakerGenApiC.h:



This graph shows which files directly or indirectly include this file:



Functions

- SPINNAKERC_API spinNodeMapGetNode (spinNodeMapHandle hNodeMap, const char ∗pName, spinNodeHandle ∗phNode)

  Retrieves a node from the nodemap by name.
- SPINNAKERC_API spinNodeMapGetNumNodes (spinNodeMapHandle hNodeMap, size_t ∗pValue)

  Gets the number of nodes in the map.
- SPINNAKERC_API spinNodeMapGetNodeByIndex (spinNodeMapHandle hNodeMap, size_t index, spinNodeHandle ∗phNode)

Retrieves a node from the nodemap by index.
- SPINNAKERC_API spinNodeMapReleaseNode (spinNodeMapHandle hNodeMap, spinNodeHandle hNode)

  Releases the entry node handle.
- SPINNAKERC_API spinNodeMapPoll (spinNodeMapHandle hNodeMap, int64_t timestamp)

  Fires nodes which have a polling time.
- SPINNAKERC_API spinNodeIsImplemented (spinNodeHandle hNode, bool8_t ∗pbResult)

  Checks whether a node is implemented.
- SPINNAKERC_API spinNodeIsReadable (spinNodeHandle hNode, bool8_t ∗pbResult)

  Checks whether a node is readable.
- SPINNAKERC_API spinNodeIsWritable (spinNodeHandle hNode, bool8_t ∗pbResult)

  Checks whether a node is writable.
- SPINNAKERC_API spinNodeIsAvailable (spinNodeHandle hNode, bool8_t ∗pbResult)

  Checks whether a node is available.
- SPINNAKERC_API spinNodeIsEqual (spinNodeHandle hNodeFirst, spinNodeHandle hNodeSecond, bool8_t ∗pbResult)

  Checks whether two nodes are equal.
- SPINNAKERC_API spinNodeGetAccessMode (spinNodeHandle hNode, spinAccessMode ∗pAccessMode)

  Retrieves the access mode of a node (as an enum, spinAccessMode)
- SPINNAKERC_API spinNodeGetName (spinNodeHandle hNode, char ∗pBuf, size_t ∗pBufLen)

  Retrieves the name of a node (no whitespace)
- SPINNAKERC_API spinNodeGetNameSpace (spinNodeHandle hNode, spinNameSpace ∗pNamespace)

  Retrieve the namespace of a node (as an enum, spinNameSpace)
- SPINNAKERC_API spinNodeGetVisibility (spinNodeHandle hNode, spinVisibility ∗pVisibility)

  Retrieves the recommended visibility of a node (as an enum, spinVisibility)
- SPINNAKERC_API spinNodeInvalidateNode (spinNodeHandle hNode)

  Invalidates a node in case its values may have changed, rendering it no longer valid.
- SPINNAKERC_API spinNodeGetCachingMode (spinNodeHandle hNode, spinCachingMode ∗pCaching↩ Mode)

  Retrieves the caching mode of a node (as an enum, spinCachingMode)
- SPINNAKERC_API spinNodeGetToolTip (spinNodeHandle hNode, char ∗pBuf, size_t ∗pBufLen)

  Retrieves a short description of a node.
- SPINNAKERC_API spinNodeGetDescription (spinNodeHandle hNode, char ∗pBuf, size_t ∗pBufLen)

  Retrieves a longer description of a node.
- SPINNAKERC_API spinNodeGetDisplayName (spinNodeHandle hNode, char ∗pBuf, size_t ∗pBufLen)

  Retrieves the display name of a node (whitespace possible)
- SPINNAKERC_API spinNodeGetType (spinNodeHandle hNode, spinNodeType ∗pType)

  Retrieves the type of a node (as an enum, spinNodeType)
- SPINNAKERC_API spinNodeGetPollingTime (spinNodeHandle hNode, int64_t ∗pPollingTime)

  Retrieve the polling time of a node.
- SPINNAKERC_API spinNodeRegisterCallback (spinNodeHandle hNode, spinNodeCallbackFunction pCb↩ Function, spinNodeCallbackHandle ∗phCb)

  Registers a callback to a node.
- SPINNAKERC_API spinNodeDeregisterCallback (spinNodeHandle hNode, spinNodeCallbackHandle hCb)

  Unregisters a callback from a node.
- SPINNAKERC_API spinNodeGetImposedAccessMode (spinNodeHandle hNode, spinAccessMode imposedAccessMode)

  Retrieves the imposed access mode of a node.
- SPINNAKERC_API spinNodeGetImposedVisibility (spinNodeHandle hNode, spinVisibility imposedVisibility)

  Retrieves the imposed visibility of a node.
- SPINNAKERC_API spinNodeToString (spinNodeHandle hNode, char ∗pBuf, size_t ∗pBufLen)

  Retrieves the value of any node type as a c-string.

- SPINNAKERC_API spinNodeToStringEx (spinNodeHandle hNode, bool8_t bVerify, char ∗pBuf, size_t ∗p←↩ BufLen)

    Retrieves the value of any node type as a c-string; manually set whether to verify the node.
- SPINNAKERC_API spinNodeFromString (spinNodeHandle hNode, const char ∗pBuf)

    Sets the value of any node type from a c-string; it is important to ensure that the value of the c-string is appropriate to the node type.
- SPINNAKERC_API spinNodeFromStringEx (spinNodeHandle hNode, bool8_t bVerify, const char ∗pBuf)

    Sets the value of any node type from a c-string; manually set whether to verify the node; ensure the value of the c-string is appropriate to the node type.
- SPINNAKERC_API spinStringSetValue (spinNodeHandle hNode, const char ∗pBuf)

    Sets the value of a string node.
- SPINNAKERC_API spinStringSetValueEx (spinNodeHandle hNode, bool8_t bVerify, const char ∗pBuf)

    Sets the value of a string node; manually set whether to verify the node.
- SPINNAKERC_API spinStringGetValue (spinNodeHandle hNode, char ∗pBuf, size_t ∗pBufLen)

    Retrieves the value of a string node as a c-string.
- SPINNAKERC_API spinStringGetValueEx (spinNodeHandle hNode, bool8_t bVerify, char ∗pBuf, size_t ∗p←↩ BufLen)

    Retrieves the value of a string node as a cstring; manually set whether to verify the node.
- SPINNAKERC_API spinStringGetMaxLength (spinNodeHandle hNode, int64_t ∗pValue)

    Retrieves the maximum length of the c-string to be returned.
- SPINNAKERC_API spinIntegerSetValue (spinNodeHandle hNode, int64_t value)

    Sets the value of an integer node.
- SPINNAKERC_API spinIntegerSetValueEx (spinNodeHandle hNode, bool8_t bVerify, int64_t value)

    Sets the value of an integer node; manually set whether to verify the node.
- SPINNAKERC_API spinIntegerGetValue (spinNodeHandle hNode, int64_t ∗pValue)

    Retrieves the value of an integer node.
- SPINNAKERC_API spinIntegerGetValueEx (spinNodeHandle hNode, bool8_t bVerify, int64_t ∗pValue)

    Retrieves the value of an integer node; manually set whether to verify the node.
- SPINNAKERC_API spinIntegerGetMin (spinNodeHandle hNode, int64_t ∗pValue)

    Retrieves the minimum value of an integer node; all potential values must be greater than or equal to the minimum.
- SPINNAKERC_API spinIntegerGetMax (spinNodeHandle hNode, int64_t ∗pValue)

    Retrieves the maximum value of an integer node; all potential values must be lesser than or equal to the maximum.
- SPINNAKERC_API spinIntegerGetInc (spinNodeHandle hNode, int64_t ∗pValue)

    Retrieves the increment of an integer node; all possible values must be divisible by the increment.
- SPINNAKERC_API spinIntegerGetRepresentation (spinNodeHandle hNode, spinRepresentation ∗pValue)

    Retrieves the numerical representation of the value of a node; i.e.
- SPINNAKERC_API spinFloatSetValue (spinNodeHandle hNode, double value)

    Sets the value of a float node.
- SPINNAKERC_API spinFloatSetValueEx (spinNodeHandle hNode, bool8_t bVerify, double value)

    Sets the value of a float node; manually set whether to verify the node.
- SPINNAKERC_API spinFloatGetValue (spinNodeHandle hNode, double ∗pValue)

    Retrieves the value of a float node.
- SPINNAKERC_API spinFloatGetValueEx (spinNodeHandle hNode, bool8_t bVerify, double ∗pValue)

    Retrieves the value of a float node; manually set whether to verify the node.
- SPINNAKERC_API spinFloatGetMin (spinNodeHandle hNode, double ∗pValue)

    Retrieves the minimum value of a float node; all potential values must be greater than or equal to the minimum.
- SPINNAKERC_API spinFloatGetMax (spinNodeHandle hNode, double ∗pValue)

    Retrieves the maximum value of a float node; all potential values must be lesser than or equal to the maximum.
- SPINNAKERC_API spinFloatGetRepresentation (spinNodeHandle hNode, spinRepresentation ∗pValue)

    Retrieves the numerical representation of the value of a node; i.e.
- SPINNAKERC_API spinFloatGetUnit (spinNodeHandle hNode, char ∗pBuf, size_t ∗pBufLen)

Retrieves the units of the float node value.
- SPINNAKERC_API spinEnumerationGetNumEntries (spinNodeHandle hEnumNode, size_t ∗pValue)

  Retrieves the number of entries of an enum node.
- SPINNAKERC_API spinEnumerationGetEntryByIndex (spinNodeHandle hEnumNode, size_t index, spinNodeHandle ∗phEntry)

  Retrieves an entry node from an enum node using an index.
- SPINNAKERC_API spinEnumerationGetEntryByName (spinNodeHandle hEnumNode, const char ∗pName, spinNodeHandle ∗phEntry)

  Retrieves an entry node from an enum node using the entry's symbolic.
- SPINNAKERC_API spinEnumerationGetCurrentEntry (spinNodeHandle hEnumNode, spinNodeHandle ∗phEntry)

