

Spinnaker C

2.4.0.143

Generated by Doxygen 1.8.13

Contents

1	Getting Started	1
2	Programmer's Guide	3
3	Benefits of Spinnaker	5
4	FlyCapture2 Feature Comparison with Spinnaker	7
5	Working with GenICam GenTL Devices	9
5.1	GenTL Overview	9
5.2	Installation	9
5.3	Troubleshooting	10
5.3.1	Enable FLIR GenTL Logging	10
5.3.2	USB3 Device Image Tearing	10
6	Software Licensing Information	11
7	Software Maintenance Policy	13
7.1	GenTL Overview	13
7.2	Platform Support Policy	13
7.2.1	Windows Support	13
7.2.2	Linux Desktop Support	13
7.2.3	Linux Embedded Support	13
7.2.4	MacOS Support	14
7.3	Versioning Policy	14

8	Module Index	15
8.1	Modules	15
9	Data Structure Index	17
9.1	Data Structures	17
10	File Index	19
10.1	File List	19
11	Module Documentation	21
11.1	Spinnaker C Definitions	21
11.1.1	Detailed Description	22
11.1.2	Typedef Documentation	22
11.1.2.1	bool8_t	22
11.1.3	Variable Documentation	22
11.1.3.1	False	22
11.1.3.2	True	22
11.2	Camera Enumerations	23
11.2.1	Detailed Description	55
11.2.2	Enumeration Type Documentation	55
11.2.2.1	spinAcquisitionModeEnums	55
11.2.2.2	spinAcquisitionStatusSelectorEnums	55
11.2.2.3	spinActionUnconditionalModeEnums	56
11.2.2.4	spinAdcBitDepthEnums	56
11.2.2.5	spinAutoAlgorithmSelectorEnums	57
11.2.2.6	spinAutoExposureControlPriorityEnums	57
11.2.2.7	spinAutoExposureLightingModeEnums	57
11.2.2.8	spinAutoExposureMeteringModeEnums	58
11.2.2.9	spinAutoExposureTargetGreyValueAutoEnums	58
11.2.2.10	spinBalanceRatioSelectorEnums	58
11.2.2.11	spinBalanceWhiteAutoEnums	59
11.2.2.12	spinBalanceWhiteAutoProfileEnums	59

11.2.2.13 spinBinningHorizontalModeEnums	59
11.2.2.14 spinBinningSelectorEnums	60
11.2.2.15 spinBinningVerticalModeEnums	60
11.2.2.16 spinBlackLevelAutoBalanceEnums	60
11.2.2.17 spinBlackLevelAutoEnums	61
11.2.2.18 spinBlackLevelSelectorEnums	61
11.2.2.19 spinChunkBlackLevelSelectorEnums	61
11.2.2.20 spinChunkCounterSelectorEnums	62
11.2.2.21 spinChunkEncoderSelectorEnums	62
11.2.2.22 spinChunkEncoderStatusEnums	62
11.2.2.23 spinChunkExposureTimeSelectorEnums	63
11.2.2.24 spinChunkGainSelectorEnums	63
11.2.2.25 spinChunkImageComponentEnums	63
11.2.2.26 spinChunkPixelFormatEnums	64
11.2.2.27 spinChunkRegionIDEnums	64
11.2.2.28 spinChunkScan3dCoordinateReferenceSelectorEnums	65
11.2.2.29 spinChunkScan3dCoordinateSelectorEnums	65
11.2.2.30 spinChunkScan3dCoordinateSystemEnums	65
11.2.2.31 spinChunkScan3dCoordinateSystemReferenceEnums	66
11.2.2.32 spinChunkScan3dCoordinateTransformSelectorEnums	66
11.2.2.33 spinChunkScan3dDistanceUnitEnums	66
11.2.2.34 spinChunkScan3dOutputModeEnums	67
11.2.2.35 spinChunkSelectorEnums	68
11.2.2.36 spinChunkSourceIDEnums	68
11.2.2.37 spinChunkTimerSelectorEnums	68
11.2.2.38 spinChunkTransferStreamIDEnums	69
11.2.2.39 spinCIConfigurationEnums	69
11.2.2.40 spinCITimeSlotsCountEnums	70
11.2.2.41 spinColorTransformationSelectorEnums	70
11.2.2.42 spinColorTransformationValueSelectorEnums	70

11.2.2.43 spinCompressionSaturationPriorityEnums	71
11.2.2.44 spinCounterEventActivationEnums	71
11.2.2.45 spinCounterEventSourceEnums	71
11.2.2.46 spinCounterResetActivationEnums	72
11.2.2.47 spinCounterResetSourceEnums	72
11.2.2.48 spinCounterSelectorEnums	73
11.2.2.49 spinCounterStatusEnums	73
11.2.2.50 spinCounterTriggerActivationEnums	74
11.2.2.51 spinCounterTriggerSourceEnums	74
11.2.2.52 spinCxpConnectionTestModeEnums	75
11.2.2.53 spinCxpLinkConfigurationEnums	75
11.2.2.54 spinCxpLinkConfigurationPreferredEnums	76
11.2.2.55 spinCxpLinkConfigurationStatusEnums	77
11.2.2.56 spinCxpPoCxpStatusEnums	78
11.2.2.57 spinDecimationHorizontalModeEnums	78
11.2.2.58 spinDecimationSelectorEnums	78
11.2.2.59 spinDecimationVerticalModeEnums	79
11.2.2.60 spinDefectCorrectionModeEnums	79
11.2.2.61 spinDeinterlacingEnums	79
11.2.2.62 spinDeviceCharacterSetEnums	80
11.2.2.63 spinDeviceClockSelectorEnums	80
11.2.2.64 spinDeviceConnectionStatusEnums	80
11.2.2.65 spinDeviceIndicatorModeEnums	81
11.2.2.66 spinDeviceLinkHeartbeatModeEnums	81
11.2.2.67 spinDeviceLinkThroughputLimitModeEnums	81
11.2.2.68 spinDevicePowerSupplySelectorEnums	82
11.2.2.69 spinDeviceRegistersEndiannessEnums	82
11.2.2.70 spinDeviceScanTypeEnums	82
11.2.2.71 spinDeviceSerialPortBaudRateEnums	82
11.2.2.72 spinDeviceSerialPortSelectorEnums	83

11.2.2.73 spinDeviceStreamChannelEndiannessEnums	83
11.2.2.74 spinDeviceStreamChannelTypeEnums	83
11.2.2.75 spinDeviceTapGeometryEnums	84
11.2.2.76 spinDeviceTemperatureSelectorEnums	85
11.2.2.77 spinDeviceTLTypeEnums	85
11.2.2.78 spinDeviceTypeEnums	86
11.2.2.79 spinEncoderModeEnums	86
11.2.2.80 spinEncoderOutputModeEnums	86
11.2.2.81 spinEncoderResetActivationEnums	87
11.2.2.82 spinEncoderResetSourceEnums	87
11.2.2.83 spinEncoderSelectorEnums	88
11.2.2.84 spinEncoderSourceAEnums	89
11.2.2.85 spinEncoderSourceBEnums	89
11.2.2.86 spinEncoderStatusEnums	89
11.2.2.87 spinEventNotificationEnums	90
11.2.2.88 spinEventSelectorEnums	90
11.2.2.89 spinExposureActiveModeEnums	90
11.2.2.90 spinExposureAutoEnums	91
11.2.2.91 spinExposureModeEnums	91
11.2.2.92 spinExposureTimeModeEnums	91
11.2.2.93 spinExposureTimeSelectorEnums	92
11.2.2.94 spinFileOpenModeEnums	92
11.2.2.95 spinFileOperationSelectorEnums	92
11.2.2.96 spinFileOperationStatusEnums	93
11.2.2.97 spinFileSelectorEnums	93
11.2.2.98 spinGainAutoBalanceEnums	93
11.2.2.99 spinGainAutoEnums	95
11.2.2.100 spinGainSelectorEnums	95
11.2.2.101 spinGevCCPEnums	95
11.2.2.102 spinGevCurrentPhysicalLinkConfigurationEnums	96

11.2.2.103spinGevGVCPExtendedStatusCodesSelectorEnums	96
11.2.2.104spinGevGVSPExtendedIDModeEnums	96
11.2.2.105spinGevIEEE1588ClockAccuracyEnums	97
11.2.2.106spinGevIEEE1588ModeEnums	97
11.2.2.107spinGevIEEE1588StatusEnums	97
11.2.2.108spinGevIPConfigurationStatusEnums	98
11.2.2.109spinGevPhysicalLinkConfigurationEnums	98
11.2.2.110spinGevSupportedOptionSelectorEnums	99
11.2.2.111spinImageComponentSelectorEnums	99
11.2.2.112spinImageCompressionJPEGFormatOptionEnums	100
11.2.2.113spinImageCompressionModeEnums	100
11.2.2.114spinImageCompressionRateOptionEnums	101
11.2.2.115spinLineFormatEnums	101
11.2.2.116spinLineInputFilterSelectorEnums	101
11.2.2.117spinLineModeEnums	102
11.2.2.118spinLineSelectorEnums	102
11.2.2.119spinLineSourceEnums	102
11.2.2.120spinLogicBlockLUTInputActivationEnums	103
11.2.2.121spinLogicBlockLUTInputSelectorEnums	103
11.2.2.122spinLogicBlockLUTInputSourceEnums	104
11.2.2.123spinLogicBlockLUTSelectorEnums	104
11.2.2.124spinLogicBlockSelectorEnums	105
11.2.2.125spinLUTSelectorEnums	105
11.2.2.126spinPixelColorFilterEnums	105
11.2.2.127spinPixelFormatEnums	106
11.2.2.128spinPixelFormatInfoSelectorEnums	111
11.2.2.129spinPixelSizeEnums	117
11.2.2.130spinRegionDestinationEnums	118
11.2.2.131spinRegionModeEnums	118
11.2.2.132spinRegionSelectorEnums	118

11.2.2.133spinRgbTransformLightSourceEnums	119
11.2.2.134spinScan3dCoordinateReferenceSelectorEnums	119
11.2.2.135spinScan3dCoordinateSelectorEnums	120
11.2.2.136spinScan3dCoordinateSystemEnums	120
11.2.2.137spinScan3dCoordinateSystemReferenceEnums	120
11.2.2.138spinScan3dCoordinateTransformSelectorEnums	121
11.2.2.139spinScan3dDistanceUnitEnums	121
11.2.2.140spinScan3dOutputModeEnums	121
11.2.2.141spinSensorDigitizationTapsEnums	122
11.2.2.142spinSensorShutterModeEnums	123
11.2.2.143spinSensorTapsEnums	123
11.2.2.144spinSequencerConfigurationModeEnums	123
11.2.2.145spinSequencerConfigurationValidEnums	124
11.2.2.146spinSequencerModeEnums	124
11.2.2.147spinSequencerSetValidEnums	124
11.2.2.148spinSequencerTriggerActivationEnums	124
11.2.2.149spinSequencerTriggerSourceEnums	125
11.2.2.150spinSerialPortBaudRateEnums	125
11.2.2.151spinSerialPortParityEnums	126
11.2.2.152spinSerialPortSelectorEnums	126
11.2.2.153spinSerialPortSourceEnums	126
11.2.2.154spinSerialPortStopBitsEnums	127
11.2.2.155spinSoftwareSignalSelectorEnums	127
11.2.2.156spinSourceSelectorEnums	127
11.2.2.157spinTestPatternEnums	128
11.2.2.158spinTestPatternGeneratorSelectorEnums	128
11.2.2.159spinTimerSelectorEnums	128
11.2.2.160spinTimerStatusEnums	129
11.2.2.161spinTimerTriggerActivationEnums	129
11.2.2.162spinTimerTriggerSourceEnums	129

11.2.2.163	spinTransferComponentSelectorEnums	131
11.2.2.164	spinTransferControlModeEnums	131
11.2.2.165	spinTransferOperationModeEnums	131
11.2.2.166	spinTransferQueueModeEnums	132
11.2.2.167	spinTransferSelectorEnums	132
11.2.2.168	spinTransferStatusSelectorEnums	132
11.2.2.169	spinTransferTriggerActivationEnums	133
11.2.2.170	spinTransferTriggerModeEnums	133
11.2.2.171	spinTransferTriggerSelectorEnums	134
11.2.2.172	spinTransferTriggerSourceEnums	134
11.2.2.173	spinTriggerActivationEnums	135
11.2.2.174	spinTriggerModeEnums	135
11.2.2.175	spinTriggerOverlapEnums	136
11.2.2.176	spinTriggerSelectorEnums	136
11.2.2.177	spinTriggerSourceEnums	136
11.2.2.178	spinUserOutputSelectorEnums	137
11.2.2.179	spinUserSetDefaultEnums	137
11.2.2.180	spinUserSetSelectorEnums	138
11.2.2.181	spinWhiteClipSelectorEnums	138
11.3	Chunk Data Structures	139
11.3.1	Detailed Description	139
11.4	Spinnaker C QuickSpin API	140
11.4.1	Detailed Description	140
11.5	QuickSpin Access	141
11.5.1	Detailed Description	141
11.5.2	Function Documentation	141
11.5.2.1	quickSpinInit()	141
11.5.2.2	quickSpinInitEx()	142
11.5.2.3	quickSpinTLDeviceInit()	142
11.5.2.4	quickSpinTLInterfaceInit()	142

11.5.2.5	quickSpinTLStreamInit()	142
11.5.2.6	quickSpinTLSystemInit()	142
11.6	Spinnaker C API	143
11.6.1	Detailed Description	144
11.6.2	Function Documentation	144
11.6.2.1	spinCameraDiscoverMaxPacketSize()	144
11.7	Error Handling	145
11.7.1	Detailed Description	145
11.7.2	Function Documentation	145
11.7.2.1	spinErrorGetLast()	145
11.7.2.2	spinErrorGetLastBuildDate()	146
11.7.2.3	spinErrorGetLastBuildTime()	146
11.7.2.4	spinErrorGetLastFileName()	147
11.7.2.5	spinErrorGetLastFullMessage()	147
11.7.2.6	spinErrorGetLastFunctionName()	148
11.7.2.7	spinErrorGetLastLineNumber()	148
11.7.2.8	spinErrorGetLastMessage()	149
11.8	System Access	150
11.8.1	Detailed Description	151
11.8.2	Function Documentation	151
11.8.2.1	spinSystemGetCameras()	151
11.8.2.2	spinSystemGetCamerasEx()	152
11.8.2.3	spinSystemGetInstance()	152
11.8.2.4	spinSystemGetInterfaces()	154
11.8.2.5	spinSystemGetLibraryVersion()	154
11.8.2.6	spinSystemGetLoggingLevel()	155
11.8.2.7	spinSystemGetTLNodeMap()	155
11.8.2.8	spinSystemIsInUse()	156
11.8.2.9	spinSystemRegisterDeviceArrivalEventHandler()	156
11.8.2.10	spinSystemRegisterDeviceRemovalEventHandler()	157

11.8.2.11 spinSystemRegisterInterfaceEventHandler()	157
11.8.2.12 spinSystemRegisterLogEventHandler()	158
11.8.2.13 spinSystemReleaseInstance()	158
11.8.2.14 spinSystemSendActionCommand()	158
11.8.2.15 spinSystemSetLoggingLevel()	159
11.8.2.16 spinSystemUnregisterAllLogEventHandlers()	160
11.8.2.17 spinSystemUnregisterDeviceArrivalEventHandler()	160
11.8.2.18 spinSystemUnregisterDeviceRemovalEventHandler()	161
11.8.2.19 spinSystemUnregisterInterfaceEventHandler()	161
11.8.2.20 spinSystemUnregisterLogEventHandler()	162
11.8.2.21 spinSystemUpdateCameras()	162
11.8.2.22 spinSystemUpdateCamerasEx()	162
11.9 InterfaceList Access	164
11.9.1 Detailed Description	164
11.9.2 Function Documentation	164
11.9.2.1 spinInterfaceListClear()	164
11.9.2.2 spinInterfaceListCreateEmpty()	165
11.9.2.3 spinInterfaceListDestroy()	165
11.9.2.4 spinInterfaceListGet()	166
11.9.2.5 spinInterfaceListGetSize()	166
11.10 CameraList Access	168
11.10.1 Detailed Description	168
11.10.2 Function Documentation	168
11.10.2.1 spinCameraListAppend()	169
11.10.2.2 spinCameraListClear()	169
11.10.2.3 spinCameraListCreateEmpty()	169
11.10.2.4 spinCameraListDestroy()	170
11.10.2.5 spinCameraListGet()	170
11.10.2.6 spinCameraListGetBySerial()	171
11.10.2.7 spinCameraListGetSize()	171

11.10.2.8 spinCameraListRemove()	172
11.10.2.9 spinCameraListRemoveBySerial()	172
11.11 Interface Access	174
11.11.1 Detailed Description	175
11.11.2 Function Documentation	175
11.11.2.1 spinInterfaceGetCameras()	175
11.11.2.2 spinInterfaceGetCamerasEx()	175
11.11.2.3 spinInterfaceGetTLNodeMap()	176
11.11.2.4 spinInterfaceIsInUse()	176
11.11.2.5 spinInterfaceRegisterDeviceArrivalEventHandler()	177
11.11.2.6 spinInterfaceRegisterDeviceRemovalEventHandler()	177
11.11.2.7 spinInterfaceRegisterInterfaceEventHandler()	178
11.11.2.8 spinInterfaceRelease()	178
11.11.2.9 spinInterfaceSendActionCommand()	179
11.11.2.10 spinInterfaceUnregisterDeviceArrivalEventHandler()	179
11.11.2.11 spinInterfaceUnregisterDeviceRemovalEventHandler()	180
11.11.2.12 spinInterfaceUnregisterInterfaceEventHandler()	180
11.11.2.13 spinInterfaceUpdateCameras()	181
11.12 Camera Access	182
11.12.1 Detailed Description	183
11.12.2 Function Documentation	183
11.12.2.1 spinCameraBeginAcquisition()	183
11.12.2.2 spinCameraDeInit()	184
11.12.2.3 spinCameraEndAcquisition()	184
11.12.2.4 spinCameraGetAccessMode()	185
11.12.2.5 spinCameraGetGuiXml()	185
11.12.2.6 spinCameraGetNextImage()	186
11.12.2.7 spinCameraGetNextImageEx()	186
11.12.2.8 spinCameraGetNodeMap()	187
11.12.2.9 spinCameraGetTLDeviceNodeMap()	187

11.12.2.10	<code>spinCameraGetTLStreamNodeMap()</code>	188
11.12.2.11	<code>spinCameraGetUniqueID()</code>	188
11.12.2.12	<code>spinCameraInit()</code>	189
11.12.2.13	<code>spinCameralIsInitialized()</code>	189
11.12.2.14	<code>spinCameralIsStreaming()</code>	189
11.12.2.15	<code>spinCameralIsValid()</code>	190
11.12.2.16	<code>spinCameraReadPort()</code>	190
11.12.2.17	<code>spinCameraRegisterDeviceEventHandler()</code>	191
11.12.2.18	<code>spinCameraRegisterDeviceEventHandlerEx()</code>	191
11.12.2.19	<code>spinCameraRegisterImageEventHandler()</code>	192
11.12.2.20	<code>spinCameraRelease()</code>	192
11.12.2.21	<code>spinCameraUnregisterDeviceEventHandler()</code>	192
11.12.2.22	<code>spinCameraUnregisterImageEventHandler()</code>	193
11.12.2.23	<code>spinCameraWritePort()</code>	193
11.13	Image Access	194
11.13.1	Detailed Description	196
11.13.2	Function Documentation	196
11.13.2.1	<code>spinImageCalculateStatistics()</code>	196
11.13.2.2	<code>spinImageCheckCRC()</code>	197
11.13.2.3	<code>spinImageConvert()</code>	197
11.13.2.4	<code>spinImageConvertEx()</code>	198
11.13.2.5	<code>spinImageCreate()</code>	198
11.13.2.6	<code>spinImageCreateEmpty()</code>	199
11.13.2.7	<code>spinImageCreateEx()</code>	199
11.13.2.8	<code>spinImageCreateEx2()</code>	200
11.13.2.9	<code>spinImageDeepCopy()</code>	201
11.13.2.10	<code>spinImageDestroy()</code>	201
11.13.2.11	<code>spinImageGetBitsPerPixel()</code>	202
11.13.2.12	<code>spinImageGetBufferSize()</code>	202
11.13.2.13	<code>spinImageGetChunkLayoutID()</code>	203

11.13.2.14spinImageGetColorProcessing()	203
11.13.2.15spinImageGetData()	204
11.13.2.16spinImageGetDefaultColorProcessing()	204
11.13.2.17spinImageGetFrameID()	204
11.13.2.18spinImageGetHeight()	205
11.13.2.19spinImageGetID()	205
11.13.2.20spinImageGetNumDecompressionThreads()	206
11.13.2.21spinImageGetOffsetX()	206
11.13.2.22spinImageGetOffsetY()	207
11.13.2.23spinImageGetPaddingX()	207
11.13.2.24spinImageGetPaddingY()	208
11.13.2.25spinImageGetPayloadType()	208
11.13.2.26spinImageGetPixelFormat()	209
11.13.2.27spinImageGetPixelFormatName()	209
11.13.2.28spinImageGetPrivateData()	210
11.13.2.29spinImageGetSize()	210
11.13.2.30spinImageGetStatus()	211
11.13.2.31spinImageGetStatusDescription()	211
11.13.2.32spinImageGetStride()	212
11.13.2.33spinImageGetTimeStamp()	212
11.13.2.34spinImageGetTLPayloadType()	213
11.13.2.35spinImageGetTLPixelFormat()	213
11.13.2.36spinImageGetTLPixelFormatNamespace()	214
11.13.2.37spinImageGetValidPayloadSize()	214
11.13.2.38spinImageGetWidth()	215
11.13.2.39spinImageHasCRC()	215
11.13.2.40spinImageIsIncomplete()	216
11.13.2.41spinImageRelease()	216
11.13.2.42spinImageReset()	217
11.13.2.43spinImageResetEx()	217

11.13.2.44	<code>spinImageSave()</code>	218
11.13.2.45	<code>spinImageSaveBmp()</code>	218
11.13.2.46	<code>spinImageSaveFromExt()</code>	219
11.13.2.47	<code>spinImageSaveJpeg()</code>	219
11.13.2.48	<code>spinImageSaveJpg2()</code>	220
11.13.2.49	<code>spinImageSavePgm()</code>	220
11.13.2.50	<code>spinImageSavePng()</code>	221
11.13.2.51	<code>spinImageSavePpm()</code>	221
11.13.2.52	<code>spinImageSaveTiff()</code>	222
11.13.2.53	<code>spinImageSetDefaultColorProcessing()</code>	222
11.13.2.54	<code>spinImageSetNumDecompressionThreads()</code>	223
11.14	Event Access	224
11.14.1	Detailed Description	224
11.14.2	Function Documentation	224
11.14.2.1	<code>spinDeviceArrivalEventHandlerCreate()</code>	225
11.14.2.2	<code>spinDeviceArrivalEventHandlerDestroy()</code>	225
11.14.2.3	<code>spinDeviceEventHandlerCreate()</code>	226
11.14.2.4	<code>spinDeviceEventHandlerDestroy()</code>	226
11.14.2.5	<code>spinDeviceRemovalEventHandlerCreate()</code>	227
11.14.2.6	<code>spinDeviceRemovalEventHandlerDestroy()</code>	227
11.14.2.7	<code>spinImageEventHandlerCreate()</code>	228
11.14.2.8	<code>spinImageEventHandlerDestroy()</code>	228
11.14.2.9	<code>spinInterfaceEventHandlerCreate()</code>	229
11.14.2.10	<code>spinInterfaceEventHandlerDestroy()</code>	229
11.14.2.11	<code>spinLogEventHandlerCreate()</code>	230
11.14.2.12	<code>spinLogEventHandlerDestroy()</code>	230
11.15	ImageStatistics Access	231
11.15.1	Detailed Description	231
11.15.2	Function Documentation	232
11.15.2.1	<code>spinImageStatisticsCreate()</code>	232

11.15.2.2 spinImageStatisticsDestroy()	232
11.15.2.3 spinImageStatisticsDisableAll()	232
11.15.2.4 spinImageStatisticsEnableAll()	233
11.15.2.5 spinImageStatisticsEnableGreyOnly()	233
11.15.2.6 spinImageStatisticsEnableHslOnly()	234
11.15.2.7 spinImageStatisticsEnableRgbOnly()	234
11.15.2.8 spinImageStatisticsGetAll()	235
11.15.2.9 spinImageStatisticsGetChannelStatus()	235
11.15.2.10 spinImageStatisticsGetHistogram()	236
11.15.2.11 spinImageStatisticsGetMean()	236
11.15.2.12 spinImageStatisticsGetNumPixelValues()	237
11.15.2.13 spinImageStatisticsGetPixelValueRange()	237
11.15.2.14 spinImageStatisticsGetRange()	238
11.15.2.15 spinImageStatisticsSetChannelStatus()	238
11.16 Logging Event Data Access	240
11.16.1 Detailed Description	240
11.16.2 Function Documentation	240
11.16.2.1 spinLogDataGetCategoryName()	240
11.16.2.2 spinLogDataGetLogMessage()	241
11.16.2.3 spinLogDataGetNDC()	241
11.16.2.4 spinLogDataGetPriority()	242
11.16.2.5 spinLogDataGetPriorityName()	242
11.16.2.6 spinLogDataGetThreadName()	243
11.16.2.7 spinLogDataGetTimestamp()	243
11.17 Device Event Data Access	245
11.17.1 Detailed Description	245
11.17.2 Function Documentation	245
11.17.2.1 spinDeviceEventGetId()	245
11.17.2.2 spinDeviceEventGetName()	246
11.17.2.3 spinDeviceEventGetPayloadData()	246

11.17.2.4 spinDeviceEventGetPayloadDataSize()	247
11.18 Chunk data access	248
11.18.1 Detailed Description	248
11.18.2 Function Documentation	248
11.18.2.1 spinImageChunkDataGetFloatValue()	248
11.18.2.2 spinImageChunkDataGetIntValue()	248
11.19 Spinnaker C Handles	249
11.19.1 Detailed Description	250
11.19.2 Typedef Documentation	250
11.19.2.1 spinCamera	250
11.19.2.2 spinCameraList	250
11.19.2.3 spinDeviceArrivalEventHandler	250
11.19.2.4 spinDeviceEventData	250
11.19.2.5 spinDeviceEventHandler	251
11.19.2.6 spinDeviceRemovalEventHandler	251
11.19.2.7 spinImage	251
11.19.2.8 spinImageEventHandler	251
11.19.2.9 spinImageStatistics	251
11.19.2.10 spinInterface	251
11.19.2.11 spinInterfaceEventHandler	252
11.19.2.12 spinInterfaceList	252
11.19.2.13 spinLogEventData	252
11.19.2.14 spinLogEventHandler	252
11.19.2.15 spinSystem	252
11.19.2.16 spinVideo	252
11.20 Spinnaker C Function Signatures	253
11.20.1 Detailed Description	253
11.20.2 Typedef Documentation	253
11.20.2.1 spinArrivalEventFunction	253
11.20.2.2 spinDeviceEventFunction	253

11.20.2.3 spinImageEventFunction	254
11.20.2.4 spinLogEventFunction	254
11.20.2.5 spinRemovalEventFunction	254
11.21 Spinnaker C Enumerations	255
11.21.1 Detailed Description	257
11.21.2 Enumeration Type Documentation	257
11.21.2.1 spinColorProcessingAlgorithm	257
11.21.2.2 spinError	258
11.21.2.3 spinImageFileFormat	259
11.21.2.4 spinImageStatus	260
11.21.2.5 spinnakerLogLevel	260
11.21.2.6 spinPayloadTypeInfolDs	261
11.21.2.7 spinPixelFormatNamespaceID	261
11.21.2.8 spinStatisticsChannel	262
11.22 Spinnaker C Structures	263
11.22.1 Detailed Description	264
11.22.2 Enumeration Type Documentation	264
11.22.2.1 actionCommandStatus	264
11.22.2.2 spinCompressionMethod	264
11.23 Spinnaker C GenICam API	265
11.23.1 Detailed Description	266
11.23.2 Function Documentation	266
11.23.2.1 spinCategoryReleaseNode()	266
11.24 Node Map Access	268
11.24.1 Detailed Description	268
11.24.2 Function Documentation	268
11.24.2.1 spinNodeMapGetNode()	268
11.24.2.2 spinNodeMapGetNodeByIndex()	269
11.24.2.3 spinNodeMapGetNumNodes()	269
11.24.2.4 spinNodeMapPoll()	270

11.24.2.5 spinNodeMapReleaseNode()	270
11.25 Node Access	272
11.25.1 Detailed Description	273
11.25.2 Function Documentation	273
11.25.2.1 spinNodeDeregisterCallback()	273
11.25.2.2 spinNodeGetAccessMode()	274
11.25.2.3 spinNodeGetCachingMode()	274
11.25.2.4 spinNodeGetDescription()	275
11.25.2.5 spinNodeGetDisplayName()	275
11.25.2.6 spinNodeGetImposedAccessMode()	276
11.25.2.7 spinNodeGetImposedVisibility()	276
11.25.2.8 spinNodeGetName()	277
11.25.2.9 spinNodeGetNameSpace()	277
11.25.2.10 spinNodeGetPollingTime()	278
11.25.2.11 spinNodeGetToolTip()	278
11.25.2.12 spinNodeGetType()	279
11.25.2.13 spinNodeGetVisibility()	279
11.25.2.14 spinNodeInvalidateNode()	280
11.25.2.15 spinNodesAvailable()	280
11.25.2.16 spinNodesEqual()	280
11.25.2.17 spinNodesImplemented()	281
11.25.2.18 spinNodesReadable()	281
11.25.2.19 spinNodesWritable()	282
11.25.2.20 spinNodeRegisterCallback()	282
11.26 Value Access	284
11.26.1 Detailed Description	284
11.26.2 Function Documentation	284
11.26.2.1 spinNodeFromString()	284
11.26.2.2 spinNodeFromStringEx()	285
11.26.2.3 spinNodeToString()	285

11.26.2.4 spinNodeToStringEx()	286
11.27 String Access	287
11.27.1 Detailed Description	287
11.27.2 Function Documentation	287
11.27.2.1 spinStringGetMaxLength()	287
11.27.2.2 spinStringGetValue()	288
11.27.2.3 spinStringGetValueEx()	288
11.27.2.4 spinStringSetValue()	289
11.27.2.5 spinStringSetValueEx()	289
11.28 Integer Access	291
11.28.1 Detailed Description	291
11.28.2 Function Documentation	291
11.28.2.1 spinIntegerGetInc()	291
11.28.2.2 spinIntegerGetMax()	292
11.28.2.3 spinIntegerGetMin()	292
11.28.2.4 spinIntegerGetRepresentation()	293
11.28.2.5 spinIntegerGetValue()	293
11.28.2.6 spinIntegerGetValueEx()	294
11.28.2.7 spinIntegerSetValue()	294
11.28.2.8 spinIntegerSetValueEx()	295
11.29 Float Access	296
11.29.1 Detailed Description	296
11.29.2 Function Documentation	296
11.29.2.1 spinFloatGetMax()	296
11.29.2.2 spinFloatGetMin()	297
11.29.2.3 spinFloatGetRepresentation()	297
11.29.2.4 spinFloatGetUnit()	298
11.29.2.5 spinFloatGetValue()	298
11.29.2.6 spinFloatGetValueEx()	299
11.29.2.7 spinFloatSetValue()	299

11.29.2.8 spinFloatSetValueEx()	300
11.30 IEnumeration Access	301
11.30.1 Detailed Description	301
11.30.2 Function Documentation	301
11.30.2.1 spinEnumerationGetCurrentEntry()	302
11.30.2.2 spinEnumerationGetEntryByIndex()	302
11.30.2.3 spinEnumerationGetEntryByName()	303
11.30.2.4 spinEnumerationGetNumEntries()	303
11.30.2.5 spinEnumerationReleaseNode()	304
11.30.2.6 spinEnumerationSetEnumValue()	304
11.30.2.7 spinEnumerationSetIntValue()	305
11.31 IEnumEntry Access	306
11.31.1 Detailed Description	306
11.31.2 Function Documentation	306
11.31.2.1 spinEnumerationEntryGetEnumValue()	306
11.31.2.2 spinEnumerationEntryGetIntValue()	307
11.31.2.3 spinEnumerationEntryGetSymbolic()	307
11.32 IBoolean Access	309
11.32.1 Detailed Description	309
11.32.2 Function Documentation	309
11.32.2.1 spinBooleanGetValue()	309
11.32.2.2 spinBooleanSetValue()	310
11.33 ICommand Access	311
11.33.1 Detailed Description	311
11.33.2 Function Documentation	311
11.33.2.1 spinCommandExecute()	311
11.33.2.2 spinCommandIsDone()	312
11.34 ICategory Access	313
11.34.1 Detailed Description	313
11.34.2 Function Documentation	313

11.34.2.1 spinCategoryGetFeatureByIndex()	313
11.34.2.2 spinCategoryGetNumFeatures()	314
11.35 IRegister Access	315
11.35.1 Detailed Description	315
11.35.2 Function Documentation	315
11.35.2.1 spinRegisterGet()	316
11.35.2.2 spinRegisterGetAddress()	316
11.35.2.3 spinRegisterGetEx()	317
11.35.2.4 spinRegisterGetLength()	317
11.35.2.5 spinRegisterSet()	318
11.35.2.6 spinRegisterSetEx()	318
11.35.2.7 spinRegisterSetReference()	319
11.36 Spinnaker C GenICam Handles	320
11.36.1 Detailed Description	320
11.36.2 Typedef Documentation	320
11.36.2.1 spinNodeCallbackFunction	320
11.36.2.2 spinNodeCallbackHandle	320
11.36.2.3 spinNodeHandle	321
11.36.2.4 spinNodeMapHandle	321
11.37 Spinnaker C GenICam Enumerations	322
11.37.1 Detailed Description	324
11.37.2 Enumeration Type Documentation	324
11.37.2.1 spinAccessMode	324
11.37.2.2 spinCachingMode	325
11.37.2.3 spinDisplayNotation	325
11.37.2.4 spinEndianess	325
11.37.2.5 spinIncMode	326
11.37.2.6 spinInputDirection	326
11.37.2.7 spinInterfaceType	326
11.37.2.8 spinLinkType	327