  Retrieves the currently selected entry node from an enum node.
- SPINNAKERC_API spinEnumerationReleaseNode (spinNodeHandle hEnumNode, spinNodeHandle hEntry)

  Releases the entry node from the enum node handle.
- SPINNAKERC_API spinEnumerationSetIntValue (spinNodeHandle hEnumNode, int64_t value)

  Sets a new entry using its integer value retrieved from a call to spinEnumerationEntryGetIntValue(); note that enumeration entry int and enum values are different - int values defined on camera, enum values found in SpinnakerDefsC.h.
- SPINNAKERC_API spinEnumerationSetEnumValue (spinNodeHandle hEnumNode, size_t value)

  Sets a new entry using its enum; note that enumeration entry int and enum values are different - int values defined on camera, enum values found in SpinnakerDefsC.h.
- SPINNAKERC_API spinEnumerationEntryGetIntValue (spinNodeHandle hNode, int64_t ∗pValue)

  Retrieves the integer value of an entry node; note that enumeration entry int and enum values are different - int values defined on camera, enum values found in SpinnakerDefsC.h.
- SPINNAKERC_API spinEnumerationEntryGetEnumValue (spinNodeHandle hNode, size_t ∗pValue)

  Retrieves the enum value (as an integer) of an entry node; note that enumeraiton entry int and enum values are different - int values defined on camera, enum values found in SpinnakerDefsC.h.
- SPINNAKERC_API spinEnumerationEntryGetSymbolic (spinNodeHandle hNode, char ∗pBuf, size_t ∗pBuf←Len)

  Retrieves the symbolic of an entry node as a c-string.
- SPINNAKERC_API spinBooleanSetValue (spinNodeHandle hNode, bool8_t value)

  Sets the value of a boolean node; boolean values are represented by 'True' (which equals '0') and 'False' (which equals '1')
- SPINNAKERC_API spinBooleanGetValue (spinNodeHandle hNode, bool8_t ∗pbValue)

  Retrieves the value of a boolean node; boolean values are represented by 'True' (which equals '0') and 'False' (which equals '1')
- SPINNAKERC_API spinCommandExecute (spinNodeHandle hNode)

  Executes the action associated to a command node.
- SPINNAKERC_API spinCommandIsDone (spinNodeHandle hNode, bool8_t ∗pbValue)

  Retrieves whether or not the action of a command node has completed.
- SPINNAKERC_API spinCategoryGetNumFeatures (spinNodeHandle hCategoryNode, size_t ∗pValue)

  Retrieves the number of a features (or child nodes) or a category node.
- SPINNAKERC_API spinCategoryGetFeatureByIndex (spinNodeHandle hCategoryNode, size_t index, spinNodeHandle ∗phFeature)

  Retrieves a node from a category node using an index.
- SPINNAKERC_API spinCategoryReleaseNode (spinNodeHandle hCategoryNode, spinNodeHandle h←Feature)

  Releases the feature node from the category node.
- SPINNAKERC_API spinRegisterGet (spinNodeHandle hNode, uint8_t ∗pBuf, int64_t length)

  Retrieves the value of a register node.
- SPINNAKERC_API spinRegisterGetEx (spinNodeHandle hNode, bool8_t bVerify, bool8_t bIgnoreCache, uint8_t ∗pBuf, int64_t length)

  Retrieves the value of a register node; manually set whether to verify the node and whether to ignore the cache.
- SPINNAKERC_API spinRegisterGetAddress (spinNodeHandle hNode, int64_t ∗pAddress)

Retrieves the address of a register node.
- SPINNAKERC_API spinRegisterGetLength (spinNodeHandle hNode, int64_t ∗pLength)

  Retrieves the length (in bytes) of the value of a register node.
- SPINNAKERC_API spinRegisterSet (spinNodeHandle hNode, const uint8_t ∗pBuf, int64_t length)

  Sets the value of a register node.
- SPINNAKERC_API spinRegisterSetEx (spinNodeHandle hNode, bool8_t bVerify, const uint8_t ∗pBuf, int64←_t length)

  Sets the value of a register node; manually set whether to verify the node.
- SPINNAKERC_API spinRegisterSetReference (spinNodeHandle hNode, spinNodeHandle hRef)

  Uses a second node as a reference for a register node.

13.14.1 Function Documentation

13.14.1.1 spinBooleanGetValue()

```
SPINNAKERC_API spinBooleanGetValue (
            spinNodeHandle hNode,
            bool8_t ∗ pbValue )
```

Retrieves the value of a boolean node; boolean values are represented by 'True' (which equals '0') and 'False' (which equals '1')

See also

spinError

Parameters

hNode	The boolean node of the value to read
pValue	The boolean pointer in which the value is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.2 spinBooleanSetValue()

```
SPINNAKERC_API spinBooleanSetValue (
            spinNodeHandle hNode,
            bool8_t value )
```

Sets the value of a boolean node; boolean values are represented by 'True' (which equals '0') and 'False' (which equals '1')

See also

spinError

Parameters

hNode	The boolean node having its value changed
value	The boolean value to set

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.3 spinCategoryGetFeatureByIndex()

```
SPINNAKERC_API spinCategoryGetFeatureByIndex (
            spinNodeHandle hCategoryNode,
            size_t index,
            spinNodeHandle * phFeature )
```

Retrieves a node from a category node using an index.

See also

spinError

Parameters

hCategoryNode	The category node of the node to retrieve
index	The index of the feature node
phFeature	The node handle pointer in which the feature node is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.4 spinCategoryGetNumFeatures()

```
SPINNAKERC_API spinCategoryGetNumFeatures (
            spinNodeHandle hCategoryNode,
            size_t * pValue )
```

Retrieves the number of a features (or child nodes) or a category node.

See also

spinError

Parameters

hCategoryNode	The category node where the features to be counted are
pValue	The unsigned integer pointer in which the number of features is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.5 spinCategoryReleaseNode()

```
SPINNAKERC_API spinCategoryReleaseNode (
            spinNodeHandle hCategoryNode,
            spinNodeHandle hFeature )
```

Releases the feature node from the category node.

Make sure node handle is cleaned up properly by setting it to NULL after the node is released If this function is not explicitly called, the handle will be released upon the release of the camera handle.

See also

spinCameraRelease

spinError

Parameters

hCategoryNode	The category node handle from which the feature node is retrieved
hFeature	The feature node handle to be released

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.6 spinCommandExecute()

```
SPINNAKERC_API spinCommandExecute (
            spinNodeHandle hNode )
```

Executes the action associated to a command node.

See also

spinError

Parameters

hNode	The command node to execute

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.7 spinCommandIsDone()

```
SPINNAKERC_API spinCommandIsDone (
            spinNodeHandle hNode,
            bool8_t * pbValue )
```

Retrieves whether or not the action of a command node has completed.

See also

spinError

Parameters

hNode	The command node to check
pValue	The boolean pointer to return whether or not the command has completed

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.8 spinEnumerationEntryGetEnumValue()

```
SPINNAKERC_API spinEnumerationEntryGetEnumValue (
            spinNodeHandle hNode,
            size_t * pValue )
```

Retrieves the enum value (as an integer) of an entry node; note that enumeraiton entry int and enum values are different - int values defined on camera, enum values found in SpinnakerDefsC.h.

See also

spinEnumerationSetEnumValue()

spinError

Parameters

hNode	The entry node of the enum value to retrieve
pValue	The unsigned integer pointer in which the enum value of the entry is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.9 spinEnumerationEntryGetIntValue()

```
SPINNAKERC_API spinEnumerationEntryGetIntValue (
            spinNodeHandle hNode,
            int64_t * pValue )
```

Retrieves the integer value of an entry node; note that enumeration entry int and enum values are different - int values defined on camera, enum values found in SpinnakerDefsC.h.

See also

spinEnumerationSetIntValue()

spinError

Parameters

hNode	The entry node of the integer value to retrieve
pValue	The integer pointer in which the integer value of the entry is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.10 spinEnumerationEntryGetSymbolic()

```
SPINNAKERC_API spinEnumerationEntryGetSymbolic (
            spinNodeHandle hNode,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the symbolic of an entry node as a c-string.

See also

spinError

Parameters

hNode	The entry node of the symbolic to retrieve
pBuf	The c-string character buffer in which the symbolic of the entry node is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.11 spinEnumerationGetCurrentEntry()

```
SPINNAKERC_API spinEnumerationGetCurrentEntry (
            spinNodeHandle hEnumNode,
            spinNodeHandle * phEntry )
```

Retrieves the currently selected entry node from an enum node.

See also

spinError

Parameters

hEnumNode	The enum node from which the current entry node is retrieved
phEntry	The node handle pointer in which the current entry node is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.12 spinEnumerationGetEntryByIndex()

```
SPINNAKERC_API spinEnumerationGetEntryByIndex (
            spinNodeHandle hEnumNode,
            size_t index,
            spinNodeHandle * phEntry )
```

Retrieves an entry node from an enum node using an index.

See also

spinError

Parameters

hEnumNode	The enum node from which the entry node is retrieved
index	The index of the entry node
phEntry	The node handle pointer in which the entry node is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.13 spinEnumerationGetEntryByName()

```
SPINNAKERC_API spinEnumerationGetEntryByName (
            spinNodeHandle hEnumNode,
            const char * pName,
            spinNodeHandle * phEntry )
```

Retrieves an entry node from an enum node using the entry's symbolic.

See also

spinError

Parameters

hEnumNode	The enum node from which the entry node is retrieved
pName	The name of the entry node
phEntry	The node handle pointer in which the entry node is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.14 spinEnumerationGetNumEntries()

```
SPINNAKERC_API spinEnumerationGetNumEntries (
            spinNodeHandle hEnumNode,
            size_t * pValue )
```

Retrieves the number of entries of an enum node.

See also

spinError

Parameters

hEnumNode	The enum node where the entries to be counted are
pValue	The unsigned integer pointer in which the number of entries is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.15 spinEnumerationReleaseNode()