11.37.2.9 spinNameSpace	328
11.37.2.10 spinNodeType	328
11.37.2.11 spinRepresentation	329
11.37.2.12 spinSign	329
11.37.2.13 spinSlope	329
11.37.2.14 spinStandardNameSpace	330
11.37.2.15 spinVisibility	330
11.37.2.16 spinXMLValidation	330
11.37.2.17 spinYesNo	332
11.38 SpinVideo Recording Access	333
11.38.1 Detailed Description	333
11.38.2 Function Documentation	333
11.38.2.1 spinVideoAppend()	333
11.38.2.2 spinVideoClose()	334
11.38.2.3 spinVideoOpenH264()	334
11.38.2.4 spinVideoOpenMJPEG()	334
11.38.2.5 spinVideoOpenUncompressed()	334
11.38.2.6 spinVideoSetMaximumFileSize()	334
11.39 Transport Layer Enumerations	336
11.39.1 Detailed Description	337
11.39.2 Enumeration Type Documentation	337
11.39.2.1 spinTLDeviceAccessStatusEnums	338
11.39.2.2 spinTLDeviceCurrentSpeedEnums	339
11.39.2.3 spinTLDeviceEndiannessMechanismEnums	339
11.39.2.4 spinTLDeviceTypeEnums	340
11.39.2.5 spinTLFilterDriverStatusEnums	340
11.39.2.6 spinTLGenICamXMLLocationEnums	340
11.39.2.7 spinTLGevCCPEnums	341
11.39.2.8 spinTLGUIXMLLocationEnums	341
11.39.2.9 spinTLInterfaceTypeEnums	341
11.39.2.10 spinTLPOEStatusEnums	342
11.39.2.11 spinTLStreamBufferCountModeEnums	342
11.39.2.12 spinTLStreamBufferHandlingModeEnums	342
11.39.2.13 spinTLStreamTypeEnums	343
11.39.2.14 spinTLTLTypeEnums	343
11.40 TLDevice Structures	345
11.40.1 Detailed Description	345
11.41 TLInterface Structures	346
11.41.1 Detailed Description	346
11.42 TLStream Structures	347
11.42.1 Detailed Description	347
11.43 TLSystem Structures	348
11.43.1 Detailed Description	348

12 Data Structure Documentation	349
12.1 actionCommandResult Struct Reference	349
12.1.1 Detailed Description	349
12.1.2 Field Documentation	349
12.1.2.1 DeviceAddress	349
12.1.2.2 Status	349
12.2 quickSpin Struct Reference	350
12.2.1 Field Documentation	362
12.2.1.1 AasRoiEnable	362
12.2.1.2 AasRoiHeight	362
12.2.1.3 AasRoiOffsetX	362
12.2.1.4 AasRoiOffsetY	362
12.2.1.5 AasRoiWidth	363
12.2.1.6 AcquisitionAbort	363
12.2.1.7 AcquisitionArm	363
12.2.1.8 AcquisitionBurstFrameCount	363
12.2.1.9 AcquisitionFrameCount	363
12.2.1.10 AcquisitionFrameRate	363
12.2.1.11 AcquisitionFrameRateEnable	363
12.2.1.12 AcquisitionLineRate	363
12.2.1.13 AcquisitionMode	364
12.2.1.14 AcquisitionResultingFrameRate	364
12.2.1.15 AcquisitionStart	364
12.2.1.16 AcquisitionStatus	364
12.2.1.17 AcquisitionStatusSelector	364
12.2.1.18 AcquisitionStop	364
12.2.1.19 ActionDeviceKey	364
12.2.1.20 ActionGroupKey	364
12.2.1.21 ActionGroupMask	365
12.2.1.22 ActionQueueSize	365

12.2.1.23 ActionSelector	365
12.2.1.24 ActionUnconditionalMode	365
12.2.1.25 AdaptiveCompressionEnable	365
12.2.1.26 AdcBitDepth	365
12.2.1.27 aPAUSEMACCtrlFramesReceived	365
12.2.1.28 aPAUSEMACCtrlFramesTransmitted	365
12.2.1.29 AutoAlgorithmSelector	366
12.2.1.30 AutoExposureControlLoopDamping	366
12.2.1.31 AutoExposureControlPriority	366
12.2.1.32 AutoExposureEVCompensation	366
12.2.1.33 AutoExposureExposureTimeLowerLimit	366
12.2.1.34 AutoExposureExposureTimeUpperLimit	366
12.2.1.35 AutoExposureGainLowerLimit	366
12.2.1.36 AutoExposureGainUpperLimit	366
12.2.1.37 AutoExposureGreyValueLowerLimit	367
12.2.1.38 AutoExposureGreyValueUpperLimit	367
12.2.1.39 AutoExposureLightingMode	367
12.2.1.40 AutoExposureMeteringMode	367
12.2.1.41 AutoExposureTargetGreyValue	367
12.2.1.42 AutoExposureTargetGreyValueAuto	367
12.2.1.43 BalanceRatio	367
12.2.1.44 BalanceRatioSelector	367
12.2.1.45 BalanceWhiteAuto	368
12.2.1.46 BalanceWhiteAutoDamping	368
12.2.1.47 BalanceWhiteAutoLowerLimit	368
12.2.1.48 BalanceWhiteAutoProfile	368
12.2.1.49 BalanceWhiteAutoUpperLimit	368
12.2.1.50 BinningHorizontal	368
12.2.1.51 BinningHorizontalMode	368
12.2.1.52 BinningSelector	368

12.2.1.53 BinningVertical	369
12.2.1.54 BinningVerticalMode	369
12.2.1.55 BlackLevel	369
12.2.1.56 BlackLevelAuto	369
12.2.1.57 BlackLevelAutoBalance	369
12.2.1.58 BlackLevelClampingEnable	369
12.2.1.59 BlackLevelRaw	369
12.2.1.60 BlackLevelSelector	369
12.2.1.61 ChunkBlackLevel	370
12.2.1.62 ChunkBlackLevelSelector	370
12.2.1.63 ChunkCompressionMode	370
12.2.1.64 ChunkCompressionRatio	370
12.2.1.65 ChunkCounterSelector	370
12.2.1.66 ChunkCounterValue	370
12.2.1.67 ChunkCRC	370
12.2.1.68 ChunkEnable	370
12.2.1.69 ChunkEncoderSelector	371
12.2.1.70 ChunkEncoderStatus	371
12.2.1.71 ChunkEncoderValue	371
12.2.1.72 ChunkExposureEndLineStatusAll	371
12.2.1.73 ChunkExposureTime	371
12.2.1.74 ChunkExposureTimeSelector	371
12.2.1.75 ChunkFrameID	371
12.2.1.76 ChunkGain	371
12.2.1.77 ChunkGainSelector	372
12.2.1.78 ChunkHeight	372
12.2.1.79 ChunkImage	372
12.2.1.80 ChunkImageComponent	372
12.2.1.81 ChunkInferenceBoundingBoxResult	372
12.2.1.82 ChunkInferenceConfidence	372

12.2.1.83 ChunkInferenceFrameId	372
12.2.1.84 ChunkInferenceResult	372
12.2.1.85 ChunkLinePitch	373
12.2.1.86 ChunkLineStatusAll	373
12.2.1.87 ChunkModeActive	373
12.2.1.88 ChunkOffsetX	373
12.2.1.89 ChunkOffsetY	373
12.2.1.90 ChunkPartSelector	373
12.2.1.91 ChunkPixelDynamicRangeMax	373
12.2.1.92 ChunkPixelDynamicRangeMin	373
12.2.1.93 ChunkPixelFormat	374
12.2.1.94 ChunkRegionID	374
12.2.1.95 ChunkScan3dAxisMax	374
12.2.1.96 ChunkScan3dAxisMin	374
12.2.1.97 ChunkScan3dCoordinateOffset	374
12.2.1.98 ChunkScan3dCoordinateReferenceSelector	374
12.2.1.99 ChunkScan3dCoordinateReferenceValue	374
12.2.1.100 ChunkScan3dCoordinateScale	374
12.2.1.101 ChunkScan3dCoordinateSelector	375
12.2.1.102 ChunkScan3dCoordinateSystem	375
12.2.1.103 ChunkScan3dCoordinateSystemReference	375
12.2.1.104 ChunkScan3dCoordinateTransformSelector	375
12.2.1.105 ChunkScan3dDistanceUnit	375
12.2.1.106 ChunkScan3dInvalidDataFlag	375
12.2.1.107 ChunkScan3dInvalidDataValue	375
12.2.1.108 ChunkScan3dOutputMode	375
12.2.1.109 ChunkScan3dTransformValue	376
12.2.1.110 ChunkScanLineSelector	376
12.2.1.111 ChunkSelector	376
12.2.1.112 ChunkSequencerSetActive	376

12.2.1.113	ChunkSerialData	376
12.2.1.114	ChunkSerialDataLength	376
12.2.1.115	ChunkSerialReceiveOverflow	376
12.2.1.116	ChunkSourceID	376
12.2.1.117	ChunkStreamChannelID	377
12.2.1.118	ChunkTimerSelector	377
12.2.1.119	ChunkTimerValue	377
12.2.1.120	ChunkTimestamp	377
12.2.1.121	ChunkTimestampLatchValue	377
12.2.1.122	ChunkTransferBlockID	377
12.2.1.123	ChunkTransferQueueCurrentBlockCount	377
12.2.1.124	ChunkTransferStreamID	377
12.2.1.125	ChunkWidth	378
12.2.1.126	CIConfiguration	378
12.2.1.127	CITimeSlotsCount	378
12.2.1.128	ColorTransformationEnable	378
12.2.1.129	ColorTransformationSelector	378
12.2.1.130	ColorTransformationValue	378
12.2.1.131	ColorTransformationValueSelector	378
12.2.1.132	CompressionRatio	378
12.2.1.133	CompressionSaturationPriority	379
12.2.1.134	CounterDelay	379
12.2.1.135	CounterDuration	379
12.2.1.136	CounterEventActivation	379
12.2.1.137	CounterEventSource	379
12.2.1.138	CounterReset	379
12.2.1.139	CounterResetActivation	379
12.2.1.140	CounterResetSource	379
12.2.1.141	CounterSelector	380
12.2.1.142	CounterStatus	380

12.2.1.143CounterTriggerActivation	380
12.2.1.144CounterTriggerSource	380
12.2.1.145CounterValue	380
12.2.1.146CounterValueAtReset	380
12.2.1.147CxpConnectionSelector	380
12.2.1.148CxpConnectionTestErrorCount	380
12.2.1.149CxpConnectionTestMode	381
12.2.1.150CxpConnectionTestPacketCount	381
12.2.1.151CxpLinkConfiguration	381
12.2.1.152CxpLinkConfigurationPreferred	381
12.2.1.153CxpLinkConfigurationStatus	381
12.2.1.154CxpPoCxpAuto	381
12.2.1.155CxpPoCxpStatus	381
12.2.1.156CxpPoCxpTripReset	381
12.2.1.157CxpPoCxpTurnOff	382
12.2.1.158DecimationHorizontal	382
12.2.1.159DecimationHorizontalMode	382
12.2.1.160DecimationSelector	382
12.2.1.161DecimationVertical	382
12.2.1.162DecimationVerticalMode	382
12.2.1.163DefectCorrectionMode	382
12.2.1.164DefectCorrectStaticEnable	382
12.2.1.165DefectTableApply	383
12.2.1.166DefectTableCoordinateX	383
12.2.1.167DefectTableCoordinateY	383
12.2.1.168DefectTableFactoryRestore	383
12.2.1.169DefectTableIndex	383
12.2.1.170DefectTablePixelCount	383
12.2.1.171DefectTableSave	383
12.2.1.172Deinterlacing	383

12.2.1.173	DeviceCharacterSet	384
12.2.1.174	DeviceClockFrequency	384
12.2.1.175	DeviceClockSelector	384
12.2.1.176	DeviceConnectionSelector	384
12.2.1.177	DeviceConnectionSpeed	384
12.2.1.178	DeviceConnectionStatus	384
12.2.1.179	DeviceEventChannelCount	384
12.2.1.180	DeviceFamilyName	384
12.2.1.181	DeviceFeaturePersistenceEnd	385
12.2.1.182	DeviceFeaturePersistenceStart	385
12.2.1.183	DeviceFirmwareVersion	385
12.2.1.184	DeviceGenCPVersionMajor	385
12.2.1.185	DeviceGenCPVersionMinor	385
12.2.1.186	DeviceID	385
12.2.1.187	DeviceIndicatorMode	385
12.2.1.188	DeviceLinkBandwidthReserve	385
12.2.1.189	DeviceLinkCommandTimeout	386
12.2.1.190	DeviceLinkConnectionCount	386
12.2.1.191	DeviceLinkCurrentThroughput	386
12.2.1.192	DeviceLinkHeartbeatMode	386
12.2.1.193	DeviceLinkHeartbeatTimeout	386
12.2.1.194	DeviceLinkSelector	386
12.2.1.195	DeviceLinkSpeed	386
12.2.1.196	DeviceLinkThroughputLimit	386
12.2.1.197	DeviceLinkThroughputLimitMode	387
12.2.1.198	DeviceManifestEntrySelector	387
12.2.1.199	DeviceManifestPrimaryURL	387
12.2.1.200	DeviceManifestSchemaMajorVersion	387
12.2.1.201	DeviceManifestSchemaMinorVersion	387
12.2.1.202	DeviceManifestSecondaryURL	387

12.2.1.203	DeviceManifestXMLMajorVersion	387
12.2.1.204	DeviceManifestXMLMinorVersion	387
12.2.1.205	DeviceManifestXMLSubMinorVersion	388
12.2.1.206	DeviceManufacturerInfo	388
12.2.1.207	DeviceMaxThroughput	388
12.2.1.208	DeviceModelName	388
12.2.1.209	DevicePowerSupplySelector	388
12.2.1.210	DeviceRegistersCheck	388
12.2.1.211	DeviceRegistersEndianness	388
12.2.1.212	DeviceRegistersStreamingEnd	388
12.2.1.213	DeviceRegistersStreamingStart	389
12.2.1.214	DeviceRegistersValid	389
12.2.1.215	DeviceReset	389
12.2.1.216	DeviceScanType	389
12.2.1.217	DeviceSerialNumber	389
12.2.1.218	DeviceSerialPortBaudRate	389
12.2.1.219	DeviceSerialPortSelector	389
12.2.1.220	DeviceSFNCVersionMajor	389
12.2.1.221	DeviceSFNCVersionMinor	390
12.2.1.222	DeviceSFNCVersionSubMinor	390
12.2.1.223	DeviceStreamChannelCount	390
12.2.1.224	DeviceStreamChannelEndianness	390
12.2.1.225	DeviceStreamChannelLink	390
12.2.1.226	DeviceStreamChannelPacketSize	390
12.2.1.227	DeviceStreamChannelSelector	390
12.2.1.228	DeviceStreamChannelType	390
12.2.1.229	DeviceTapGeometry	391
12.2.1.230	DeviceTemperature	391
12.2.1.231	DeviceTemperatureSelector	391
12.2.1.232	DeviceTLType	391

12.2.1.233	DeviceTLVersionMajor	391
12.2.1.234	DeviceTLVersionMinor	391
12.2.1.235	DeviceTLVersionSubMinor	391
12.2.1.236	DeviceType	391
12.2.1.237	DeviceUptime	392
12.2.1.238	DeviceUserID	392
12.2.1.239	DeviceVendorName	392
12.2.1.240	DeviceVersion	392
12.2.1.241	EncoderDivider	392
12.2.1.242	EncoderMode	392
12.2.1.243	EncoderOutputMode	392
12.2.1.244	EncoderReset	392
12.2.1.245	EncoderResetActivation	393
12.2.1.246	EncoderResetSource	393
12.2.1.247	EncoderSelector	393
12.2.1.248	EncoderSourceA	393
12.2.1.249	EncoderSourceB	393
12.2.1.250	EncoderStatus	393
12.2.1.251	EncoderTimeout	393
12.2.1.252	EncoderValue	393
12.2.1.253	EncoderValueAtReset	394
12.2.1.254	EnumerationCount	394
12.2.1.255	EventAcquisitionEnd	394
12.2.1.256	EventAcquisitionEndFrameID	394
12.2.1.257	EventAcquisitionEndTimestamp	394
12.2.1.258	EventAcquisitionError	394
12.2.1.259	EventAcquisitionErrorFrameID	394
12.2.1.260	EventAcquisitionErrorTimestamp	394
12.2.1.261	EventAcquisitionStart	395
12.2.1.262	EventAcquisitionStartFrameID	395

12.2.1.263	EventAcquisitionStartTimestamp	395
12.2.1.264	EventAcquisitionTransferEnd	395
12.2.1.265	EventAcquisitionTransferEndFrameID	395
12.2.1.266	EventAcquisitionTransferEndTimestamp	395
12.2.1.267	EventAcquisitionTransferStart	395
12.2.1.268	EventAcquisitionTransferStartFrameID	395
12.2.1.269	EventAcquisitionTransferStartTimestamp	396
12.2.1.270	EventAcquisitionTrigger	396
12.2.1.271	EventAcquisitionTriggerFrameID	396
12.2.1.272	EventAcquisitionTriggerTimestamp	396
12.2.1.273	EventActionLate	396
12.2.1.274	EventActionLateFrameID	396
12.2.1.275	EventActionLateTimestamp	396
12.2.1.276	EventCounter0End	396
12.2.1.277	EventCounter0EndFrameID	397
12.2.1.278	EventCounter0EndTimestamp	397
12.2.1.279	EventCounter0Start	397
12.2.1.280	EventCounter0StartFrameID	397
12.2.1.281	EventCounter0StartTimestamp	397
12.2.1.282	EventCounter1End	397
12.2.1.283	EventCounter1EndFrameID	397
12.2.1.284	EventCounter1EndTimestamp	397
12.2.1.285	EventCounter1Start	398
12.2.1.286	EventCounter1StartFrameID	398
12.2.1.287	EventCounter1StartTimestamp	398
12.2.1.288	EventEncoder0Restarted	398
12.2.1.289	EventEncoder0RestartedFrameID	398
12.2.1.290	EventEncoder0RestartedTimestamp	398
12.2.1.291	EventEncoder0Stopped	398
12.2.1.292	EventEncoder0StoppedFrameID	398

12.2.1.293	EventEncoder0StoppedTimestamp	399
12.2.1.294	EventEncoder1Restarted	399
12.2.1.295	EventEncoder1RestartedFrameID	399
12.2.1.296	EventEncoder1RestartedTimestamp	399
12.2.1.297	EventEncoder1Stopped	399
12.2.1.298	EventEncoder1StoppedFrameID	399
12.2.1.299	EventEncoder1StoppedTimestamp	399
12.2.1.300	EventError	399
12.2.1.301	EventErrorCode	400
12.2.1.302	EventErrorFrameID	400
12.2.1.303	EventErrorTimestamp	400
12.2.1.304	EventExposureEnd	400
12.2.1.305	EventExposureEndFrameID	400
12.2.1.306	EventExposureEndTimestamp	400
12.2.1.307	EventExposureStart	400
12.2.1.308	EventExposureStartFrameID	400
12.2.1.309	EventExposureStartTimestamp	401
12.2.1.310	EventFrameBurstEnd	401
12.2.1.311	EventFrameBurstEndFrameID	401
12.2.1.312	EventFrameBurstEndTimestamp	401
12.2.1.313	EventFrameBurstStart	401
12.2.1.314	EventFrameBurstStartFrameID	401
12.2.1.315	EventFrameBurstStartTimestamp	401
12.2.1.316	EventFrameEnd	401
12.2.1.317	EventFrameEndFrameID	402
12.2.1.318	EventFrameEndTimestamp	402
12.2.1.319	EventFrameStart	402
12.2.1.320	EventFrameStartFrameID	402
12.2.1.321	EventFrameStartTimestamp	402
12.2.1.322	EventFrameTransferEnd	402

12.2.1.323	EventFrameTransferEndFrameID	402
12.2.1.324	EventFrameTransferEndTimestamp	402
12.2.1.325	EventFrameTransferStart	403
12.2.1.326	EventFrameTransferStartFrameID	403
12.2.1.327	EventFrameTransferStartTimestamp	403
12.2.1.328	EventFrameTrigger	403
12.2.1.329	EventFrameTriggerFrameID	403
12.2.1.330	EventFrameTriggerTimestamp	403
12.2.1.331	EventLine0AnyEdge	403
12.2.1.332	EventLine0AnyEdgeFrameID	403
12.2.1.333	EventLine0AnyEdgeTimestamp	404
12.2.1.334	EventLine0FallingEdge	404
12.2.1.335	EventLine0FallingEdgeFrameID	404
12.2.1.336	EventLine0FallingEdgeTimestamp	404
12.2.1.337	EventLine0RisingEdge	404
12.2.1.338	EventLine0RisingEdgeFrameID	404
12.2.1.339	EventLine0RisingEdgeTimestamp	404
12.2.1.340	EventLine1AnyEdge	404
12.2.1.341	EventLine1AnyEdgeFrameID	405
12.2.1.342	EventLine1AnyEdgeTimestamp	405
12.2.1.343	EventLine1FallingEdge	405
12.2.1.344	EventLine1FallingEdgeFrameID	405
12.2.1.345	EventLine1FallingEdgeTimestamp	405
12.2.1.346	EventLine1RisingEdge	405
12.2.1.347	EventLine1RisingEdgeFrameID	405
12.2.1.348	EventLine1RisingEdgeTimestamp	405
12.2.1.349	EventLinkSpeedChange	406
12.2.1.350	EventLinkSpeedChangeFrameID	406
12.2.1.351	EventLinkSpeedChangeTimestamp	406
12.2.1.352	EventLinkTrigger0	406

12.2.1.353	EventLinkTrigger0FrameID	406
12.2.1.354	EventLinkTrigger0Timestamp	406
12.2.1.355	EventLinkTrigger1	406
12.2.1.356	EventLinkTrigger1FrameID	406
12.2.1.357	EventLinkTrigger1Timestamp	407
12.2.1.358	EventNotification	407
12.2.1.359	EventSelector	407
12.2.1.360	EventSequencerSetChange	407
12.2.1.361	EventSequencerSetChangeFrameID	407
12.2.1.362	EventSequencerSetChangeTimestamp	407
12.2.1.363	EventSerialData	407
12.2.1.364	EventSerialDataLength	407
12.2.1.365	EventSerialPortReceive	408
12.2.1.366	EventSerialPortReceiveTimestamp	408
12.2.1.367	EventSerialReceiveOverflow	408
12.2.1.368	EventStream0TransferBlockEnd	408
12.2.1.369	EventStream0TransferBlockEndFrameID	408
12.2.1.370	EventStream0TransferBlockEndTimestamp	408
12.2.1.371	EventStream0TransferBlockStart	408
12.2.1.372	EventStream0TransferBlockStartFrameID	408
12.2.1.373	EventStream0TransferBlockStartTimestamp	409
12.2.1.374	EventStream0TransferBlockTrigger	409
12.2.1.375	EventStream0TransferBlockTriggerFrameID	409
12.2.1.376	EventStream0TransferBlockTriggerTimestamp	409
12.2.1.377	EventStream0TransferBurstEnd	409
12.2.1.378	EventStream0TransferBurstEndFrameID	409
12.2.1.379	EventStream0TransferBurstEndTimestamp	409
12.2.1.380	EventStream0TransferBurstStart	409
12.2.1.381	EventStream0TransferBurstStartFrameID	410
12.2.1.382	EventStream0TransferBurstStartTimestamp	410

12.2.1.383	EventStream0TransferEnd	410
12.2.1.384	EventStream0TransferEndFrameID	410
12.2.1.385	EventStream0TransferEndTimestamp	410
12.2.1.386	EventStream0TransferOverflow	410
12.2.1.387	EventStream0TransferOverflowFrameID	410
12.2.1.388	EventStream0TransferOverflowTimestamp	410
12.2.1.389	EventStream0TransferPause	411
12.2.1.390	EventStream0TransferPauseFrameID	411
12.2.1.391	EventStream0TransferPauseTimestamp	411
12.2.1.392	EventStream0TransferResume	411
12.2.1.393	EventStream0TransferResumeFrameID	411
12.2.1.394	EventStream0TransferResumeTimestamp	411
12.2.1.395	EventStream0TransferStart	411
12.2.1.396	EventStream0TransferStartFrameID	411
12.2.1.397	EventStream0TransferStartTimestamp	412
12.2.1.398	EventTest	412
12.2.1.399	EventTestTimestamp	412
12.2.1.400	EventTimer0End	412
12.2.1.401	EventTimer0EndFrameID	412
12.2.1.402	EventTimer0EndTimestamp	412
12.2.1.403	EventTimer0Start	412
12.2.1.404	EventTimer0StartFrameID	412
12.2.1.405	EventTimer0StartTimestamp	413
12.2.1.406	EventTimer1End	413
12.2.1.407	EventTimer1EndFrameID	413
12.2.1.408	EventTimer1EndTimestamp	413
12.2.1.409	EventTimer1Start	413
12.2.1.410	EventTimer1StartFrameID	413
12.2.1.411	EventTimer1StartTimestamp	413
12.2.1.412	ExposureActiveMode	413

12.2.1.413	ExposureAuto	414
12.2.1.414	ExposureMode	414
12.2.1.415	ExposureTime	414
12.2.1.416	ExposureTimeMode	414
12.2.1.417	ExposureTimeSelector	414
12.2.1.418	FactoryReset	414
12.2.1.419	FileAccessBuffer	414
12.2.1.420	FileAccessLength	414
12.2.1.421	FileAccessOffset	415
12.2.1.422	FileOpenMode	415
12.2.1.423	FileOperationExecute	415
12.2.1.424	FileOperationResult	415
12.2.1.425	FileOperationSelector	415
12.2.1.426	FileOperationStatus	415
12.2.1.427	FileSelector	415
12.2.1.428	FileSize	415
12.2.1.429	Gain	416
12.2.1.430	GainAuto	416
12.2.1.431	GainAutoBalance	416
12.2.1.432	GainSelector	416
12.2.1.433	Gamma	416
12.2.1.434	GammaEnable	416
12.2.1.435	GevActiveLinkCount	416
12.2.1.436	GevCCP	416
12.2.1.437	GevCurrentDefaultGateway	417
12.2.1.438	GevCurrentIPAddress	417
12.2.1.439	GevCurrentIPConfigurationDHCP	417
12.2.1.440	GevCurrentIPConfigurationLLA	417
12.2.1.441	GevCurrentIPConfigurationPersistentIP	417
12.2.1.442	GevCurrentPhysicalLinkConfiguration	417

12.2.1.443GevCurrentSubnetMask	417
12.2.1.444GevDiscoveryAckDelay	417
12.2.1.445GevFirstURL	418
12.2.1.446GevGVCPExtendedStatusCodes	418
12.2.1.447GevGVCPExtendedStatusCodesSelector	418
12.2.1.448GevGVCPHeartbeatDisable	418
12.2.1.449GevGVCPPendingAck	418
12.2.1.450GevGVCPPendingTimeout	418
12.2.1.451GevGVSPExtendedIDMode	418
12.2.1.452GevHeartbeatTimeout	418
12.2.1.453GevIEEE1588	419
12.2.1.454GevIEEE1588ClockAccuracy	419
12.2.1.455GevIEEE1588Mode	419
12.2.1.456GevIEEE1588Status	419
12.2.1.457GevInterfaceSelector	419
12.2.1.458GevIPConfigurationStatus	419
12.2.1.459GevMACAddress	419
12.2.1.460GevMCDA	419
12.2.1.461GevMCPHostPort	420
12.2.1.462GevMCRC	420
12.2.1.463GevMCSP	420
12.2.1.464GevMCTT	420
12.2.1.465GevNumberOfInterfaces	420
12.2.1.466GevPAUSEFrameReception	420
12.2.1.467GevPAUSEFrameTransmission	420
12.2.1.468GevPersistentDefaultGateway	420
12.2.1.469GevPersistentIPAddress	421
12.2.1.470GevPersistentSubnetMask	421
12.2.1.471GevPhysicalLinkConfiguration	421
12.2.1.472GevPrimaryApplicationIPAddress	421

12.2.1.473GevPrimaryApplicationSocket	421
12.2.1.474GevPrimaryApplicationSwitchoverKey	421
12.2.1.475GevSCCFGAllInTransmission	421
12.2.1.476GevSCCFGExtendedChunkData	421
12.2.1.477GevSCCFGPacketResendDestination	422
12.2.1.478GevSCCFGUnconditionalStreaming	422
12.2.1.479GevSCDA	422
12.2.1.480GevSCPD	422
12.2.1.481GevSCPDirection	422
12.2.1.482GevSCPHostPort	422
12.2.1.483GevSCPInterfaceIndex	422
12.2.1.484GevSCPSBigEndian	422
12.2.1.485GevSCPSDoNotFragment	423
12.2.1.486GevSCPSFireTestPacket	423
12.2.1.487GevSCPSPacketSize	423
12.2.1.488GevSCSP	423
12.2.1.489GevSCZoneConfigurationLock	423
12.2.1.490GevSCZoneCount	423
12.2.1.491GevSCZoneDirectionAll	423
12.2.1.492GevSecondURL	423
12.2.1.493GevStreamChannelSelector	424
12.2.1.494GevSupportedOption	424
12.2.1.495GevSupportedOptionSelector	424
12.2.1.496GevTimestampTickFrequency	424
12.2.1.497GuiXmlManifestAddress	424
12.2.1.498Height	424
12.2.1.499HeightMax	424
12.2.1.500ImageComponentEnable	424
12.2.1.501ImageComponentSelector	425
12.2.1.502ImageCompressionBitrate	425

12.2.1.503ImageCompressionJPEGFormatOption	425
12.2.1.504ImageCompressionMode	425
12.2.1.505ImageCompressionQuality	425
12.2.1.506ImageCompressionRateOption	425
12.2.1.507spEnable	425
12.2.1.508LineFilterWidth	425
12.2.1.509LineFormat	426
12.2.1.510LineInputFilterSelector	426
12.2.1.511LineInverter	426
12.2.1.512LineMode	426
12.2.1.513LinePitch	426
12.2.1.514LineSelector	426
12.2.1.515LineSource	426
12.2.1.516LineStatus	426
12.2.1.517LineStatusAll	427
12.2.1.518LinkErrorCount	427
12.2.1.519LinkUptime	427
12.2.1.520LogicBlockLUTInputActivation	427
12.2.1.521LogicBlockLUTInputSelector	427
12.2.1.522LogicBlockLUTInputSource	427
12.2.1.523LogicBlockLUTOutputValue	427
12.2.1.524LogicBlockLUTOutputValueAll	427
12.2.1.525LogicBlockLUTRowIndex	428
12.2.1.526LogicBlockLUTSelector	428
12.2.1.527LogicBlockSelector	428
12.2.1.528LUTEnable	428
12.2.1.529LUTIndex	428
12.2.1.530LUTSelector	428
12.2.1.531LUTValue	428
12.2.1.532LUTValueAll	428

12.2.1.533MaxDeviceResetTime	429
12.2.1.534OffsetX	429
12.2.1.535OffsetY	429
12.2.1.536PacketResendRequestCount	429
12.2.1.537PayloadSize	429
12.2.1.538PixelColorFilter	429
12.2.1.539PixelDynamicRangeMax	429
12.2.1.540PixelDynamicRangeMin	429
12.2.1.541PixelFormat	430
12.2.1.542PixelFormatInfoID	430
12.2.1.543PixelFormatInfoSelector	430
12.2.1.544PixelSize	430
12.2.1.545PowerSupplyCurrent	430
12.2.1.546PowerSupplyVoltage	430
12.2.1.547RegionDestination	430
12.2.1.548RegionMode	430
12.2.1.549RegionSelector	431
12.2.1.550ReverseX	431
12.2.1.551ReverseY	431
12.2.1.552RgbTransformLightSource	431
12.2.1.553Saturation	431
12.2.1.554SaturationEnable	431
12.2.1.555Scan3dAxisMax	431
12.2.1.556Scan3dAxisMin	431
12.2.1.557Scan3dCoordinateOffset	432
12.2.1.558Scan3dCoordinateReferenceSelector	432
12.2.1.559Scan3dCoordinateReferenceValue	432
12.2.1.560Scan3dCoordinateScale	432
12.2.1.561Scan3dCoordinateSelector	432
12.2.1.562Scan3dCoordinateSystem	432

12.2.1.563Scan3dCoordinateSystemReference	432
12.2.1.564Scan3dCoordinateTransformSelector	432
12.2.1.565Scan3dDistanceUnit	433
12.2.1.566Scan3dInvalidDataFlag	433
12.2.1.567Scan3dInvalidDataValue	433
12.2.1.568Scan3dOutputMode	433
12.2.1.569Scan3dTransformValue	433
12.2.1.570SensorDescription	433
12.2.1.571SensorDigitizationTaps	433
12.2.1.572SensorHeight	433
12.2.1.573SensorShutterMode	434
12.2.1.574SensorTaps	434
12.2.1.575SensorWidth	434
12.2.1.576SequencerConfigurationMode	434
12.2.1.577SequencerConfigurationValid	434
12.2.1.578SequencerFeatureEnable	434
12.2.1.579SequencerMode	434
12.2.1.580SequencerPathSelector	434
12.2.1.581SequencerSetActive	435
12.2.1.582SequencerSetLoad	435
12.2.1.583SequencerSetNext	435
12.2.1.584SequencerSetSave	435
12.2.1.585SequencerSetSelector	435
12.2.1.586SequencerSetStart	435
12.2.1.587SequencerSetValid	435
12.2.1.588SequencerTriggerActivation	435
12.2.1.589SequencerTriggerSource	436
12.2.1.590SerialPortBaudRate	436
12.2.1.591SerialPortDataBits	436
12.2.1.592SerialPortParity	436