```
SPINNAKERC_API spinEnumerationReleaseNode (
            spinNodeHandle hEnumNode,
            spinNodeHandle hEntry )
```

Releases the entry node from the enum node handle.

Make sure node handle is cleaned up properly by setting it to NULL after the node is released If this function is not explicitly called, the handle will be released upon the release of the camera handle.

See also

spinCameraRelease

spinError

Parameters

hEnumNode	The enum node from which the current entry node is retrieved
hEntry	The entry node handle to be released

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.16 spinEnumerationSetEnumValue()

```
SPINNAKERC_API spinEnumerationSetEnumValue (
            spinNodeHandle hEnumNode,
            size_t value )
```

Sets a new entry using its enum; note that enumeration entry int and enum values are different - int values defined on camera, enum values found in SpinnakerDefsC.h.

See also

spinEnumerationEntryGetEnumValue()

spinError

Parameters

hEnumNode	The enum node have its entry changed
value	The enum value of the entry node to set; this corresponds to its integer value created in the library

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.17 spinEnumerationSetIntValue()

```
SPINNAKERC_API spinEnumerationSetIntValue (
            spinNodeHandle hEnumNode,
            int64_t value )
```

Sets a new entry using its integer value retrieved from a call to spinEnumerationEntryGetIntValue(); note that enumeration entry int and enum values are different - int values defined on camera, enum values found in SpinnakerDefsC.h.

See also

spinEnumerationEntryGetIntValue()

spinError

Parameters

hEnumNode	The enum node having its entry changed
value	The integer value of the entry node to set; this corresponds to the integer value internal to the camera

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.18 spinFloatGetMax()

```
SPINNAKERC_API spinFloatGetMax (
            spinNodeHandle hNode,
            double * pValue )
```

Retrieves the maximum value of a float node; all potential values must be lesser than or equal to the maximum.

See also

spinError

Parameters

hNode	The float node of the maximum value to retrieve
pValue	The double pointer in which the maximum value is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.19 spinFloatGetMin()

```
SPINNAKERC_API spinFloatGetMin (
            spinNodeHandle hNode,
            double * pValue )
```

Retrieves the minimum value of a float node; all potential values must be greater than or equal to the minimum.

See also

spinError

Parameters

hNode	The float node of the minimum value to retrieve
pValue	The double pointer in which the minimum value is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.20 spinFloatGetRepresentation()

```
SPINNAKERC_API spinFloatGetRepresentation (
            spinNodeHandle hNode,
            spinRepresentation * pValue )
```

Retrieves the numerical representation of the value of a node; i.e.

linear, logarithmic, hexidecimal, MAC address, etc.

See also

spinError

Parameters

hNode	The float node of the numerical representation to retrieve
pValue	The representation enum pointer in which the type of numerical representation is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.21 spinFloatGetUnit()

```
SPINNAKERC_API spinFloatGetUnit (
             spinNodeHandle hNode,
             char * pBuf,
             size_t * pBufLen )
```

Retrieves the units of the float node value.

See also

spinError

Parameters

hNode	The float node of the units to retrieve
pBuf	The c-string character buffer in which the value units are returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.22 spinFloatGetValue()

```
SPINNAKERC_API spinFloatGetValue (
             spinNodeHandle hNode,
             double * pValue )
```

Retrieves the value of a float node.

See also

spinError

Parameters

hNode	The float node of the value to read
pValue	The double pointer in which the value is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.23 spinFloatGetValueEx()

```
SPINNAKERC_API spinFloatGetValueEx (
            spinNodeHandle hNode,
            bool8_t bVerify,
            double * pValue )
```

Retrieves the value of a float node; manually set whether to verify the node.

See also

spinError

Parameters

hNode	The float node of the value to read
pValue	The double pointer in which the value is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.24 spinFloatSetValue()

```
SPINNAKERC_API spinFloatSetValue (
            spinNodeHandle hNode,
            double value )
```

Sets the value of a float node.

See also

spinError

Parameters

hNode	The float node having its value changed
value	The float value to set

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.25 spinFloatSetValueEx()

```
SPINNAKERC_API spinFloatSetValueEx (
            spinNodeHandle hNode,
            bool8_t bVerify,
            double value )
```

Sets the value of a float node; manually set whether to verify the node.

See also

spinError

Parameters

hNode	The float node having its value changed
bVerify	The boolean of whether to verify the node
value	The float value to set

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.26 spinIntegerGetInc()

```
SPINNAKERC_API spinIntegerGetInc (
            spinNodeHandle hNode,
            int64_t * pValue )
```

Retrieves the increment of an integer node; all possible values must be divisible by the increment.

See also

spinError

Parameters

hNode	The integer node of the increment to retrieve
pValue	The integer pointer in which the increment is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.27 spinIntegerGetMax()

```
SPINNAKERC_API spinIntegerGetMax (
            spinNodeHandle hNode,
            int64_t * pValue )
```

Retrieves the maximum value of an integer node; all potential values must be lesser than or equal to the maximum.

See also

spinError

Parameters

hNode	The integer node of the maximum value to retrieve
pValue	The integer pointer in which the maximum value is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.28 spinIntegerGetMin()

```
SPINNAKERC_API spinIntegerGetMin (
            spinNodeHandle hNode,
            int64_t * pValue )
```

Retrieves the minimum value of an integer node; all potential values must be greater than or equal to the minimum.

See also

spinError

Parameters

hNode	The integer node of the minimum value to retrieve
pValue	The integer pointer in which the minimum value is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.29 spinIntegerGetRepresentation()

```
SPINNAKERC_API spinIntegerGetRepresentation (
            spinNodeHandle hNode,
            spinRepresentation * pValue )
```

Retrieves the numerical representation of the value of a node; i.e.

linear, logarithmic, hexidecimal, MAC address, etc.

See also

spinError

Parameters

hNode	The integer node of the numerical representation to retrieve
pValue	The representation enum pointer in which the type of numerical representation is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.30 spinIntegerGetValue()

```
SPINNAKERC_API spinIntegerGetValue (
            spinNodeHandle hNode,
            int64_t * pValue )
```

Retrieves the value of an integer node.

See also

spinError

Parameters

hNode	The integer node of the value to read
pValue	The integer pointer in which the value is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.31 spinIntegerGetValueEx()

```
SPINNAKERC_API spinIntegerGetValueEx (
            spinNodeHandle hNode,
            bool8_t bVerify,
            int64_t * pValue )
```

Retrieves the value of an integer node; manually set whether to verify the node.

See also

spinError

Parameters

hNode	The integer node of the value to read
bVerify	The boolean of whether to verify the node
pValue	The integer pointer in which the value is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.32 spinIntegerSetValue()

```
SPINNAKERC_API spinIntegerSetValue (
            spinNodeHandle hNode,
            int64_t value )
```

Sets the value of an integer node.

See also

spinError

Parameters

hNode	The integer node having its value changed
value	The integer value to set

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.33 spinIntegerSetValueEx()

```
SPINNAKERC_API spinIntegerSetValueEx (
            spinNodeHandle hNode,
            bool8_t bVerify,
            int64_t value )
```

Sets the value of an integer node; manually set whether to verify the node.

See also

spinError

Parameters

hNode	The integer node having its value changed
bVerify	The boolean of whether to verify the node
value	The integer value to set

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.34 spinNodeDeregisterCallback()

```
SPINNAKERC_API spinNodeDeregisterCallback (
            spinNodeHandle hNode,
            spinNodeCallbackHandle hCb )
```

Unregisters a callback from a node.

See also

spinError

Parameters

hNode	The node from which to unregister the callback
hCb	The callback handle to unregister

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.35 spinNodeFromString()

```
SPINNAKERC_API spinNodeFromString (
            spinNodeHandle hNode,
            const char * pBuf )
```

Sets the value of any node type from a c-string; it is important to ensure that the value of the c-string is appropriate to the node type.

See also

spinError

Parameters

hNode	The node having its value changed
pBuf	The c-string of the value to set

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.36 spinNodeFromStringEx()

```
SPINNAKERC_API spinNodeFromStringEx (
            spinNodeHandle hNode,
            bool8_t bVerify,
            const char * pBuf )
```

Sets the value of any node type from a c-string; manually set whether to verify the node; ensure the value of the c-string is appropriate to the node type.

See also

spinError

Parameters

hNode	The node having its value changed
bVerify	The boolean of whether to verify the node
pBuf	The c-string of the value to set

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.37 spinNodeGetAccessMode()

```
SPINNAKERC_API spinNodeGetAccessMode (
            spinNodeHandle hNode,
            spinAccessMode * pAccessMode )
```

Retrieves the access mode of a node (as an enum, spinAccessMode)

See also

spinError

spinAccessMode

Parameters

hNode	The node of the access mode to retrieve
pAccessMode	The access mode enum pointer in which the access mode is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.38 spinNodeGetCachingMode()

```
SPINNAKERC_API spinNodeGetCachingMode (
            spinNodeHandle hNode,
            spinCachingMode * pCachingMode )
```

Retrieves the caching mode of a node (as an enum, spinCachingMode)

See also

spinError

spinCachingMode

Parameters

hNode	The node of the caching mode to retrieve
pCachingMode	The caching mode enum pointer in which the caching mode is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.39 spinNodeGetDescription()

```
SPINNAKERC_API spinNodeGetDescription (
            spinNodeHandle hNode,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves a longer description of a node.

See also

spinError

Parameters

hNode	The node of the description to retrieve
pBuf	The c-string character buffer in which the longer descrition of the node is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.40 spinNodeGetDisplayName()

```
SPINNAKERC_API spinNodeGetDisplayName (
            spinNodeHandle hNode,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the display name of a node (whitespace possible)

See also

spinError

Parameters

hNode	The node of the display name to retrieve
pBuf	The c-string character buffer in which the display name of the node is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.41 spinNodeGetImposedAccessMode()

```
SPINNAKERC_API spinNodeGetImposedAccessMode (
            spinNodeHandle hNode,
            spinAccessMode imposedAccessMode )
```

Retrieves the imposed access mode of a node.

See also

spinError

Parameters

hNode	The node of the imposed access mode to retrieve
imposedAccessMode	The access mode enum pointer in which the imposed access mode is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.42 spinNodeGetImposedVisibility()

```
SPINNAKERC_API spinNodeGetImposedVisibility (
            spinNodeHandle hNode,
            spinVisibility imposedVisibility )
```

Retrieves the imposed visibility of a node.

See also

spinError

Parameters

hNode	The node of the visibility to impose
imposedVisibility	The visibility enum pointer in which the imposed visibility is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.43 spinNodeGetName()

```
SPINNAKERC_API spinNodeGetName (
             spinNodeHandle hNode,
             char * pBuf,
             size_t * pBufLen )
```

Retrieves the name of a node (no whitespace)

See also

spinError

Parameters

hNode	The node of the name to retrieve
pBuf	The c-string character buffer in which the name of the node is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.44 spinNodeGetNameSpace()

```
SPINNAKERC_API spinNodeGetNameSpace (
             spinNodeHandle hNode,
             spinNameSpace * pNamespace )
```

Retrieve the namespace of a node (as an enum, spinNameSpace)

See also

spinError

spinNameSpace

Parameters

hNode	The node of the namespace to retrieve
pNamespace	The namespace enum pointer in which the namespace is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.45 spinNodeGetPollingTime()

```
SPINNAKERC_API spinNodeGetPollingTime (
            spinNodeHandle hNode,
            int64_t * pPollingTime )
```

Retrieve the polling time of a node.

See also

spinError

Parameters

hNode	The node of the polling time to retrieve
pPollingTime	The integer pointer in which the polling time is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.46 spinNodeGetToolTip()

```
SPINNAKERC_API spinNodeGetToolTip (
            spinNodeHandle hNode,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves a short description of a node.

See also

spinError

Parameters

hNode	The node of the tooltip to retrieve
pBuf	The c-string character buffer in which the short description of the node is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.47 spinNodeGetType()

```
SPINNAKERC_API spinNodeGetType (
            spinNodeHandle hNode,
            spinNodeType * pType )
```

Retrieves the type of a node (as an enum, spinNodeType)

See also

spinError

spinNodeType

Parameters

hNode	The node of the node type to retrieve
pType	The node type enum pointer in which the type of node is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.48 spinNodeGetVisibility()

```
SPINNAKERC_API spinNodeGetVisibility (
            spinNodeHandle hNode,
            spinVisibility * pVisibility )
```

Retrieves the recommended visibility of a node (as an enum, spinVisibility)

See also

spinError

spinVisibility

Parameters

hNode	The node of the visibility to retrieve
pVisibility	The visibility enum pointer in which the visibility is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.49 spinNodeInvalidateNode()

```
SPINNAKERC_API spinNodeInvalidateNode (
            spinNodeHandle hNode )
```

Invalidates a node in case its values may have changed, rendering it no longer valid.

See also

spinError

Parameters

hNode	The node whose values may have changed

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.50 spinNodeIsAvailable()

```
SPINNAKERC_API spinNodeIsAvailable (
            spinNodeHandle hNode,
            bool8_t * pbResult )
```

Checks whether a node is available.

See also

spinError

Parameters

hNode	The node to check
pbResult	The boolean pointer to return whether or not the node is available

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.51 spinNodeIsEqual()

```
SPINNAKERC_API spinNodeIsEqual (
            spinNodeHandle hNodeFirst,
            spinNodeHandle hNodeSecond,
            bool8_t * pbResult )
```

Checks whether two nodes are equal.

See also

spinError

Parameters

hNodeFirst	The first node to check
hNodeSecond	The second node to check
pbResult	The boolean pointer to return whether or not the two nodes are equal

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.52 spinNodeIsImplemented()

```
SPINNAKERC_API spinNodeIsImplemented (
            spinNodeHandle hNode,
            bool8_t * pbResult )
```

Checks whether a node is implemented.

See also

spinError

Parameters

hNode	The node to check
pbResult	The boolean pointer to return whether or not the node is implemented

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.53 spinNodeIsReadable()

```
SPINNAKERC_API spinNodeIsReadable (
             spinNodeHandle hNode,
             bool8_t * pbResult )
```

Checks whether a node is readable.

See also

spinError

Parameters

hNode	The node to check
pbResult	The boolean pointer to return whether or not the node is readable

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.54 spinNodeIsWritable()

```
SPINNAKERC_API spinNodeIsWritable (
             spinNodeHandle hNode,
             bool8_t * pbResult )
```

Checks whether a node is writable.

See also

spinError

Parameters

hNode	The node to check
pbResult	The boolean pointer to return whether or not the node is writable

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.55 spinNodeMapGetNode()

```
SPINNAKERC_API spinNodeMapGetNode (
            spinNodeMapHandle hNodeMap,
            const char * pName,
            spinNodeHandle * phNode )
```

Retrieves a node from the nodemap by name.

See also

spinError

Parameters

hNodeMap	The node map where the node is
pName	The name of the node
phNode	The node handle pointer in which the node is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.56 spinNodeMapGetNodeByIndex()

```
SPINNAKERC_API spinNodeMapGetNodeByIndex (
            spinNodeMapHandle hNodeMap,
            size_t index,
            spinNodeHandle * phNode )
```

Retrieves a node from the nodemap by index.

See also

spinError

Parameters

hNodeMap	The node map where the node is
index	The index of the node
phNode	The node handle pointer in which the node is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.57 spinNodeMapGetNumNodes()

```
SPINNAKERC_API spinNodeMapGetNumNodes (
             spinNodeMapHandle hNodeMap,
             size_t * pValue )
```

Gets the number of nodes in the map.

See also

spinError

Parameters

hNodeMap	The node map where the nodes to be counted are
pValue	The unsigned integer pointer in which the number of nodes is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.58 spinNodeMapPoll()

```
SPINNAKERC_API spinNodeMapPoll (
             spinNodeMapHandle hNodeMap,
             int64_t timestamp )
```

Fires nodes which have a polling time.

See also

spinError

Parameters

hNodeMap	The nodemap to poll
timestamp	The timestamp

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.59 spinNodeMapReleaseNode()

```
SPINNAKERC_API spinNodeMapReleaseNode (
            spinNodeMapHandle hNodeMap,
            spinNodeHandle hNode )
```

Releases the entry node handle.

Make sure node handle is cleaned up properly by setting it to NULL after the node is released. If this function is not explicitly called, the handle will be released upon the release of the camera handle.

See also

spinCameraRelease

spinError

Parameters

hNodeMap	The node map from which the node handle is retrieved
hNode	The node handle to be released

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.60 spinNodeRegisterCallback()

```
SPINNAKERC_API spinNodeRegisterCallback (
            spinNodeHandle hNode,
            spinNodeCallbackFunction pCbFunction,
            spinNodeCallbackHandle * phCb )
```

Registers a callback to a node.

See also

spinError

Parameters

hNode	The node on which to register the callback
pCbFunction	The function pointer of the function that will execute when the callback is triggered; must match signature "void spinNodeCallbackFunction(spinNodeHandle hNode)"
phCb	The callback handle pointer in which the callback is returned; used to unregister callbacks

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.61 spinNodeToString()

```
SPINNAKERC_API spinNodeToString (
             spinNodeHandle hNode,
             char * pBuf,
             size_t * pBufLen )
```

Retrieves the value of any node type as a c-string.

See also

spinError

Parameters

hNode	The node of the value to read
pBuf	The c-string character buffer in which the value of the node is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.62 spinNodeToStringEx()

```
SPINNAKERC_API spinNodeToStringEx (
             spinNodeHandle hNode,
             bool8_t bVerify,
             char * pBuf,
             size_t * pBufLen )
```

Retrieves the value of any node type as a c-string; manually set whether to verify the node.

See also

spinError

Parameters

hNode	The node of the value to read
bVerify	The boolean of whether to verify the node
pBuf	The c-string character buffer in which the value of the node is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.63 spinRegisterGet()

```
SPINNAKERC_API spinRegisterGet (
            spinNodeHandle hNode,
            uint8_t * pBuf,
            int64_t length )
```

Retrieves the value of a register node.

See also

spinError

Parameters

hNode	The register node of the value to retrieve
pBuf	The unsigned integer buffer in which the value is returned
length	The integer pointer in which the length of the register array is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.64 spinRegisterGetAddress()

```
SPINNAKERC_API spinRegisterGetAddress (
            spinNodeHandle hNode,
            int64_t * pAddress )
```

Retrieves the address of a register node.

See also

spinError

Parameters

hNode	The register node of the address to retrieve
pAddress	The integer pointer in which the address is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.65 spinRegisterGetEx()

```
SPINNAKERC_API spinRegisterGetEx (
            spinNodeHandle hNode,
            bool8_t bVerify,
            bool8_t bIgnoreCache,
            uint8_t * pBuf,
            int64_t length )
```

Retrieves the value of a register node; manually set whether to verify the node and whether to ignore the cache.

See also

spinError

Parameters

hNode	The register node of the value to retrieve
bVerify	The boolean of whether to verify the node
IgnoreCache	The boolean of whether to ignore the cache
pBuf	The unsigned integer buffer in which the value is returned
length	The integer pointer in which the length of the register array is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.66 spinRegisterGetLength()

```
SPINNAKERC_API spinRegisterGetLength (
            spinNodeHandle hNode,
            int64_t * pLength )
```

Retrieves the length (in bytes) of the value of a register node.

See also

spinError

Parameters

hNode	The register node of the length to retrieve
plength	The integer in which the number of bytes is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.67 spinRegisterSet()

```
SPINNAKERC_API spinRegisterSet (
            spinNodeHandle hNode,
            const uint8_t * pBuf,
            int64_t length )
```

Sets the value of a register node.