12.2.1.593SerialPortSelector	436
12.2.1.594SerialPortSource	436
12.2.1.595SerialPortStopBits	436
12.2.1.596SerialReceiveFramingErrorCount	436
12.2.1.597SerialReceiveParityErrorCount	437
12.2.1.598SerialReceiveQueueClear	437
12.2.1.599SerialReceiveQueueCurrentCharacterCount	437
12.2.1.600SerialReceiveQueueMaxCharacterCount	437
12.2.1.601SerialTransmitQueueCurrentCharacterCount	437
12.2.1.602SerialTransmitQueueMaxCharacterCount	437
12.2.1.603Sharpening	437
12.2.1.604SharpeningAuto	437
12.2.1.605SharpeningEnable	438
12.2.1.606SharpeningThreshold	438
12.2.1.607SoftwareSignalPulse	438
12.2.1.608SoftwareSignalSelector	438
12.2.1.609SourceCount	438
12.2.1.610SourceSelector	438
12.2.1.611Test0001	438
12.2.1.612TestEventGenerate	438
12.2.1.613TestPattern	439
12.2.1.614TestPatternGeneratorSelector	439
12.2.1.615TestPendingAck	439
12.2.1.616TimerDelay	439
12.2.1.617TimerDuration	439
12.2.1.618TimerReset	439
12.2.1.619TimerSelector	439
12.2.1.620TimerStatus	439
12.2.1.621TimerTriggerActivation	440
12.2.1.622TimerTriggerSource	440

12.2.1.623TimerValue	440
12.2.1.624Timestamp	440
12.2.1.625TimestampLatch	440
12.2.1.626TimestampLatchValue	440
12.2.1.627TimestampReset	440
12.2.1.628TLParamsLocked	440
12.2.1.629TransferAbort	441
12.2.1.630TransferBlockCount	441
12.2.1.631TransferBurstCount	441
12.2.1.632TransferComponentSelector	441
12.2.1.633TransferControlMode	441
12.2.1.634TransferOperationMode	441
12.2.1.635TransferPause	441
12.2.1.636TransferQueueCurrentBlockCount	441
12.2.1.637TransferQueueMaxBlockCount	442
12.2.1.638TransferQueueMode	442
12.2.1.639TransferQueueOverflowCount	442
12.2.1.640TransferResume	442
12.2.1.641TransferSelector	442
12.2.1.642TransferStart	442
12.2.1.643TransferStatus	442
12.2.1.644TransferStatusSelector	442
12.2.1.645TransferStop	443
12.2.1.646TransferStreamChannel	443
12.2.1.647TransferTriggerActivation	443
12.2.1.648TransferTriggerMode	443
12.2.1.649TransferTriggerSelector	443
12.2.1.650TransferTriggerSource	443
12.2.1.651TriggerActivation	443
12.2.1.652TriggerDelay	443

12.2.1.653TriggerDivider	444
12.2.1.654TriggerEventTest	444
12.2.1.655TriggerMode	444
12.2.1.656TriggerMultiplier	444
12.2.1.657TriggerOverlap	444
12.2.1.658TriggerSelector	444
12.2.1.659TriggerSoftware	444
12.2.1.660TriggerSource	444
12.2.1.661UserOutputSelector	445
12.2.1.662UserOutputValue	445
12.2.1.663UserOutputValueAll	445
12.2.1.664UserOutputValueAllMask	445
12.2.1.665UserSetDefault	445
12.2.1.666UserSetFeatureEnable	445
12.2.1.667UserSetLoad	445
12.2.1.668UserSetSave	445
12.2.1.669UserSetSelector	446
12.2.1.670V3_3Enable	446
12.2.1.671WhiteClip	446
12.2.1.672WhiteClipSelector	446
12.2.1.673Width	446
12.2.1.674WidthMax	446
12.3 quickSpinTLDevice Struct Reference	447
12.3.1 Field Documentation	447
12.3.1.1 DeviceAccessStatus	448
12.3.1.2 DeviceCurrentSpeed	448
12.3.1.3 DeviceDisplayName	448
12.3.1.4 DeviceDriverVersion	448
12.3.1.5 DeviceEndianessMechanism	448
12.3.1.6 DeviceID	448

12.3.1.7 DeviceInstanceId	448
12.3.1.8 DevicesUpdater	448
12.3.1.9 DeviceLinkSpeed	449
12.3.1.10 DeviceLocation	449
12.3.1.11 DeviceModelName	449
12.3.1.12 DeviceMulticastMonitorMode	449
12.3.1.13 DevicePortId	449
12.3.1.14 DeviceSerialNumber	449
12.3.1.15 DeviceType	449
12.3.1.16 DeviceU3VProtocol	449
12.3.1.17 DeviceUserID	450
12.3.1.18 DeviceVendorName	450
12.3.1.19 DeviceVersion	450
12.3.1.20 GenICamXMLLocation	450
12.3.1.21 GenICamXMLPath	450
12.3.1.22 GevCCP	450
12.3.1.23 GevDeviceAutoForceIP	450
12.3.1.24 GevDeviceDiscoverMaximumPacketSize	450
12.3.1.25 GevDeviceForceGateway	451
12.3.1.26 GevDeviceForceIP	451
12.3.1.27 GevDeviceForceIPAddress	451
12.3.1.28 GevDeviceForceSubnetMask	451
12.3.1.29 GevDeviceGateway	451
12.3.1.30 GevDeviceIPAddress	451
12.3.1.31 GevDevicesWrongSubnet	451
12.3.1.32 GevDeviceMACAddress	451
12.3.1.33 GevDeviceMaximumPacketSize	452
12.3.1.34 GevDeviceMaximumRetryCount	452
12.3.1.35 GevDeviceModelsBigEndian	452
12.3.1.36 GevDevicePort	452

12.3.1.37	GevDeviceReadAndWriteTimeout	452
12.3.1.38	GevDeviceSubnetMask	452
12.3.1.39	GevVersionMajor	452
12.3.1.40	GevVersionMinor	452
12.3.1.41	GUIXMLLocation	453
12.3.1.42	GUIXMLPath	453
12.4	quickSpinTLInterface Struct Reference	453
12.4.1	Field Documentation	454
12.4.1.1	ActionCommand	454
12.4.1.2	DeviceAccessStatus	454
12.4.1.3	DeviceCount	454
12.4.1.4	DeviceID	454
12.4.1.5	DeviceModelName	455
12.4.1.6	DeviceSelector	455
12.4.1.7	DeviceSerialNumber	455
12.4.1.8	DeviceUnlock	455
12.4.1.9	DeviceUpdateList	455
12.4.1.10	DeviceVendorName	455
12.4.1.11	FilterDriverStatus	455
12.4.1.12	GevActionDeviceKey	455
12.4.1.13	GevActionGroupKey	456
12.4.1.14	GevActionGroupMask	456
12.4.1.15	GevActionTime	456
12.4.1.16	GevDeviceAutoForceIP	456
12.4.1.17	GevDeviceForceGateway	456
12.4.1.18	GevDeviceForceIP	456
12.4.1.19	GevDeviceForceIPAddress	456
12.4.1.20	GevDeviceForceSubnetMask	456
12.4.1.21	GevDeviceGateway	457
12.4.1.22	GevDeviceIPAddress	457

12.4.1.23	GevDeviceMACAddress	457
12.4.1.24	GevDeviceSubnetMask	457
12.4.1.25	GevInterfaceGateway	457
12.4.1.26	GevInterfaceGatewaySelector	457
12.4.1.27	GevInterfaceMACAddress	457
12.4.1.28	GevInterfaceMTU	457
12.4.1.29	GevInterfaceReceiveLinkSpeed	458
12.4.1.30	GevInterfaceSubnetIPAddress	458
12.4.1.31	GevInterfaceSubnetMask	458
12.4.1.32	GevInterfaceSubnetSelector	458
12.4.1.33	GevInterfaceTransmitLinkSpeed	458
12.4.1.34	HostAdapterDriverVersion	458
12.4.1.35	HostAdapterName	458
12.4.1.36	HostAdapterVendor	458
12.4.1.37	IncompatibleDeviceCount	459
12.4.1.38	IncompatibleDeviceID	459
12.4.1.39	IncompatibleDeviceModelName	459
12.4.1.40	IncompatibleDeviceSelector	459
12.4.1.41	IncompatibleDeviceVendorName	459
12.4.1.42	IncompatibleGevDeviceIPAddress	459
12.4.1.43	IncompatibleGevDeviceMACAddress	459
12.4.1.44	IncompatibleGevDeviceSubnetMask	459
12.4.1.45	InterfaceDisplayName	460
12.4.1.46	InterfaceID	460
12.4.1.47	InterfaceType	460
12.4.1.48	POEStatus	460
12.5	quickSpinTLStream Struct Reference	460
12.5.1	Field Documentation	461
12.5.1.1	GevFailedPacketCount	461
12.5.1.2	GevMaximumNumberResendRequests	461

12.5.1.3	GevPacketResendMode	461
12.5.1.4	GevPacketResendTimeout	461
12.5.1.5	GevResendPacketCount	461
12.5.1.6	GevResendRequestCount	461
12.5.1.7	GevTotalPacketCount	461
12.5.1.8	StreamAnnounceBufferMinimum	462
12.5.1.9	StreamAnnouncedBufferCount	462
12.5.1.10	StreamBlockTransferSize	462
12.5.1.11	StreamBufferAlignment	462
12.5.1.12	StreamBufferCountManual	462
12.5.1.13	StreamBufferCountMax	462
12.5.1.14	StreamBufferCountMode	462
12.5.1.15	StreamBufferCountResult	462
12.5.1.16	StreamBufferHandlingMode	463
12.5.1.17	StreamChunkCountMaximum	463
12.5.1.18	StreamCRCCheckEnable	463
12.5.1.19	StreamDeliveredFrameCount	463
12.5.1.20	StreamDroppedFrameCount	463
12.5.1.21	StreamFailedBufferCount	463
12.5.1.22	StreamID	463
12.5.1.23	StreamInputBufferCount	463
12.5.1.24	StreamIsGrabbing	464
12.5.1.25	StreamLostFrameCount	464
12.5.1.26	StreamOutputBufferCount	464
12.5.1.27	StreamStartedFrameCount	464
12.5.1.28	StreamType	464
12.6	quickSpinTLSystem Struct Reference	464
12.6.1	Field Documentation	465
12.6.1.1	EnumerateGEVInterfaces	465
12.6.1.2	EnumerateUSBInterfaces	465

12.6.1.3	GenTLSFNCVersionMajor	465
12.6.1.4	GenTLSFNCVersionMinor	465
12.6.1.5	GenTLSFNCVersionSubMinor	465
12.6.1.6	GenTLVersionMajor	466
12.6.1.7	GenTLVersionMinor	466
12.6.1.8	GevInterfaceDefaultGateway	466
12.6.1.9	GevInterfaceDefaultIPAddress	466
12.6.1.10	GevInterfaceDefaultSubnetMask	466
12.6.1.11	GevInterfaceMACAddress	466
12.6.1.12	GevVersionMajor	466
12.6.1.13	GevVersionMinor	466
12.6.1.14	InterfaceDisplayName	467
12.6.1.15	InterfaceID	467
12.6.1.16	InterfaceSelector	467
12.6.1.17	InterfaceUpdateList	467
12.6.1.18	TLDisplayName	467
12.6.1.19	TLFileName	467
12.6.1.20	TLID	467
12.6.1.21	TLModelName	467
12.6.1.22	TLPath	468
12.6.1.23	TLType	468
12.6.1.24	TLVendorName	468
12.6.1.25	TLVersion	468
12.7	spinAVIOption Struct Reference	468
12.7.1	Detailed Description	468
12.7.2	Field Documentation	469
12.7.2.1	frameRate	469
12.7.2.2	reserved	469
12.8	spinBMPOption Struct Reference	469
12.8.1	Detailed Description	469

12.8.2	Field Documentation	469
12.8.2.1	indexedColor_8bit	470
12.8.2.2	reserved	470
12.9	spinChunkData Struct Reference	470
12.9.1	Detailed Description	471
12.9.2	Field Documentation	471
12.9.2.1	m_blackLevel	471
12.9.2.2	m_compressionMode	471
12.9.2.3	m_compressionRatio	471
12.9.2.4	m_counterValue	471
12.9.2.5	m_cRC	472
12.9.2.6	m_encoderValue	472
12.9.2.7	m_exposureEndLineStatusAll	472
12.9.2.8	m_exposureTime	472
12.9.2.9	m_frameID	472
12.9.2.10	m_gain	472
12.9.2.11	m_height	472
12.9.2.12	m_image	472
12.9.2.13	m_inferenceConfidence	473
12.9.2.14	m_inferenceFrameId	473
12.9.2.15	m_inferenceResult	473
12.9.2.16	m_linePitch	473
12.9.2.17	m_lineStatusAll	473
12.9.2.18	m_offsetX	473
12.9.2.19	m_offsetY	473
12.9.2.20	m_partSelector	473
12.9.2.21	m_pixelDynamicRangeMax	474
12.9.2.22	m_pixelDynamicRangeMin	474
12.9.2.23	m_scan3dAxisMax	474
12.9.2.24	m_scan3dAxisMin	474

12.9.2.25 m_scan3dCoordinateOffset	474
12.9.2.26 m_scan3dCoordinateReferenceValue	474
12.9.2.27 m_scan3dCoordinateScale	474
12.9.2.28 m_scan3dInvalidDataValue	474
12.9.2.29 m_scan3dTransformValue	475
12.9.2.30 m_scanLineSelector	475
12.9.2.31 m_sequencerSetActive	475
12.9.2.32 m_serialDataLength	475
12.9.2.33 m_streamChannelID	475
12.9.2.34 m_timerValue	475
12.9.2.35 m_timestamp	475
12.9.2.36 m_timestampLatchValue	475
12.9.2.37 m_transferBlockID	476
12.9.2.38 m_transferQueueCurrentBlockCount	476
12.9.2.39 m_width	476
12.10 spinH264Option Struct Reference	476
12.10.1 Detailed Description	476
12.10.2 Field Documentation	476
12.10.2.1 bitrate	477
12.10.2.2 frameRate	477
12.10.2.3 height	477
12.10.2.4 reserved	477
12.10.2.5 width	477
12.11 spinJPEGOption Struct Reference	477
12.11.1 Detailed Description	478
12.11.2 Field Documentation	478
12.11.2.1 progressive	478
12.11.2.2 quality	478
12.11.2.3 reserved	478
12.12 spinJPG2Option Struct Reference	479

12.12.1 Detailed Description	479
12.12.2 Field Documentation	479
12.12.2.1 quality	479
12.12.2.2 reserved	479
12.13spinLibraryVersion Struct Reference	479
12.13.1 Detailed Description	480
12.13.2 Field Documentation	480
12.13.2.1 build	480
12.13.2.2 major	480
12.13.2.3 minor	480
12.13.2.4 type	481
12.14spinMJPGOption Struct Reference	481
12.14.1 Detailed Description	481
12.14.2 Field Documentation	481
12.14.2.1 frameRate	481
12.14.2.2 quality	481
12.14.2.3 reserved	482
12.15spinPGMOption Struct Reference	482
12.15.1 Detailed Description	482
12.15.2 Field Documentation	482
12.15.2.1 binaryFile	482
12.15.2.2 reserved	482
12.16spinPNGOption Struct Reference	483
12.16.1 Detailed Description	483
12.16.2 Field Documentation	483
12.16.2.1 compressionLevel	483
12.16.2.2 interlaced	483
12.16.2.3 reserved	483
12.17spinPPMOption Struct Reference	484
12.17.1 Detailed Description	484
12.17.2 Field Documentation	484
12.17.2.1 binaryFile	484
12.17.2.2 reserved	484
12.18spinTIFFOption Struct Reference	484
12.18.1 Detailed Description	485
12.18.2 Field Documentation	485
12.18.2.1 compression	485
12.18.2.2 reserved	485

13 File Documentation	487
13.1 doc/spindocs/C/GettingStarted.dox File Reference	487
13.2 doc/spindocs/C/ProgrammerGuide.dox File Reference	487
13.3 doc/spindocs/shared/Benefits.dox File Reference	487
13.4 doc/spindocs/shared/FlyCapture2Comparison.dox File Reference	487
13.5 doc/spindocs/shared/GenICamGenTL.dox File Reference	487
13.6 doc/spindocs/shared/Licensing.dox File Reference	487
13.7 doc/spindocs/shared/Maintenance.dox File Reference	487
13.8 include/spinc/CameraDefsC.h File Reference	487
13.9 include/spinc/ChunkDataDefC.h File Reference	520
13.10include/spinc/QuickSpinC.h File Reference	521
13.11include/spinc/QuickSpinDefsC.h File Reference	521
13.11.1 Typedef Documentation	522
13.11.1.1 quickSpinBooleanNode	522
13.11.1.2 quickSpinCommandNode	522
13.11.1.3 quickSpinEnumerationNode	522
13.11.1.4 quickSpinFloatNode	523
13.11.1.5 quickSpinIntegerNode	523
13.11.1.6 quickSpinRegisterNode	523
13.11.1.7 quickSpinStringNode	523
13.12include/spinc/SpinnakerC.h File Reference	523
13.12.1 Function Documentation	532
13.12.1.1 spinCameraForceIP()	532
13.13include/spinc/SpinnakerDefsC.h File Reference	533
13.14include/spinc/SpinnakerGenApiC.h File Reference	538
13.15include/spinc/SpinnakerGenApiDefsC.h File Reference	542
13.16include/spinc/SpinnakerPlatformC.h File Reference	545
13.16.1 Macro Definition Documentation	545
13.16.1.1 SPINNAKERC_API	545
13.17include/spinc/SpinVideoC.h File Reference	546
13.18include/spinc/TransportLayerDefsC.h File Reference	546
13.19include/spinc/TransportLayerDeviceC.h File Reference	548
13.20include/spinc/TransportLayerInterfaceC.h File Reference	549
13.21include/spinc/TransportLayerStreamC.h File Reference	550
13.22include/spinc/TransportLayerSystemC.h File Reference	550
Index	553

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](#)

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 $(\text{bufferSize} + 1024 - 1) / 1024 * 1024$ 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-solutions>

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 (CPO-OL)
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 (CPO-OL) 1.02
FreeImage	FreeImage public license
Boost	Boost Software License
Libusb	GPLv2.1 License
Libraw1394	GPLv2.0 License
FFMPEG	GPLv2.1 License
log4Net	Apache license 2.0
log4Cpp	GPL 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-machi>

Chapter 8

Module Index

8.1 Modules

Here is a list of all modules:

Spinnaker C QuickSpin API	140
QuickSpin Access	141
Transport Layer Enumerations	336
TLDevice Structures	345
TLInterface Structures	346
TLStream Structures	347
TLSystem Structures	348
Spinnaker C API	143
Spinnaker C Definitions	21
Camera Enumerations	23
Chunk Data Structures	139
Spinnaker C Handles	249
Spinnaker C Function Signatures	253
Spinnaker C Enumerations	255
Spinnaker C Structures	263
Error Handling	145
System Access	150
InterfaceList Access	164
CameraList Access	168
Interface Access	174
Camera Access	182
SpinVideo Recording Access	333
Image Access	194
Event Access	224
ImageStatistics Access	231
Logging Event Data Access	240
Device Event Data Access	245
Chunk data access	248
Spinnaker C GenICam API	265
Node Map Access	268
Node Access	272
IValue Access	284
String Access	287
Integer Access	291

IFloat Access	296
IEnumeration Access	301
IEnumEntry Access	306
IBoolean Access	309
ICommand Access	311
ICategory Access	313
IRegister Access	315
Spinnaker C GenICam Handles	320
Spinnaker C GenICam Enumerations	322

Chapter 9

Data Structure Index

9.1 Data Structures

Here are the data structures with brief descriptions:

actionCommandResult	
Action Command Result	349
quickSpin	350
quickSpinTLDevice	447
quickSpinTLInterface	453
quickSpinTLStream	460
quickSpinTLSystem	464
spinAVIOption	
Options for saving uncompressed videos	468
spinBMPOption	
Options for saving BMP images	469
spinChunkData	
The type of information that can be obtained from image chunk data	470
spinH264Option	
Options for saving H264 videos	476
spinJPEGOption	
Options for saving JPEG images	477
spinJPG2Option	
Options for saving JPEG 2000 images	479
spinLibraryVersion	
Provides easier access to the current version of Spinnaker	479
spinMJPGOption	
Options for saving MJPG videos	481
spinPGMOption	
Options for saving PGM images	482
spinPNGOption	
Options for saving PNG images	483
spinPPMOption	
Options for saving PPM images	484
spinTIFFOption	
Options for saving TIFF images	484

Chapter 10

File Index

10.1 File List

Here is a list of all files with brief descriptions:

include/spinc/ CameraDefsC.h	487
include/spinc/ ChunkDataDefC.h	520
include/spinc/ QuickSpinC.h	521
include/spinc/ QuickSpinDefsC.h	521
include/spinc/ SpinnakerC.h	523
include/spinc/ SpinnakerDefsC.h	533
include/spinc/ SpinnakerGenApiC.h	538
include/spinc/ SpinnakerGenApiDefsC.h	542
include/spinc/ SpinnakerPlatformC.h	545
include/spinc/ SpinVideoC.h	546
include/spinc/ TransportLayerDefsC.h	546
include/spinc/ TransportLayerDeviceC.h	548
include/spinc/ TransportLayerInterfaceC.h	549
include/spinc/ TransportLayerStreamC.h	550
include/spinc/ TransportLayerSystemC.h	550

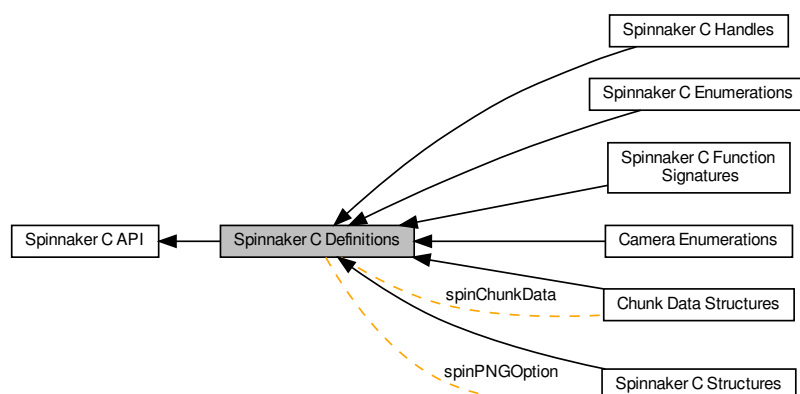
Chapter 11

Module Documentation

11.1 Spinnaker C Definitions

Definitions for Spinnaker C.

Collaboration diagram for Spinnaker C Definitions:



Modules

- [Camera Enumerations](#)
- [Chunk Data Structures](#)
- [Spinnaker C Handles](#)
Spinnaker C handle definitions.
- [Spinnaker C Function Signatures](#)
Spinnaker C function signature definitions.
- [Spinnaker C Enumerations](#)
Spinnaker C enumeration definitions.
- [Spinnaker C Structures](#)
Spinnaker C structure definitions.

Data Structures

- struct [spinChunkData](#)
The type of information that can be obtained from image chunk data.
- struct [spinPNGOption](#)
Options for saving PNG images.

Typedefs

- typedef uint8_t [bool8_t](#)

Variables

- static const [bool8_t](#) [False](#) = 0
- static const [bool8_t](#) [True](#) = 1

11.1.1 Detailed Description

Definitions for Spinnaker C.

Definitions for Spinnaker C API.

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

11.1.2 Typedef Documentation

11.1.2.1 bool8_t

```
typedef uint8_t bool8_t
```

11.1.3 Variable Documentation

11.1.3.1 False

```
const bool8_t False = 0 [static]
```

11.1.3.2 True

```
const bool8_t True = 1 [static]
```

11.2 Camera Enumerations

Collaboration diagram for Camera Enumerations:



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_DEVICEREGISTERSENDIANNES }

• 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_DEVICE TLTYPE }

• enum spinDevicePowerSupplySelectorEnums {
DevicePowerSupplySelector_External,
NUM_DEVICEPOWERSUPPLYSELECTOR }

• enum spinDeviceTemperatureSelectorEnums {
DeviceTemperatureSelector_Sensor,
NUM_DEVICE TEMPERATURESELECTOR }

• enum spinDeviceIndicatorModeEnums {
DeviceIndicatorMode_Inactive,
DeviceIndicatorMode_Active,
DeviceIndicatorMode_ErrorStatus,
NUM_DEVICE INDICATORMODE }

• enum spinAutoExposureControlPriorityEnums {
AutoExposureControlPriority_Gain,
AutoExposureControlPriority_ExposureTime,
NUM_AUTOEXPOSURECONTROLPRIORITY }

• enum spinAutoExposureMeteringModeEnums {
AutoExposureMeteringMode_Average,
AutoExposureMeteringMode_Spot,
AutoExposureMeteringMode_Partial,
AutoExposureMeteringMode_CenterWeighted,
AutoExposureMeteringMode_HistogramPeak,
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,
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_Deglintch,
    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_ENCODERRESETEACTIVATION }

• 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 }

```

11.2.1 Detailed Description

11.2.2 Enumeration Type Documentation

11.2.2.1 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	

11.2.2.2 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_ACQUISITIONSTATUSSELECTION	

11.2.2.3 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	

11.2.2.4 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	

11.2.2.5 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	

11.2.2.6 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	

11.2.2.7 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	

11.2.2.8 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_HistogramPeak	
NUM_AUTOEXPOSUREMETERINGMODE	

11.2.2.9 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	

11.2.2.10 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	

11.2.2.11 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	

11.2.2.12 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	

11.2.2.13 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	

11.2.2.14 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	

11.2.2.15 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	

11.2.2.16 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	

11.2.2.17 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	

11.2.2.18 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	

11.2.2.19 spinChunkBlackLevelSelectorEnums

```
enum spinChunkBlackLevelSelectorEnums
```

< Selects which black level to retrieve

Enumerator

ChunkBlackLevelSelector_All	
NUM_CHUNKBLACKLEVELSELECTOR	

11.2.2.20 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	

11.2.2.21 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	

11.2.2.22 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	

11.2.2.23 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	

11.2.2.24 spinChunkGainSelectorEnums

```
enum spinChunkGainSelectorEnums
```

< Selects which gain to retrieve

Enumerator

ChunkGainSelector_All	
ChunkGainSelector_Red	
ChunkGainSelector_Green	
ChunkGainSelector_Blue	
NUM_CHUNKGAINSELECTOR	

11.2.2.25 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.
ChunkImageComponent_Confidence	The image data is the confidence map component.
ChunkImageComponent_Scatter	The image data is the scatter component.
NUM_CHUNKIMAGECOMPONENT	

11.2.2.26 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	

11.2.2.27 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	

11.2.2.28 spinChunkScan3dCoordinateReferenceSelectorEnums

enum `spinChunkScan3dCoordinateReferenceSelectorEnums`

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

Enumerator

<code>ChunkScan3dCoordinateReferenceSelector_RotationX</code>	Rotation around X axis.
<code>ChunkScan3dCoordinateReferenceSelector_RotationY</code>	Rotation around Y axis.
<code>ChunkScan3dCoordinateReferenceSelector_RotationZ</code>	Rotation around Z axis.
<code>ChunkScan3dCoordinateReferenceSelector_TranslationX</code>	X axis translation.
<code>ChunkScan3dCoordinateReferenceSelector_TranslationY</code>	Y axis translation.
<code>ChunkScan3dCoordinateReferenceSelector_TranslationZ</code>	Z axis translation.
<code>NUM_CHUNKSCAN3DCOORDINATEREFERENCESELECTOR</code>	

11.2.2.29 spinChunkScan3dCoordinateSelectorEnums

enum `spinChunkScan3dCoordinateSelectorEnums`

< Selects which Coordinate to retrieve data from.

Enumerator

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

11.2.2.30 spinChunkScan3dCoordinateSystemEnums

enum `spinChunkScan3dCoordinateSystemEnums`

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

Enumerator

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

11.2.2.31 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_CHUNKSCAN3DCOORDINATESYSTEMREF↔ ERENCE	

11.2.2.32 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	

11.2.2.33 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	

11.2.2.34 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	

11.2.2.35 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	

11.2.2.36 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	

11.2.2.37 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	

11.2.2.38 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	

11.2.2.39 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	

11.2.2.40 spinCITimeSlotsCountEnums

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

CITimeSlotsCount_One	One
CITimeSlotsCount_Two	Two
CITimeSlotsCount_Three	Three
NUM_CLTIMESLOTSCOUNT	

11.2.2.41 spinColorTransformationSelectorEnums

enum `spinColorTransformationSelectorEnums`

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

Enumerator

ColorTransformationSelector_RGBtoRGB	
ColorTransformationSelector_RGBtoYUV	
NUM_COLORTRANSFORMATIONSELECTOR	

11.2.2.42 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	

11.2.2.43 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	

11.2.2.44 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	

11.2.2.45 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	

11.2.2.46 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	

11.2.2.47 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	

11.2.2.48 spinCounterSelectorEnums

```
enum spinCounterSelectorEnums
```

< Selects which counter to configure

Enumerator

CounterSelector_Counter0	
CounterSelector_Counter1	
NUM_COUNTERSELECTOR	

11.2.2.49 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	

11.2.2.50 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	

11.2.2.51 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	

11.2.2.52 spinCxpConnectionTestModeEnums

enum `spinCxpConnectionTestModeEnums`

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

Enumerator

<code>CxpConnectionTestMode_Off</code>	Off
<code>CxpConnectionTestMode_Mode1</code>	Mode 1
<code>NUM_CXP_CONNECTIONTESTMODE</code>	

11.2.2.53 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

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

11.2.2.54 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	

11.2.2.55 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	

11.2.2.56 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	

11.2.2.57 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	

11.2.2.58 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	

11.2.2.59 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	

11.2.2.60 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	

11.2.2.61 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	

11.2.2.62 spinDeviceCharacterSetEnums

enum `spinDeviceCharacterSetEnums`

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

Enumerator

DeviceCharacterSet_UTF8	
DeviceCharacterSet_ASCII	
NUM_DEVICECHARACTERSET	

11.2.2.63 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	

11.2.2.64 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	

11.2.2.65 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	

11.2.2.66 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	

11.2.2.67 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	

11.2.2.68 spinDevicePowerSupplySelectorEnums

enum `spinDevicePowerSupplySelectorEnums`

< Selects the power supply source to control or read.

Enumerator

DevicePowerSupplySelector_External	
NUM_DEVICEPOWERSUPPLYSELECTOR	

11.2.2.69 spinDeviceRegistersEndiannessEnums

enum `spinDeviceRegistersEndiannessEnums`

< Endianness of the registers of the device.

Enumerator

DeviceRegistersEndianness_Little	
DeviceRegistersEndianness_Big	
NUM_DEVICEREGISTERSENDIANNES	

11.2.2.70 spinDeviceScanTypeEnums

enum `spinDeviceScanTypeEnums`

< Scan type of the sensor of the device.

Enumerator

DeviceScanType_Areascan	
NUM_DEVICEASCANTYPE	

11.2.2.71 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	

11.2.2.72 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	

11.2.2.73 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_DEVICESTREAMCHANNELENDIANNES	

11.2.2.74 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	

11.2.2.75 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	

11.2.2.76 spinDeviceTemperatureSelectorEnums

enum [spinDeviceTemperatureSelectorEnums](#)

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

Enumerator

DeviceTemperatureSelector_Sensor	
NUM_DEVICETEMPERATURESELECTOR	

11.2.2.77 spinDeviceTLTypeEnums

enum [spinDeviceTLTypeEnums](#)

< Transport Layer type of the device.

Enumerator

DeviceTLType_GigEVision	
DeviceTLType_CameraLink	
DeviceTLType_CameraLinkHS	
DeviceTLType_CoaXPress	
DeviceTLType_USB3Vision	
DeviceTLType_Custom	
NUM_DEVICETLTYPE	

11.2.2.78 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	

11.2.2.79 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	

11.2.2.80 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	

11.2.2.81 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	

11.2.2.82 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	

11.2.2.83 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	

11.2.2.84 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	

11.2.2.85 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	

11.2.2.86 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	

11.2.2.87 spinEventNotificationEnums

enum `spinEventNotificationEnums`

< Enables/Disables the selected event.

Enumerator

EventNotification_On	
EventNotification_Off	
NUM_EVENTNOTIFICATION	

11.2.2.88 spinEventSelectorEnums

enum `spinEventSelectorEnums`

< Selects which Event to enable or disable.

Enumerator

EventSelector_Error	
EventSelector_ExposureEnd	
EventSelector_SerialPortReceive	
NUM_EVENTSELECTOR	

11.2.2.89 spinExposureActiveModeEnums

enum `spinExposureActiveModeEnums`

< Control sensor active exposure mode.

Enumerator

ExposureActiveMode_Line1	
ExposureActiveMode_AnyPixels	
ExposureActiveMode_AllPixels	
NUM_EXPOSUREACTIVEMODE	

11.2.2.90 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	

11.2.2.91 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	

11.2.2.92 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	

11.2.2.93 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	

11.2.2.94 spinFileOpenModeEnums

enum `spinFileOpenModeEnums`

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

Enumerator

FileOpenMode_Read	
FileOpenMode_Write	
FileOpenMode_ReadWrite	
NUM_FILEOPENMODE	

11.2.2.95 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	

11.2.2.96 spinFileOperationStatusEnums