See also

spinError

Parameters

hNode	The register node of the value to set
pBuf	The unsigned integer buffer of the value to set
length	The number of bytes of the value to set

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.68 spinRegisterSetEx()

```
SPINNAKERC_API spinRegisterSetEx (
            spinNodeHandle hNode,
            bool8_t bVerify,
            const uint8_t * pBuf,
            int64_t length )
```

Sets the value of a register node; manually set whether to verify the node.

See also

spinError

Parameters

hNode	The register node of the value to set
bVerify	The boolean of whether to verify the node
pBuf	The unsigned integer buffer of the value to set
length	The number of bytes of the value to set

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.69 spinRegisterSetReference()

```
SPINNAKERC_API spinRegisterSetReference (
            spinNodeHandle hNode,
            spinNodeHandle hRef )
```

Uses a second node as a reference for a register node.

See also

spinError

Parameters

hNode	The register node that houses the reference
hRef	The reference node

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.70 spinStringGetMaxLength()

```
SPINNAKERC_API spinStringGetMaxLength (
            spinNodeHandle hNode,
            int64_t * pValue )
```

Retrieves the maximum length of the c-string to be returned.

See also

spinError

Parameters

hNode	The string node of the length to retrieve
pValue	The integer pointer in which the maximum length of the c-string is returned

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.71 spinStringGetValue()

```
SPINNAKERC_API spinStringGetValue (
            spinNodeHandle hNode,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the value of a string node as a c-string.

See also

spinError

Parameters

hNode	The string node of the value to read
pBuf	The c-string character buffer in which the value of the node is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.72 spinStringGetValueEx()

```
SPINNAKERC_API spinStringGetValueEx (
            spinNodeHandle hNode,
            bool8_t bVerify,
            char * pBuf,
            size_t * pBufLen )
```

Retrieves the value of a string node as a cstring; manually set whether to verify the node.

See also

spinError

Parameters

hNode	The string node of the value to read
bVerify	The boolean of whether to verify the node
pBuf	The c-string character buffer in which the value of the node is returned
pBufLen	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.73 spinStringSetValue()

```
SPINNAKERC_API spinStringSetValue (
            spinNodeHandle hNode,
            const char * pBuf )
```

Sets the value of a string node.

See also

spinError

Parameters

hNode	The string node having its value changed
pBuf	The c-string of the value to set

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.14.1.74 spinStringSetValueEx()

```
SPINNAKERC_API spinStringSetValueEx (
            spinNodeHandle hNode,
            bool8_t bVerify,
            const char * pBuf )
```

Sets the value of a string node; manually set whether to verify the node.

See also

spinError

Parameters

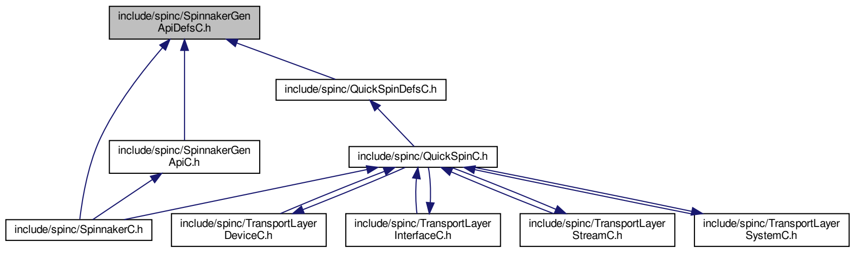
hNode	The string node having its value changed
bVerify	The boolean of whether to verify the node
pBuf	The c-string of the value to set

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.15 include/spinc/SpinnakerGenApiDefsC.h File Reference

This graph shows which files directly or indirectly include this file:


include/spinc/SpinnakerGenApiDefsC.h
include/spinc/QuickSpinDefsC.h
include/spinc/SpinnakerGenApiC.h
include/spinc/QuickSpinC.h
include/spinc/SpinnakerC.h
include/spinc/TransportLayerDeviceC.h
include/spinc/TransportLayerInterfaceC.h
include/spinc/TransportLayerStreamC.h
include/spinc/TransportLayerSystemC.h

Typedefs

- typedef void ∗ spinNodeMapHandle

  Handle for nodemap functionality.
- typedef void ∗ spinNodeHandle

  Handle for node functionality.
- typedef void ∗ spinNodeCallbackHandle

  Handle for callback functionality.
- typedef void(∗ spinNodeCallbackFunction) (spinNodeHandle hNode)

  Function signatures are used to create and trigger callbacks and events.

Enumerations

- enum spinNodeType {
  ValueNode ,
  BaseNode ,
  IntegerNode ,
  BooleanNode ,
  FloatNode ,
  CommandNode ,
  StringNode ,
  RegisterNode ,
  EnumerationNode ,
  EnumEntryNode ,
  CategoryNode ,
  PortNode ,
  UnknownNode = -1 }

- enum spinSign {
  Signed ,
  Unsigned ,
  _UndefinedSign }
- enum spinAccessMode {
  NI ,
  NA ,
  WO ,
  RO ,
  RW ,
  _UndefinedAccesMode ,
  _CycleDetectAccesMode }
- enum spinVisibility {
  Beginner = 0 ,
  Expert = 1 ,
  Guru = 2 ,
  Invisible = 3 ,
  _UndefinedVisibility = 99 }
- enum spinCachingMode {
  NoCache ,
  WriteThrough ,
  WriteAround ,
  _UndefinedCachingMode }
- enum spinRepresentation {
  Linear ,
  Logarithmic ,
  Boolean ,
  PureNumber ,
  HexNumber ,
  IPV4Address ,
  MACAddress ,
  _UndefinedRepresentation }

  recommended representation of a node value
- enum spinEndianess {
  BigEndian ,
  LittleEndian ,
  _UndefinedEndian }

  Endianess of a value in a register.
- enum spinNameSpace {
  Custom ,
  Standard ,
  _UndefinedNameSpace }

  Defines if a node name is standard or custom.
- enum spinStandardNameSpace {
  None ,
  GEV ,
  IIDC ,
  CL ,
  USB ,
  _UndefinedStandardNameSpace }

  Defines from which standard namespace a node name comes from.
- enum spinYesNo {
  Yes = 1 ,
  No = 0 ,
  _UndefinedYesNo = 2 }

  Defines the chices of a Yes/No alternaitve.

- enum spinSlope {
  Increasing ,
  Decreasing ,
  Varying ,
  Automatic ,
  _UndefinedESlope }

  typedef for fomula type
- enum spinXMLValidation {
  xvLoad = 0x00000001L ,
  xvCycles = 0x00000002L ,
  xvSFNC = 0x00000004L ,
  xvDefault = 0x00000000L ,
  xvAll = 0xffffffffL ,
  _UndefinedEXMLValidation = 0x8000000L }

  typedef describing the different validity checks which can be performed on an XML file
- enum spinDisplayNotation {
  fnAutomatic ,
  fnFixed ,
  fnScientific ,
  _UndefinedEDisplayNotation }

  typedef for float notation
- enum spinInterfaceType {
  intfIValue ,
  intfIBase ,
  intfIInteger ,
  intfIBoolean ,
  intfICommand ,
  intfIFloat ,
  intfIString ,
  intfIRegister ,
  intfICategory ,
  intfIEnumeration ,
  intfIEnumEntry ,
  intfIPort }

  typedef for interface type
- enum spinLinkType {
  ctAllDependingNodes ,
  ctAllTerminalNodes ,
  ctInvalidators ,
  ctReadingChildren ,
  ctWritingChildren ,
  ctDependingChildren }

  typedef for link type
- enum spinIncMode {
  noIncrement ,
  fixedIncrement ,
  listIncrement }

  typedef for increment mode
- enum spinInputDirection {
  idFrom ,
  idTo ,
  idNone }

  typedef for link type

13.15.1 Typedef Documentation

13.15.1.1 spinNodeCallbackFunction

```
typedef void(* spinNodeCallbackFunction) (spinNodeHandle hNode)
```

Function signatures are used to create and trigger callbacks and events.

13.15.1.2 spinNodeCallbackHandle

```
typedef void* spinNodeCallbackHandle
```

Handle for callback functionality.

Created by calling spinNodeRegisterCallback(), which requires a call to spinNodeUnregisterCallback() destroy.

13.15.1.3 spinNodeHandle

```
typedef void* spinNodeHandle
```

Handle for node functionality.

Created by calling spinNodeMapGetNode(). No need to release, clear, or destroy.

13.15.1.4 spinNodeMapHandle

```
typedef void* spinNodeMapHandle
```

Handle for nodemap functionality.

Created by calling spinCameraGetNodemap(), spinCameraGetTLDeviceNodeMap(), spinCameraGetTLStreamNodeMap() or spinInterfaceGetTLNodeMap(). No need to release, clear, or destroy.

13.15.2 Enumeration Type Documentation

13.15.2.1 spinAccessMode

```
enum spinAccessMode
```

Enumerator

NI	
NA	
WO	
RO	
RW	
_UndefinedAccesMode	
_CycleDetectAccesMode	

13.15.2.2 spinCachingMode

```
enum spinCachingMode
```

Enumerator

NoCache	
WriteThrough	
WriteAround	
_UndefinedCachingMode	

13.15.2.3 spinDisplayNotation

```
enum spinDisplayNotation
```

typedef for float notation

Enumerator

fnAutomatic	
fnFixed	the notation if either scientific or fixed depending on what is shorter
fnScientific	the notation is fixed, e.g. 123.4
_UndefinedEDisplayNotation	the notation is scientific, e.g. 1.234e2 Object is not yet initialized

13.15.2.4 spinEndianess

```
enum spinEndianess
```

Endianess of a value in a register.

Enumerator

BigEndian	Register is big endian.
LittleEndian	Register is little endian.
_UndefinedEndian	Object is not yet initialized.

13.15.2.5 spinIncMode