```
enum spinFileOperationStatusEnums
```

< Represents the file operation execution status.

Enumerator

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

11.2.2.97 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	

11.2.2.98 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	

11.2.2.99 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	

11.2.2.100 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	

11.2.2.101 spinGevCCPEnums

```
enum spinGevCCPEnums
```

< Controls the device access privilege of an application.

Enumerator

GevCCP_OpenAccess	
GevCCP_ExclusiveAccess	
GevCCP_ControlAccess	
NUM_GEVCCP	

11.2.2.102 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	

11.2.2.103 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	

11.2.2.104 spinGevGVSPExtendedIDModeEnums

```
enum spinGevGVSPExtendedIDModeEnums
```

< Enables the extended IDs mode.

Enumerator

GevGVSPExtendedIDMode_Off	Off
GevGVSPExtendedIDMode_On	On
NUM_GEVGVSPEXTENDEDIDMODE	

11.2.2.105 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	

11.2.2.106 spinGevIEEE1588ModeEnums

enum [spinGevIEEE1588ModeEnums](#)

< Provides the mode of the IEEE 1588 clock.

Enumerator

GevIEEE1588Mode_Auto	Automatic
GevIEEE1588Mode_SlaveOnly	Slave Only
NUM_GEVIEEE1588MODE	

11.2.2.107 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	

11.2.2.108 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	

11.2.2.109 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	

11.2.2.110 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	

11.2.2.111 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	

11.2.2.112 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_IMAGECOMPRESSIONJPEGFORMATOPT↔ ION	

11.2.2.113 spinImageCompressionModeEnums

enum `spinImageCompressionModeEnums`

<

Enumerator

ImageCompressionMode_Off	
ImageCompressionMode_Lossless	
NUM_IMAGECOMPRESSIONMODE	

11.2.2.114 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	

11.2.2.115 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	

11.2.2.116 spinLineInputFilterSelectorEnums

enum `spinLineInputFilterSelectorEnums`

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

Enumerator

LineInputFilterSelector_Deglitch	
LineInputFilterSelector_Debounce	
NUM_LINEINPUTFILTERSELECTOR	

11.2.2.117 spinLineModeEnums

```
enum spinLineModeEnums
```

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

Enumerator

LineMode_Input	
LineMode_Output	
NUM_LINEMODE	

11.2.2.118 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	

11.2.2.119 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	

11.2.2.120 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	

11.2.2.121 spinLogicBlockLUTInputSelectorEnums

```
enum spinLogicBlockLUTInputSelectorEnums
```

< Controls which LogicBlockLUT Input Source & Activation to access.

Enumerator

LogicBlockLUTInputSelector_Input0	
LogicBlockLUTInputSelector_Input1	
LogicBlockLUTInputSelector_Input2	
LogicBlockLUTInputSelector_Input3	
NUM_LOGICBLOCKLUTINPUTSELECTOR	

11.2.2.122 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	

11.2.2.123 spinLogicBlockLUTSelectorEnums

enum [spinLogicBlockLUTSelectorEnums](#)

< Selects which LogicBlock LUT to configure

Enumerator

LogicBlockLUTSelector_Value	
LogicBlockLUTSelector_Enable	
NUM_LOGICBLOCKLUTSELECTOR	

11.2.2.124 spinLogicBlockSelectorEnums

```
enum spinLogicBlockSelectorEnums
```

< Selects which LogicBlock to configure

Enumerator

LogicBlockSelector_LogicBlock0	
LogicBlockSelector_LogicBlock1	
NUM_LOGICBLOCKSELECTOR	

11.2.2.125 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	

11.2.2.126 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	

11.2.2.127 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
PixelFormat_RGB565p	Red-Green-Blue 5/6/5-bit packed
PixelFormat_BGRa10	Blue-Green-Red-alpha 10-bit unpacked

Enumerator

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
PixelFormat_Coord3D_A8	3D coordinate A 8-bit
PixelFormat_Coord3D_A10p	3D coordinate A 10-bit packed

Enumerator

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
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

Enumerator

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
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

Enumerator

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.
UNKNOWN_PIXELFORMAT	
NUM_PIXELFORMAT	

11.2.2.128 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

Enumerator

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
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

Enumerator

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
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

Enumerator

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
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

Enumerator

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
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

Enumerator

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_CbYY↔CrYY	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_CbY↔CrY	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_CbY↔CrY	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_CbYY↔CrYY	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_CbY↔CrY	YCbCr 4:2:2 10-bit unpacked BT.709
PixelFormatInfoSelector_YCbCr709_422_10p	YCbCr 4:2:2 10-bit packed BT.709
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_CbY↔CrY	YCbCr 4:2:2 12-bit unpacked BT.709

Enumerator

PixelFormatInfoSelector_YCbCr709_422_12p	YCbCr 4:2:2 12-bit packed BT.709
PixelFormatInfoSelector_YCbCr709_422_12p_CbCrY	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	

11.2.2.129 spinPixelFormatSizeEnums

```
enum spinPixelFormatSizeEnums
```

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

Enumerator

PixelFormatSize_Bpp1	1 bit per pixel.
PixelFormatSize_Bpp2	2 bits per pixel.
PixelFormatSize_Bpp4	4 bits per pixel.
PixelFormatSize_Bpp8	8 bits per pixel.
PixelFormatSize_Bpp10	10 bits per pixel.
PixelFormatSize_Bpp12	12 bits per pixel.
PixelFormatSize_Bpp14	14 bits per pixel.
PixelFormatSize_Bpp16	16 bits per pixel.
PixelFormatSize_Bpp20	20 bits per pixel.
PixelFormatSize_Bpp24	24 bits per pixel.
PixelFormatSize_Bpp30	30 bits per pixel.
PixelFormatSize_Bpp32	32 bits per pixel.
PixelFormatSize_Bpp36	36 bits per pixel.
PixelFormatSize_Bpp48	48 bits per pixel.
PixelFormatSize_Bpp64	64 bits per pixel.
PixelFormatSize_Bpp96	96 bits per pixel.
NUM_PIXELSIZE	

11.2.2.130 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	

11.2.2.131 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	

11.2.2.132 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	

11.2.2.133 spinRgbTransformLightSourceEnums

enum `spinRgbTransformLightSourceEnums`

< Used to select from a set of RGBtoRGB transform matrices 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

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

11.2.2.134 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

<code>Scan3dCoordinateReferenceSelector_RotationX</code>	Rotation around X axis.
<code>Scan3dCoordinateReferenceSelector_RotationY</code>	Rotation around Y axis.
<code>Scan3dCoordinateReferenceSelector_RotationZ</code>	Rotation around Z axis.
<code>Scan3dCoordinateReferenceSelector_TranslationX</code>	X axis translation.
<code>Scan3dCoordinateReferenceSelector_TranslationY</code>	Y axis translation.
<code>Scan3dCoordinateReferenceSelector_TranslationZ</code>	Z axis translation.
<code>NUM_SCAN3DCOORDINATEREFERENCESELECTOR</code>	

11.2.2.135 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	

11.2.2.136 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	

11.2.2.137 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	

11.2.2.138 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	

11.2.2.139 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	

11.2.2.140 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.

Enumerator

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	

11.2.2.141 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.
SensorDigitizationTaps_Four	4 taps.
SensorDigitizationTaps_Eight	8 taps.
SensorDigitizationTaps_Ten	10 taps.
NUM_SENSORDIGITIZATIONTAPS	

11.2.2.142 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	

11.2.2.143 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	

11.2.2.144 spinSequencerConfigurationModeEnums

enum [spinSequencerConfigurationModeEnums](#)

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

Enumerator

SequencerConfigurationMode_Off	
SequencerConfigurationMode_On	
NUM_SEQUENCERCONFIGURATIONMODE	

11.2.2.145 spinSequencerConfigurationValidEnums

enum `spinSequencerConfigurationValidEnums`

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

Enumerator

SequencerConfigurationValid_No	
SequencerConfigurationValid_Yes	
NUM_SEQUENCERCONFIGURATIONVALID	

11.2.2.146 spinSequencerModeEnums

enum `spinSequencerModeEnums`

< Controls whether or not a sequencer is active.

Enumerator

SequencerMode_Off	
SequencerMode_On	
NUM_SEQUENCERMODE	

11.2.2.147 spinSequencerSetValidEnums

enum `spinSequencerSetValidEnums`

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

Enumerator

SequencerSetValid_No	
SequencerSetValid_Yes	
NUM_SEQUENCERSETVALID	

11.2.2.148 spinSequencerTriggerActivationEnums

enum `spinSequencerTriggerActivationEnums`

< Specifies the activation mode of the sequencer trigger.

Enumerator

SequencerTriggerActivation_RisingEdge	
SequencerTriggerActivation_FallingEdge	
SequencerTriggerActivation_AnyEdge	
SequencerTriggerActivation_LevelHigh	
SequencerTriggerActivation_LevelLow	
NUM_SEQUENCERTRIGGERACTIVATION	

11.2.2.149 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	

11.2.2.150 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	

11.2.2.151 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	

11.2.2.152 spinSerialPortSelectorEnums

enum `spinSerialPortSelectorEnums`

< Selects which serial port of the device to control.

Enumerator

SerialPortSelector_SerialPort0	
NUM_SERIALPORTSELECTOR	

11.2.2.153 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	

11.2.2.154 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	

11.2.2.155 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	

11.2.2.156 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	

11.2.2.157 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	

11.2.2.158 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	

11.2.2.159 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	

11.2.2.160 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	

11.2.2.161 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	

11.2.2.162 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.

Enumerator

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 specified TimerTriggerActivation condition is met on the chosen I/O Line.
TimerTriggerSource_Line1	Starts when the specified TimerTriggerActivation condition is met on the chosen I/O Line.
TimerTriggerSource_Line2	Starts when the specified 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.
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	

11.2.2.163 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	

11.2.2.164 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	

11.2.2.165 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	

11.2.2.166 spinTransferQueueModeEnums

enum `spinTransferQueueModeEnums`

< Specifies the operation mode of the transfer queue.

Enumerator

TransferQueueMode_FirstInFirstOut	Blocks first In are transferred Out first.
NUM_TRANSFERQUEUEMODE	

11.2.2.167 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	

11.2.2.168 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.

Enumerator

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	

11.2.2.169 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	

11.2.2.170 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	

11.2.2.171 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	

11.2.2.172 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.
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.

Enumerator

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	

11.2.2.173 spinTriggerActivationEnums

```
enum spinTriggerActivationEnums
```

< Specifies the activation mode of the trigger.

Enumerator

TriggerActivation_LevelLow	
TriggerActivation_LevelHigh	
TriggerActivation_FallingEdge	
TriggerActivation_RisingEdge	
TriggerActivation_AnyEdge	
NUM_TRIGGERACTIVATION	

11.2.2.174 spinTriggerModeEnums

```
enum spinTriggerModeEnums
```

< Controls whether or not trigger is active.

Enumerator

TriggerMode_Off	
TriggerMode_On	
NUM_TRIGGERMODE	

11.2.2.175 spinTriggerOverlapEnums

enum `spinTriggerOverlapEnums`

< Specifies the overlap mode of the trigger.

Enumerator

TriggerOverlap_Off	
TriggerOverlap_ReadOut	
TriggerOverlap_PreviousFrame	
NUM_TRIGGEROVERLAP	

11.2.2.176 spinTriggerSelectorEnums

enum `spinTriggerSelectorEnums`

< Selects the type of trigger to configure.

Enumerator

TriggerSelector_AcquisitionStart	
TriggerSelector_FrameStart	
TriggerSelector_FrameBurstStart	
NUM_TRIGGERSELECTOR	

11.2.2.177 spinTriggerSourceEnums

enum `spinTriggerSourceEnums`

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

Enumerator

TriggerSource_Software	
------------------------	--

Enumerator

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	

11.2.2.178 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	

11.2.2.179 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	

11.2.2.180 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	

11.2.2.181 spinWhiteClipSelectorEnums

enum `spinWhiteClipSelectorEnums`

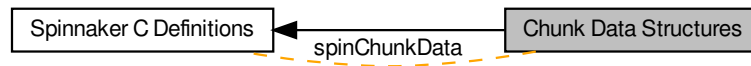
< Selects which White Clip to control.

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	

11.3 Chunk Data Structures

Collaboration diagram for Chunk Data Structures:



Data Structures

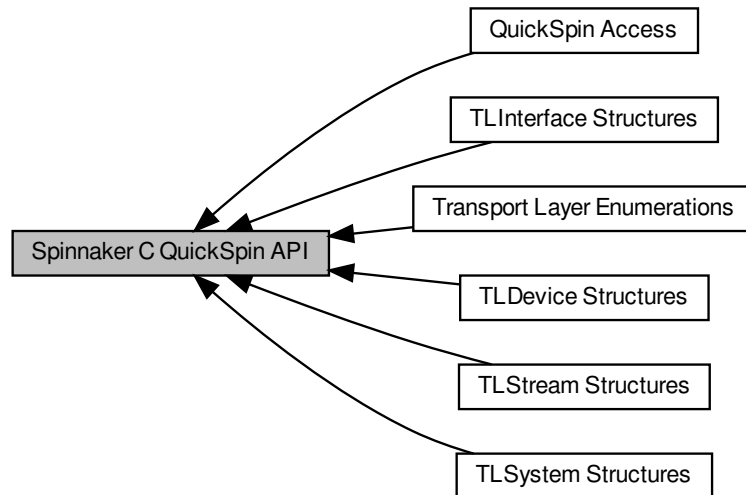
- struct [spinChunkData](#)

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

11.3.1 Detailed Description

11.4 Spinnaker C QuickSpin API

Collaboration diagram for Spinnaker C QuickSpin API:



Modules

- [QuickSpin Access](#)

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

- [Transport Layer Enumerations](#)
- [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.

Collaboration diagram for QuickSpin Access:



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](#) *pQuickSpinTLDevice)
- [SPINNAKERC_API quickSpinTLStreamInit](#) ([spinCamera](#) hCamera, [quickSpinTLStream](#) *pQuickSpinTLStream)
- [SPINNAKERC_API quickSpinTLInterfaceInit](#) ([spinInterface](#) hInterface, [quickSpinTLInterface](#) *pQuickSpinTLInterface)
- [SPINNAKERC_API quickSpinTLSystemInit](#) ([spinSystem](#) hSystem, [quickSpinTLSystem](#) *pQuickSpinTLSystem)

11.5.1 Detailed Description

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

11.5.2 Function Documentation

11.5.2.1 quickSpinInit()

```

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

11.5.2.2 quickSpinInitEx()

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

11.5.2.3 quickSpinTLDeviceInit()

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

11.5.2.4 quickSpinTLInterfaceInit()

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

11.5.2.5 quickSpinTLStreamInit()

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

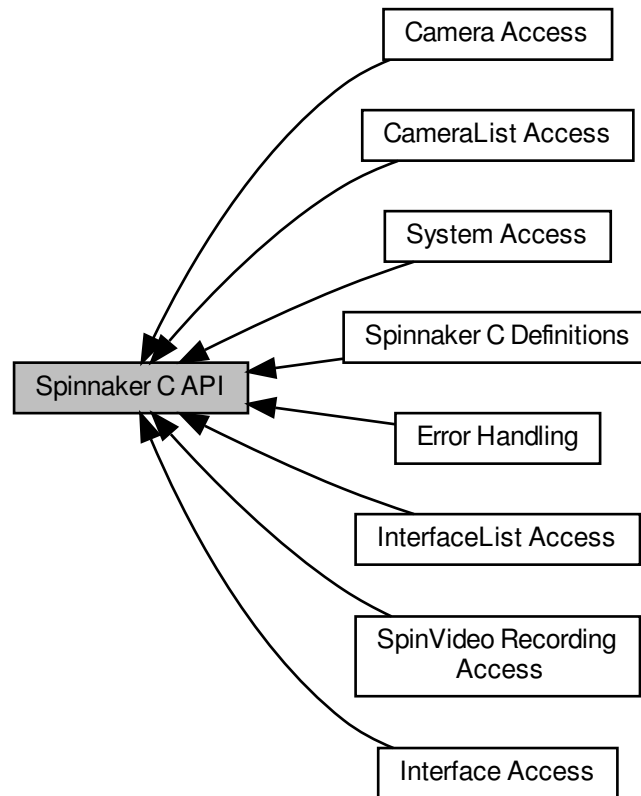
11.5.2.6 quickSpinTLSystemInit()

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

11.6 Spinnaker C API

SpinnakerPlatform C Include.

Collaboration diagram for Spinnaker C API:



Modules

- [Spinnaker C Definitions](#)
Definitions for Spinnaker C.
- [Error Handling](#)
The functions in this section provide access to additional information related to error returns.
- [System Access](#)
The functions in this section provide access to information, objects, and functionality of the system object.
- [InterfaceList Access](#)
The functions in this section provide access to information, objects, and functionality of interface lists.
- [CameraList Access](#)
The functions in this section provide access to information, objects, and functionality of camera lists.
- [Interface Access](#)
The functions in this section provide access to information, objects, and functionality of interfaces.
- [Camera Access](#)

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

- [SpinVideo Recording Access](#)

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

Functions

- [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.