```
enum spinIncMode
```

typedef for increment mode

Enumerator

noIncrement	
fixedIncrement	
listIncrement	

13.15.2.6 spinInputDirection

```
enum spinInputDirection
```

typedef for link type

Enumerator

idFrom	
idTo	Indicates a swiss knife that it is used as worker for a converter computing FROM
idNone	Indicates a swiss knife that it is used as worker for a converter computing TO SwissKnife is not used within a converter

13.15.2.7 spinInterfaceType

`enum spinInterfaceType`

typedef for interface type

Enumerator

intfIValue	
intfIBase	IValue interface
intfIInteger	IBase interface
intfIBoolean	IInteger interface
intfICommand	IBoolean interface
intfIFloat	ICommand interface
intfIString	IFloat interface
intfIRegister	IString interface
intfICategory	IRegister interface
intfIEnumeration	ICategory interface
intfIEnumEntry	IEnumeration interface
intfIPort	IEnumEntry interface IPort interface

13.15.2.8 spinLinkType

`enum spinLinkType`

typedef for link type

Enumerator

ctAllDependingNodes	
ctAllTerminalNodes	All nodes which will be invalidated if this node becomes invalid
ctInvalidators	All terminal nodes which may be written to by this node
ctReadingChildren	List of references to nodes which may invalidate this node
ctWritingChildren	All child nodes which influence this node's AccessMode
ctDependingChildren	All child nodes which may be written to All child nodes which will cause this node to be invalidated

13.15.2.9 spinNameSpace

```
enum spinNameSpace
```

Defines if a node name is standard or custom.

Enumerator

Custom	name resides in custom namespace
Standard	name resides in one of the standard namespaces
_UndefinedNameSpace	Object is not yet initialized.

13.15.2.10 spinNodeType

```
enum spinNodeType
```

Enumerator

ValueNode	
BaseNode	
IntegerNode	
BooleanNode	
FloatNode	
CommandNode	
StringNode	

Enumerator

RegisterNode	
EnumerationNode	
EnumEntryNode	
CategoryNode	
PortNode	
UnknownNode	

13.15.2.11 spinRepresentation

```
enum spinRepresentation
```

recommended representation of a node value

Enumerator

Linear	Slider with linear behavior.
Logarithmic	Slider with logarithmic behaviour.
Boolean	Check box.
PureNumber	Decimal number in an edit control.
HexNumber	Hex number in an edit control.
IPV4Address	IP-Address.
MACAddress	MAC-Address.
_UndefinedRepresentation	

13.15.2.12 spinSign

```
enum spinSign
```

Enumerator

Signed	
Unsigned	
_UndefinedSign	

13.15.2.13 spinSlope

```
enum spinSlope
```

typedef for fomula type

Enumerator

Increasing	
Decreasing	strictly monotonous increasing
Varying	strictly monotonous decreasing
Automatic	slope changes, e.g. at run-time
_UndefinedESlope	slope is determined automatically by probing the function Object is not yet initialized

13.15.2.14 spinStandardNameSpace

```
enum spinStandardNameSpace
```

Defines from which standard namespace a node name comes from.

Enumerator

None	name resides in custom namespace
GEV	name resides in GigE Vision namespace
IIDC	name resides in 1394 IIDC namespace
CL	name resides in camera link namespace
USB	name resides in USB namespace
_UndefinedStandardNameSpace	Object is not yet initialized.

13.15.2.15 spinVisibility

```
enum spinVisibility
```

Enumerator

Beginner	
Expert	
Guru	
Invisible	
_UndefinedVisibility	

13.15.2.16 spinXMLValidation

```
enum spinXMLValidation
```

typedef describing the different validity checks which can be performed on an XML file

The enum values for a bitfield of lenght uint32_t

Enumerator

xvLoad	
xvCycles	Creates a dummy node map
xvSFNC	checks for write and dependency cycles (implies xvLoad)
xvDefault	checks for conformance with the standard feature naming convention (SFNC)
xvAll	checks performed if nothing else is said
_UndefinedEXMLValidation	all possible checks Object is not yet initialized

13.15.2.17 spinYesNo

```
enum spinYesNo
```

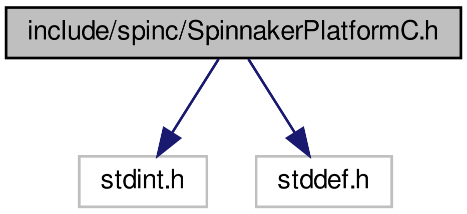
Defines the chices of a Yes/No alternaitve.

Enumerator

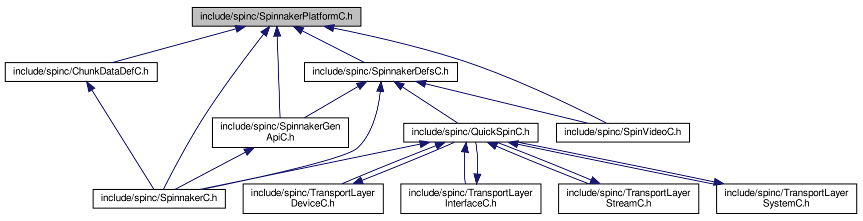
Yes	yes
No	no
_UndefinedYesNo	Object is not yet initialized.

13.16 include/spinc/SpinnakerPlatformC.h File Reference

Include dependency graph for SpinnakerPlatformC.h:



This graph shows which files directly or indirectly include this file:


include/spinc/SpinnakerPlatformC.h
include/spinc/ChunkDataDefC.h
include/spinc/SpinnakerDefsC.h
include/spinc/SpinnakerGen ApiC.h
include/spinc/QuickSpinC.h
include/spinc/SpinVideoC.h
include/spinc/SpinnakerC.h
include/spinc/TransportLayer DeviceC.h
include/spinc/TransportLayer InterfaceC.h
include/spinc/TransportLayer StreamC.h
include/spinc/TransportLayer SystemC.h

Macros

- #define SPINNAKERC_API SPINC_IMPORT_EXPORT spinError SPINC_CALLTYPE

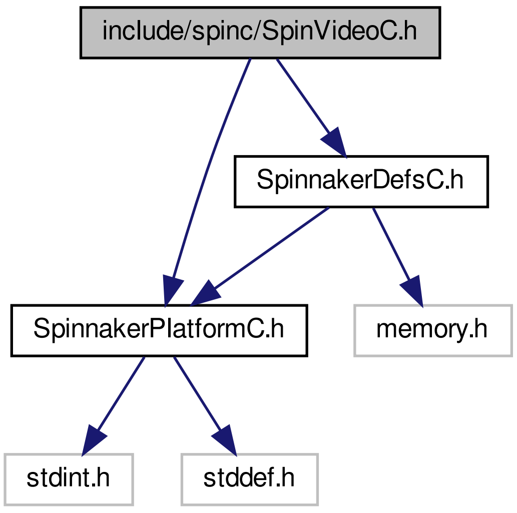
13.16.1 Macro Definition Documentation

13.16.1.1 SPINNAKERC_API

```
#define SPINNAKERC_API SPINC_IMPORT_EXPORT spinError SPINC_CALLTYPE
```

13.17 include/spinc/SpinVideoC.h File Reference

Include dependency graph for SpinVideoC.h:



Functions

- SPINNAKERC_API spinVideoOpenUncompressed (spinVideo ∗phSpinVideo, const char ∗pName, spinAVIOption option)
- SPINNAKERC_API spinVideoOpenMJPG (spinVideo ∗phSpinVideo, const char ∗pName, spinMJPGOption option)
- SPINNAKERC_API spinVideoOpenH264 (spinVideo ∗phSpinVideo, const char ∗pName, spinH264Option option)
- SPINNAKERC_API spinVideoAppend (spinVideo hSpinVideo, spinImage hImage)
- SPINNAKERC_API spinVideoSetMaximumFileSize (spinVideo hSpinVideo, unsigned int size)

  Set the maximum file size (in megabytes) of a AVI/MP4 file.
- SPINNAKERC_API spinVideoClose (spinVideo hSpinVideo)

13.17.1 Function Documentation

13.17.1.1 spinVideoAppend()

```
SPINNAKERC_API spinVideoAppend (
            spinVideo hSpinVideo,
            spinImage hImage )
```

13.17.1.2 spinVideoClose()

```
SPINNAKERC_API spinVideoClose (
            spinVideo hSpinVideo )
```

13.17.1.3 spinVideoOpenH264()

```
SPINNAKERC_API spinVideoOpenH264 (
            spinVideo * phSpinVideo,
            const char * pName,
            spinH264Option option )
```

13.17.1.4 spinVideoOpenMJPG()

```
SPINNAKERC_API spinVideoOpenMJPG (
            spinVideo * phSpinVideo,
            const char * pName,
            spinMJPGOption option )
```

13.17.1.5 spinVideoOpenUncompressed()

```
SPINNAKERC_API spinVideoOpenUncompressed (
            spinVideo * phSpinVideo,
            const char * pName,
            spinAVIOption option )
```

13.17.1.6 spinVideoSetMaximumFileSize()

```
SPINNAKERC_API spinVideoSetMaximumFileSize (
            spinVideo hSpinVideo,
            unsigned int size )
```

Set the maximum file size (in megabytes) of a AVI/MP4 file.

A new AVI/MP4 file is created automatically when file size limit is reached. Setting a maximum size of 0 indicates no limit on file size.

Parameters

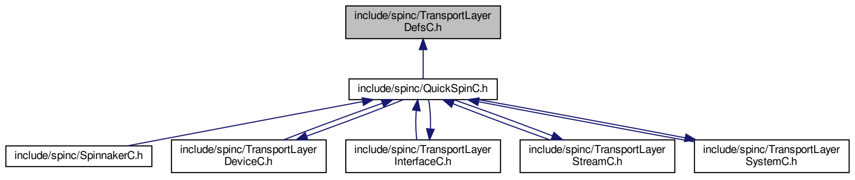
hSpinVideo	The spin video recorder to append the image to
size	The maximum video file size in MB.

Returns

spinError The error code; returns SPINNAKER_ERR_SUCCESS (or 0) for no error

13.18 include/spinc/TransportLayerDefsC.h File Reference

This graph shows which files directly or indirectly include this file:


include/spinc/TransportLayer DefsC.h
include/spinc/QuickSpinC.h
include/spinc/SpinnakerC.h
include/spinc/TransportLayer DeviceC.h
include/spinc/TransportLayer InterfaceC.h
include/spinc/TransportLayer StreamC.h
include/spinc/TransportLayer SystemC.h

Enumerations

- enum spinTLStreamTypeEnums {
  StreamType_GigEVision ,
  StreamType_CameraLink ,
  StreamType_CameraLinkHS ,
  StreamType_CoaXPress ,
  StreamType_USB3Vision ,
  StreamType_Custom ,
  NUMSTREAMTYPE }

  The enumeration definitions for transport layer nodes.
- enum spinTLStreamModeEnums {
  StreamMode_Socket ,
  StreamMode_LWF ,
  StreamMode_MVA ,
  NUMSTREAMMODE }
- enum spinTLStreamBufferCountModeEnums {
  StreamBufferCountMode_Manual ,
  NUMSTREAMBUFFERCOUNTMODE }
- enum spinTLStreamBufferHandlingModeEnums {
  StreamBufferHandlingMode_OldestFirst ,
  StreamBufferHandlingMode_OldestFirstOverwrite ,
  StreamBufferHandlingMode_NewestOnly ,
  StreamBufferHandlingMode_NewestFirst ,
  NUMSTREAMBUFFERHANDLINGMODE }
- enum spinTLDeviceTypeEnums {
  DeviceType_GigEVision ,
  DeviceType_CameraLink ,
  DeviceType_CameraLinkHS ,
  DeviceType_CoaXPress ,
  DeviceType_USB3Vision ,
  DeviceType_Custom ,
  NUMDEVICETYPE }

- enum spinTLDeviceAccessStatusEnums {
  DeviceAccessStatus_Unknown ,
  DeviceAccessStatus_ReadWrite ,
  DeviceAccessStatus_ReadOnly ,
  DeviceAccessStatus_NoAccess ,
  DeviceAccessStatus_Busy ,
  DeviceAccessStatus_OpenReadWrite ,
  DeviceAccessStatus_OpenReadOnly ,
  NUMDEVICEACCESSSTATUS }
- enum spinTLGenICamXMLLocationEnums {
  GenICamXMLLocation_Device ,
  GenICamXMLLocation_Host ,
  NUMGENICAMXMLLOCATION }
- enum spinTLGUIXMLLocationEnums {
  GUIXMLLocation_Device ,
  GUIXMLLocation_Host ,
  NUMGUIXMLLOCATION }
- enum spinTLGevCCPEnums {
  GevCCP_EnumEntry_GevCCP_OpenAccess ,
  GevCCP_EnumEntry_GevCCP_ExclusiveAccess ,
  GevCCP_EnumEntry_GevCCP_ControlAccess ,
  NUMGEVCCP }
- enum spinTLDeviceEndianessMechanismEnums {
  DeviceEndianessMechanism_Legacy ,
  DeviceEndianessMechanism_Standard ,
  NUMDEVICEENDIANESSMECHANISM }
- enum spinTLDeviceCurrentSpeedEnums {
  DeviceCurrentSpeed_UnknownSpeed ,
  DeviceCurrentSpeed_LowSpeed ,
  DeviceCurrentSpeed_FullSpeed ,
  DeviceCurrentSpeed_HighSpeed ,
  DeviceCurrentSpeed_SuperSpeed ,
  NUMDEVICECURRENTSPEED }
- enum spinTLInterfaceTypeEnums {
  InterfaceType_GigEVision ,
  InterfaceType_CameraLink ,
  InterfaceType_CameraLinkHS ,
  InterfaceType_CoaXPress ,
  InterfaceType_USB3Vision ,
  InterfaceType_Custom ,
  NUMINTERFACETYPE }
- enum spinTLPOEStatusEnums {
  POEStatus_NotSupported ,
  POEStatus_PowerOff ,
  POEStatus_PowerOn ,
  NUMPOESTATUS }
- enum spinTLFilterDriverStatusEnums {
  FilterDriverStatus_NotSupported ,
  FilterDriverStatus_Disabled ,
  FilterDriverStatus_Enabled ,
  NUMFILTERDRIVERSTATUS }
- enum spinTLTLTypeEnums {
  TLType_GigEVision ,
  TLType_CameraLink ,
  TLType_CameraLinkHS ,
  TLType_CoaXPress ,
  TLType_USB3Vision ,
  TLType_Mixed ,

TLType_Custom ,
NUMTLTYPE }

13.18.1 Enumeration Type Documentation

13.18.1.1 spinTLDeviceAccessStatusEnums

```
enum spinTLDeviceAccessStatusEnums
```

< Gets the access status the transport layer Producer has on the device.

Enumerator

DeviceAccessStatus_Unknown	Not known to producer.
DeviceAccessStatus_ReadWrite	Full access
DeviceAccessStatus_ReadOnly	Read-only access
DeviceAccessStatus_NoAccess	Not available to connect
DeviceAccessStatus_Busy	The device is already opened by another entity
DeviceAccessStatus_OpenReadWrite	Open in Read/Write mode by this GenTL host
DeviceAccessStatus_OpenReadOnly	Open in Read access mode by this GenTL host
NUMDEVICEACCESSSTATUS	

13.18.1.2 spinTLDeviceCurrentSpeedEnums

```
enum spinTLDeviceCurrentSpeedEnums
```

< The USB Speed that the device is currently operating at.

Enumerator

DeviceCurrentSpeed_UnknownSpeed	Unknown-Speed.
DeviceCurrentSpeed_LowSpeed	Low-Speed.
DeviceCurrentSpeed_FullSpeed	Full-Speed.
DeviceCurrentSpeed_HighSpeed	High-Speed.
DeviceCurrentSpeed_SuperSpeed	Super-Speed.
NUMDEVICECURRENTSPEED	

13.18.1.3 spinTLDeviceEndianessMechanismEnums

```
enum spinTLDeviceEndianessMechanismEnums
```

< Identifies the endianness handling mode.

Enumerator

DeviceEndianessMechanism_Legacy	Handling the device endianness according to GenICam Schema 1.0
DeviceEndianessMechanism_Standard	Handling the device endianness according to GenICam Schema 1.1 and later
NUMDEVICEENDIANESSMECHANISM	

13.18.1.4 spinTLDeviceTypeEnums

```
enum spinTLDeviceTypeEnums
```

< Transport layer type of the device.

Enumerator

DeviceType_GigEVision	GigE Vision
DeviceType_CameraLink	Camera Link
DeviceType_CameraLinkHS	Camera Link High Speed
DeviceType_CoaXPress	CoaXPress
DeviceType_USB3Vision	USB3 Vision
DeviceType_Custom	Custom transport layer
NUMDEVICETYPE	

13.18.1.5 spinTLFilterDriverStatusEnums

```
enum spinTLFilterDriverStatusEnums
```

< Reports whether FLIR Light Weight Filter Driver is enabled, disabled, or not installed.

Enumerator

FilterDriverStatus_NotSupported	Not Installed
FilterDriverStatus_Disabled	FLIR Light Weight Filter Driver is disabled across all interfaces
FilterDriverStatus_Enabled	FLIR Light Weight Filter Driver is enabled
NUMFILTERDRIVERSTATUS	

13.18.1.6 spinTLGenICamXMLLocationEnums

```
enum spinTLGenICamXMLLocationEnums
```

< Sets the location to load GenICam XML.

Enumerator

GenICamXMLLocation_Device	Load GenICam XML from device
GenICamXMLLocation_Host	Load GenICam XML from host
NUMGENICAMXMLLOCATION	

13.18.1.7 spinTLGevCCPEnums

```
enum spinTLGevCCPEnums
```

< Controls the device access privilege of an application.

Enumerator

GevCCP_EnumEntry_GevCCP_OpenAccess	Open access privilege.
GevCCP_EnumEntry_GevCCP_ExclusiveAccess	Exclusive access privilege.
GevCCP_EnumEntry_GevCCP_ControlAccess	Control access privilege.
NUMGEVCCP	

13.18.1.8 spinTLGUIXMLLocationEnums

```
enum spinTLGUIXMLLocationEnums
```

< Sets the location to load GUI XML.

Enumerator

GUIXMLLocation_Device	Load XML from device
GUIXMLLocation_Host	Load XML from host
NUMGUIXMLLOCATION	

13.18.1.9 spinTLInterfaceTypeEnums

```
enum spinTLInterfaceTypeEnums
```

< Transport layer type of the interface.

Enumerator

InterfaceType_GigEVision	GigE Vision
InterfaceType_CameraLink	Camera Link

Enumerator

InterfaceType_CameraLinkHS	Camera Link High Speed
InterfaceType_CoaXPress	CoaXPress
InterfaceType_USB3Vision	USB3 Vision
InterfaceType_Custom	Custom transport layer
NUMINTERFACETYPE	

13.18.1.10 spinTLPOEStatusEnums

```
enum spinTLPOEStatusEnums
```

< Reports and controls the interface's power over Ethernet status.

Enumerator

POEStatus_NotSupported	Not Supported
POEStatus_PowerOff	Power is Off
POEStatus_PowerOn	Power is On
NUMPOESTATUS	

13.18.1.11 spinTLStreamBufferCountModeEnums

```
enum spinTLStreamBufferCountModeEnums
```

< Controls access to setting the number of buffers used for the stream.

Enumerator

StreamBufferCountMode_Manual	The number of buffers used for the stream is set by the user.
NUMSTREAMBUFFERCOUNTMODE	

13.18.1.12 spinTLStreamBufferHandlingModeEnums

```
enum spinTLStreamBufferHandlingModeEnums
```

< Available buffer handling modes of this data stream:

Enumerator

StreamBufferHandlingMode_OldestFirst	The application always gets the buffer from the head of the output buffer queue (thus, the oldest available one). If the output buffer queue is empty, the application waits for a newly acquired buffer until the timeout expires.
StreamBufferHandlingMode_OldestFirstOverwrite	The application always gets the buffer from the head of the output buffer queue (thus, the oldest available one). If the output buffer queue is empty, the application waits for a newly acquired buffer until the timeout expires. If a new buffer arrives it will overwrite the existing buffer from the head of the queue (behaves like a circular buffer).
StreamBufferHandlingMode_NewestOnly	The application always gets the latest completed buffer (the newest one). If the Output Buffer Queue is empty, the application waits for a newly acquired buffer until the timeout expires. This buffer handling mode is typically used in a live display GUI where it is important that there is no lag between camera and display.
StreamBufferHandlingMode_NewestFirst	The application always gets the buffer from the tail of the output buffer queue (thus, the newest available one). If the output buffer queue is empty, the application waits for a newly acquired buffer until the timeout expires.
NUMSTREAMBUFFERHANDLINGMODE	

13.18.1.13 spinTLStreamModeEnums

```
enum spinTLStreamModeEnums
```

< Stream mode of the device.

Enumerator

StreamMode_Socket	Socket
StreamMode_LWF	Light Weight Filter Driver
StreamMode_MVA	Machine Vision Accelerator Driver
NUMSTREAMMODE	

13.18.1.14 spinTLStreamTypeEnums

```
enum spinTLStreamTypeEnums
```

The enumeration definitions for transport layer nodes.

< Stream type of the device.

Enumerator

StreamType_GigEVision	GigE Vision
StreamType_CameraLink	Camera Link
StreamType_CameraLinkHS	Camera Link High Speed
StreamType_CoaXPress	CoaXPress
StreamType_USB3Vision	USB3 Vision
StreamType_Custom	Custom transport layer
NUMSTREAMTYPE	

13.18.1.15 spinTLTLTypeEnums

```
enum spinTLTLTypeEnums
```

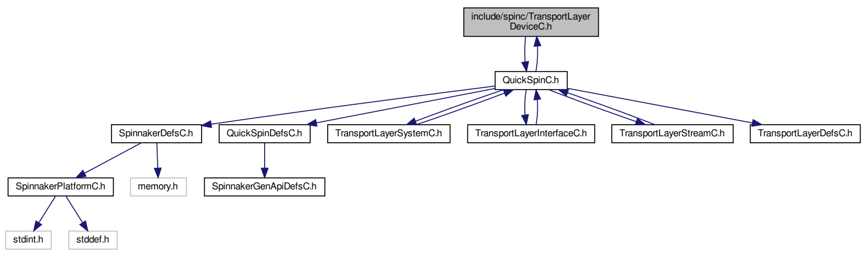
< Transport layer type of the GenTL Producer implementation.

Enumerator

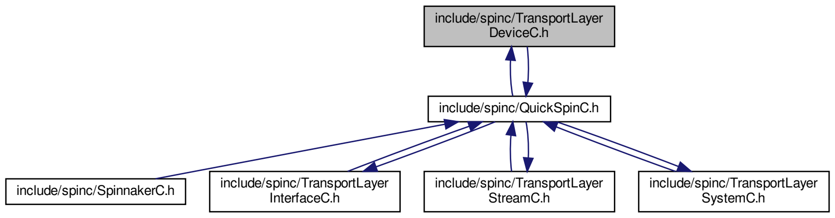
TLType_GigEVision	GigE Vision
TLType_CameraLink	Camera Link
TLType_CameraLinkHS	Camera Link High Speed
TLType_CoaXPress	CoaXPress
TLType_USB3Vision	USB3 Vision
TLType_Mixed	Different Interface modules of the GenTL Producer are of different types
TLType_Custom	Custom transport layer
NUMTLTYPE	

13.19 include/spinc/TransportLayerDeviceC.h File Reference

Include dependency graph for TransportLayerDeviceC.h:



This graph shows which files directly or indirectly include this file:

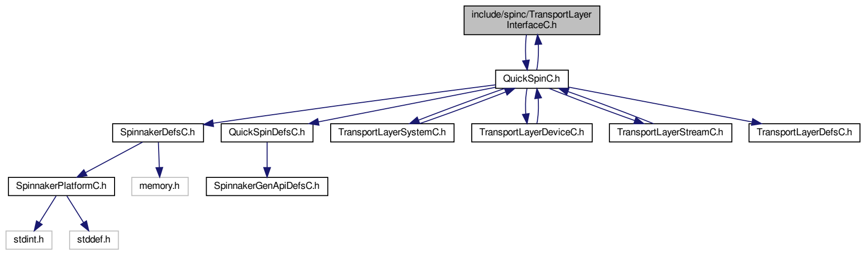

include/spinc/TransportLayer DeviceC.h
include/spinc/QuickSpinC.h
include/spinc/SpinnakerC.h
include/spinc/TransportLayer InterfaceC.h
include/spinc/TransportLayer StreamC.h
include/spinc/TransportLayer SystemC.h

Data Structures

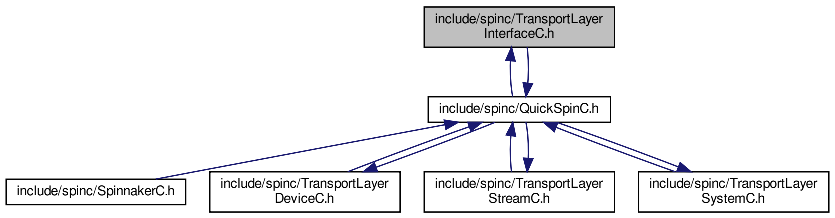
- struct quickSpinTLDevice

13.20 include/spinc/TransportLayerInterfaceC.h File Reference

Include dependency graph for TransportLayerInterfaceC.h:



This graph shows which files directly or indirectly include this file:

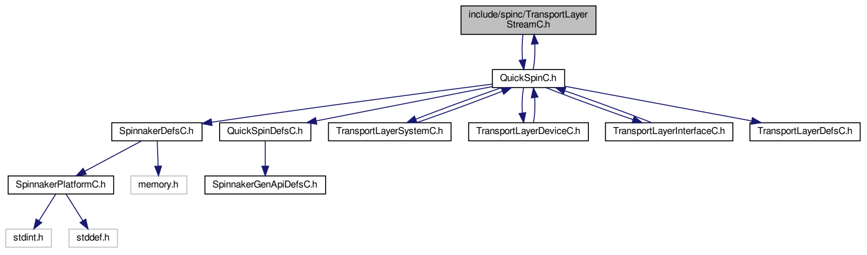

include/spinc/TransportLayer InterfaceC.h
include/spinc/QuickSpinC.h
include/spinc/SpinnakerC.h
include/spinc/TransportLayer DeviceC.h
include/spinc/TransportLayer StreamC.h
include/spinc/TransportLayer SystemC.h

Data Structures

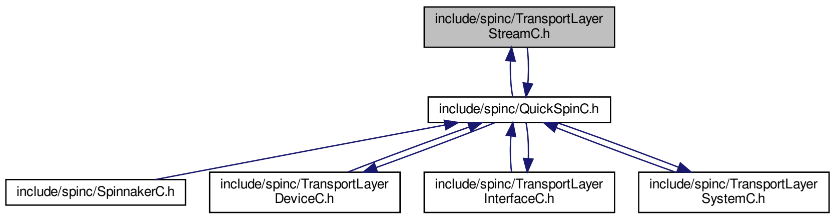
- struct quickSpinTLInterface

13.21 include/spinc/TransportLayerStreamC.h File Reference

Include dependency graph for TransportLayerStreamC.h:



This graph shows which files directly or indirectly include this file:

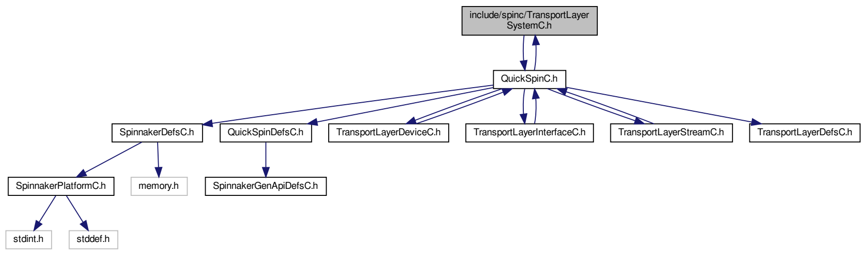

include/spinc/TransportLayer StreamC.h
include/spinc/QuickSpinC.h
include/spinc/SpinnakerC.h
include/spinc/TransportLayer DeviceC.h
include/spinc/TransportLayer InterfaceC.h
include/spinc/TransportLayer SystemC.h

Data Structures

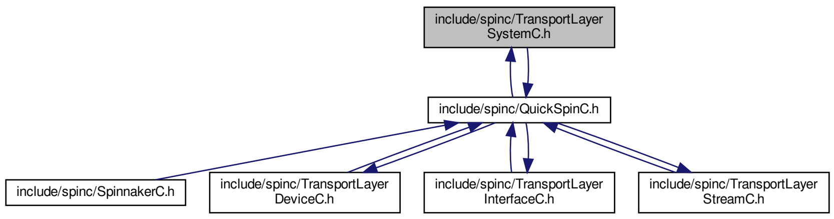
- struct quickSpinTLStream

13.22 include/spinc/TransportLayerSystemC.h File Reference

Include dependency graph for TransportLayerSystemC.h:



This graph shows which files directly or indirectly include this file:


include/spinc/TransportLayer SystemC.h
include/spinc/QuickSpinC.h
include/spinc/SpinnakerC.h
include/spinc/TransportLayer DeviceC.h
include/spinc/TransportLayer InterfaceC.h
include/spinc/TransportLayer StreamC.h

Data Structures

- struct quickSpinTLSystem

Index