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

11.6.2 Function Documentation

11.6.2.1 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

<i>hCamera</i>	The camera to check
<i>pMaxPacketSize</i>	The maximum packet size returned

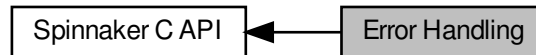
Returns

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

11.7 Error Handling

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

Collaboration diagram for Error Handling:



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.

11.7.1 Detailed Description

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

11.7.2 Function Documentation

11.7.2.1 spinErrorGetLast()

```
SPINNAKERC_API spinErrorGetLast (
    spinError * pError )
```

Retrieves the error code of the last error.

See also

[spinError](#)

Parameters

<i>pError</i>	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

11.7.2.2 spinErrorGetLastBuildDate()

```
SPINNAKERC_API spinErrorGetLastBuildDate (
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the build date of the last error.

See also

[spinError](#)

Parameters

<i>pBuf</i>	The c-string character buffer in which the build date is returned
<i>pBufLen</i>	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

11.7.2.3 spinErrorGetLastBuildTime()

```
SPINNAKERC_API spinErrorGetLastBuildTime (
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the build time of the last error.

See also

[spinError](#)

Parameters

<i>pBuf</i>	The c-string character buffer in which the build time is returned
<i>pBufLen</i>	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

11.7.2.4 spinErrorGetLastFileName()

```
SPINNAKERC_API spinErrorGetLastFileName (
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the filename of the last error.

See also

[spinError](#)

Parameters

<i>pBuf</i>	The c-string character buffer in which the file name is returned
<i>pBufLen</i>	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

11.7.2.5 spinErrorGetLastFullMessage()

```
SPINNAKERC_API spinErrorGetLastFullMessage (
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the full error message of the last error.

See also

[spinError](#)

Parameters

<i>pBuf</i>	The c-string character buffer in which the full error message is returned
<i>pBufLen</i>	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

11.7.2.6 spinErrorGetLastFunctionName()

```
SPINNAKERC_API spinErrorGetLastFunctionName (
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the function name of the last error.

See also

[spinError](#)

Parameters

<i>pBuf</i>	The c-string character buffer in which the function name is returned
<i>pBufLen</i>	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

11.7.2.7 spinErrorGetLastLineNumber()

```
SPINNAKERC_API spinErrorGetLastLineNumber (
    int64_t * pLineNum )
```

Retrieves the line number of the last error.

See also

[spinError](#)

Parameters

<i>pBuf</i>	The c-string character buffer in which the line number is returned
<i>pBufLen</i>	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

11.7.2.8 `spinErrorGetLastMessage()`

```
SPINNAKERC_API spinErrorGetLastMessage (
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the error message of the last error.

See also

[spinError](#)

Parameters

<i>pBuf</i>	The c-string character buffer in which the error message is returned
<i>pBufLen</i>	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

11.8 System Access

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

Collaboration diagram for System Access:



Functions

- [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 \(spinSystem hSystem, spinLogEventHandler hLogEventHandler\)](#)
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, spinDevice↵ArrivalEventHandler hDeviceArrivalEventHandler)

Unregisters a device arrival event handler from the system.

- **SPINNAKERC_API spinSystemUnregisterDeviceRemovalEventHandler** (spinSystem hSystem, spinDevice↵RemovalEventHandler hDeviceRemovalEventHandler)

Unregisters a device removal event handler from the system.

- **SPINNAKERC_API spinSystemRegisterInterfaceEventHandler** (spinSystem hSystem, spinInterfaceEvent↵Handler 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 spinSystem↵RegisterInterfaceEventHandler() is called, those interfaces will be automatically registered with this event.

- **SPINNAKERC_API spinSystemUnregisterInterfaceEventHandler** (spinSystem hSystem, spinInterface↵EventHandler 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.

11.8.1 Detailed Description

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.

11.8.2 Function Documentation

11.8.2.1 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

<i>hSystem</i>	The system, from which the camera list is retrieved
<i>hCameraList</i>	The camera list to house the cameras from the system

Returns

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

11.8.2.2 `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

<i>hSystem</i>	The system, from which the camera list is retrieved
<i>bUpdateInterfaces</i>	The boolean of whether to update the interface list
<i>bUpdateCameras</i>	The boolean of whether to update the camera list
<i>hCameraList</i>	The camera list to house the cameras from the system

Returns

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

11.8.2.3 `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

<i>phSystem</i>	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

11.8.2.4 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

<i>hSystem</i>	The system, from which the interface list is retrieved
<i>hInterfaceList</i>	The interface list to house the interfaces from the system

Returns

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

11.8.2.5 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).

11.8.2.6 spinSystemGetLoggingLevel()

```
SPINNAKERC_API spinSystemGetLoggingLevel (
    spinSystem hSystem,
    spinnakerLogLevel * pLogLevel )
```

Retrieves the logging level for all logging events on the system.

See also

[spinError](#)

Parameters

<i>hSystem</i>	The system, from which the logging level is retrieved
<i>logLevel</i>	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

11.8.2.7 spinSystemGetTLNodeMap()

```
SPINNAKERC_API spinSystemGetTLNodeMap (
    spinSystem hSystem,
    spinNodeMapHandle * phNodeMap )
```

Retrieves the transport layer nodemap from the system.

See also

[spinError](#)

Parameters

<i>hSystem</i>	The system handle.
<i>phNodeMap</i>	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

11.8.2.8 spinSystemIsInUse()

```
SPINNAKERC_API spinSystemIsInUse (
    spinSystem hSystem,
    bool8_t * pbIsInUse )
```

Checks whether a system is currently in use.

See also

[spinError](#)

Parameters

<i>hSystem</i>	The system to check
<i>pbIsInUse</i>	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

11.8.2.9 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

<i>hSystem</i>	The system, on which the device arrival event handler is registered
<i>hDeviceArrivalEventHandler</i>	The device arrival event handler to register on the system

Returns

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

11.8.2.10 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

<i>hSystem</i>	The system, on which the device removal event handler is registered
<i>hDeviceRemovalEventHandler</i>	The device removal event handler to register on the system

Returns

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

11.8.2.11 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 [spinSystem↵RegisterInterfaceEventHandler\(\)](#) is called, those interfaces will be automatically registered with this event.

See also

[spinError](#)
[spinInterfaceEventHandler](#)

Parameters

<i>hSystem</i>	The system, on which the interface event handler is registered
<i>hInterfaceEventHandler</i>	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

11.8.2.12 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

<i>hSystem</i>	The system, on which the logging event handler is registered
<i>hLogEventHandler</i>	The logging event handler to register on the system

Returns

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

11.8.2.13 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

<i>hSystem</i>	The system handle
----------------	-------------------

Returns

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

11.8.2.14 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

<i>hSystem</i>	The system on which to send the action command to all devices.
<i>iDeviceKey</i>	The Action Command's device key
<i>iGroupKey</i>	The Action Command's group key
<i>iGroupMask</i>	The Action Command's group mask
<i>iActionTime</i>	(Optional) Time when to assert a future action. Zero means immediate action.
<i>piResultSize</i>	(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.
<i>results</i>	(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

11.8.2.15 spinSystemSetLoggingLevel()

```

SPINNAKERC_API spinSystemSetLoggingLevel (
    spinSystem hSystem,
    spinnakerLogLevel logLevel )

```

Sets the logging level for all logging events on the system.

See also

[spinError](#)

Parameters

<i>hSystem</i>	The system, on which the logging level is set
<i>logLevel</i>	The logging level to set

Returns

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

11.8.2.16 spinSystemUnregisterAllLogEventHandlers()

```
SPINNAKERC_API spinSystemUnregisterAllLogEventHandlers (
    spinSystem hSystem )
```

Unregisters all logging event handlers from the system.

See also

[spinError](#)

Parameters

<i>hSystem</i>	The system, from which all logging event handlers are unregistered
----------------	--

Returns

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

11.8.2.17 spinSystemUnregisterDeviceArrivalEventHandler()

```
SPINNAKERC_API spinSystemUnregisterDeviceArrivalEventHandler (
    spinSystem hSystem,
    spinDeviceArrivalEventHandler hDeviceArrivalEventHandler )
```

Unregisters a device arrival event handler from the system.

See also

[spinError](#)
[spinDeviceArrivalEventHandler](#)

Parameters

<i>hSystem</i>	The system, from which the device arrival event handler is unregistered
<i>hDeviceArrivalEventHandler</i>	The device arrival event handler to unregister from the system

Returns

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

11.8.2.18 spinSystemUnregisterDeviceRemovalEventHandler()

```
SPINNAKERC_API spinSystemUnregisterDeviceRemovalEventHandler (
    spinSystem hSystem,
    spinDeviceRemovalEventHandler hDeviceRemovalEventHandler )
```

Unregisters a device removal event handler from the system.

See also

[spinError](#)
[spinDeviceRemovalEventHandler](#)

Parameters

<i>hSystem</i>	The system, from which the device removal event handler is unregistered
<i>hDeviceRemovalEventHandler</i>	The device removal event handler to unregister from the system

Returns

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

11.8.2.19 spinSystemUnregisterInterfaceEventHandler()

```
SPINNAKERC_API spinSystemUnregisterInterfaceEventHandler (
    spinSystem hSystem,
    spinInterfaceEventHandler hInterfaceEventHandler )
```

Unregisters an interface event handler from the system.

See also

[spinError](#)
[spinInterfaceEventHandler](#)

Parameters

<i>hSystem</i>	The system, from which the interface event handler is unregistered
<i>hInterfaceEventHandler</i>	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

11.8.2.20 spinSystemUnregisterLogEventHandler()

```
SPINNAKERC_API spinSystemUnregisterLogEventHandler (
    spinSystem hSystem,
    spinLogEventHandler hLogEventHandler )
```

Unregisters a selected logging event handler from the system.

See also

[spinError](#)

Parameters

<i>hSystem</i>	The system, from which the logging event handler is unregistered
<i>hLogEventHandler</i>	The logging event handler to unregister from the system

Returns

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

11.8.2.21 spinSystemUpdateCameras()

```
SPINNAKERC_API spinSystemUpdateCameras (
    spinSystem hSystem,
    bool18_t * pbChanged )
```

Updates the list of cameras on the system, informing whether there has been any changes.

See also

[spinError](#)

Parameters

<i>hSystem</i>	The system, on which the list of attached cameras is updated
<i>pbChanged</i>	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

11.8.2.22 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

<i>hSystem</i>	The system, on which the list of attached cameras is updated
<i>bUpdateInterfaces</i>	The boolean of whether to update the interface list
<i>pbChanged</i>	The boolean pointer to return whether cameras have arrived or been removed from the system

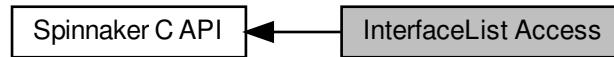
Returns

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

11.9 InterfaceList Access

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

Collaboration diagram for InterfaceList Access:



Functions

- [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.

11.9.1 Detailed Description

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

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

11.9.2 Function Documentation

11.9.2.1 spinInterfaceListClear()

```
SPINNAKERC_API spinInterfaceListClear (
    spinInterfaceList hInterfaceList )
```

Clears an interface list.

See also

[spinError](#)

Parameters

<i>hInterfaceList</i>	The interface list to clear
-----------------------	-----------------------------

Returns

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

11.9.2.2 `spinInterfaceListCreateEmpty()`

```
SPINNAKERC_API spinInterfaceListCreateEmpty (  
    spinInterfaceList * phInterfaceList )
```

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

See also

[spinError](#)

Parameters

<i>phInterfaceList</i>	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

11.9.2.3 `spinInterfaceListDestroy()`

```
SPINNAKERC_API spinInterfaceListDestroy (  
    spinInterfaceList hInterfaceList )
```

Destroys an interface list.

See also

[spinError](#)

Parameters

<i>hInterfaceList</i>	The interface list to destroy
-----------------------	-------------------------------

Returns

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

11.9.2.4 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

<i>hInterfaceList</i>	The interface list of the interface to be retrieved
<i>index</i>	The index of the interface
<i>phInterface</i>	The interface handle pointer in which the interface is returned

Returns

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

11.9.2.5 spinInterfaceListGetSize()

```
SPINNAKERC_API spinInterfaceListGetSize (
    spinInterfaceList hInterfaceList,
    size_t * pSize )
```

Retrieves the number of interfaces in an interface list.

See also

[spinError](#)

Parameters

<i>hInterfaceList</i>	The interface list where the interfaces to be counted are
<i>pSize</i>	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

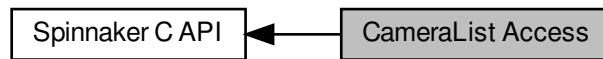
See also

[spinError](#)

11.10 CameraList Access

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

Collaboration diagram for CameraList Access:



Functions

- [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](#) *phCamera)
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](#) hCameraListToAppend)
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.

11.10.1 Detailed Description

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

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

11.10.2 Function Documentation

11.10.2.1 spinCameraListAppend()

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

Appends all the cameras from one camera list to another.

See also

[spinError](#)

Parameters

<i>hCameraListBase</i>	The camera list to receive the other
<i>hCameraListToAppend</i>	The camera list to add to the other

Returns

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

11.10.2.2 spinCameraListClear()

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

Clears a camera list.

See also

[spinError](#)

Parameters

<i>hCameraList</i>	The camera list to clear
--------------------	--------------------------

Returns

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

11.10.2.3 spinCameraListCreateEmpty()

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

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

See also

[spinError](#)

Parameters

<i>phCameraList</i>	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

11.10.2.4 spinCameraListDestroy()

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

Destroys a camera list.

See also

[spinError](#)

Parameters

<i>hCameraList</i>	The camera list to destroy
--------------------	----------------------------

Returns

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

11.10.2.5 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

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

Returns

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

11.10.2.6 `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

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

Returns

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

11.10.2.7 `spinCameraListGetSize()`

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

Retrieves the number of cameras on a camera list.

See also

[spinError](#)

Parameters

<i>hCameraList</i>	The camera list where the cameras to be counted are
<i>pSize</i>	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

11.10.2.8 spinCameraListRemove()

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

Removes a camera from a camera list using its index.

See also

[spinError](#)

Parameters

<i>hCameraList</i>	The camera list of the camera to remove
<i>index</i>	The index of the camera to remove

Returns

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

11.10.2.9 spinCameraListRemoveBySerial()

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

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

See also

[spinError](#)

Parameters

<i>hCameraList</i>	The camera of the camera to remove
<i>pSerial</i>	The serial number of the camera to remove

Returns

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

11.11 Interface Access

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

Collaboration diagram for Interface Access:



Functions

- [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](#) *phNodeMap)
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, [spinDeviceRemovalEventHandler](#) hDeviceRemovalEventHandler)
Registers a device removal event handler on an interface (event handlers registered in this way must be unregistered)
- [SPINNAKERC_API spinInterfaceUnregisterDeviceArrivalEventHandler](#) ([spinInterface](#) hInterface, [spinDeviceArrivalEventHandler](#) hDeviceArrivalEventHandler)
Unregisters a device arrival event handler from an interface.
- [SPINNAKERC_API spinInterfaceUnregisterDeviceRemovalEventHandler](#) ([spinInterface](#) hInterface, [spinDeviceRemovalEventHandler](#) 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.

11.11.1 Detailed Description

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.

11.11.2 Function Documentation

11.11.2.1 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

<i>hInterface</i>	The interface of the camera list to retrieve
<i>hCameraList</i>	The camera list to house the cameras from the interface

Returns

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

11.11.2.2 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

<i>hInterface</i>	The interface of the camera list to retrieve
<i>bUpdateCameras</i>	The boolean of whether or not to update the cameras
<i>hCameraList</i>	The camera list to house the cameras from the interface

Returns

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

11.11.2.3 spinInterfaceGetTLNodeMap()

```
SPINNAKERC_API spinInterfaceGetTLNodeMap (
    spinInterface hInterface,
    spinNodeMapHandle * phNodeMap )
```

Retrieves the transport layer nodemap from an interface.

See also

[spinError](#)

Parameters

<i>hInterface</i>	The interface of the nodemap to retrieve
<i>phNodeMap</i>	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

11.11.2.4 spinInterfaceIsInUse()

```
SPINNAKERC_API spinInterfaceIsInUse (
    spinInterface hInterface,
    bool8_t * pbIsInUse )
```

Checks whether an interface is in use.

See also

[spinError](#)

Parameters

<i>hInterface</i>	The interface to check
<i>pblsInUse</i>	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

11.11.2.5 `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

<i>hInterface</i>	The interface on which to register the device arrival event handler
<i>hDeviceArrivalEventHandler</i>	The device arrival event handler to register

Returns

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

11.11.2.6 `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

<i>hInterface</i>	the Interface on which to register the device removal event handler
<i>hDeviceRemovalEventHandler</i>	The device removal event handler to register

Returns

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

11.11.2.7 `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

<i>hInterface</i>	The interface on which to register the interface event handler
<i>hInterfaceEventHandler</i>	The interface event handler to register

Returns

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

11.11.2.8 `spinInterfaceRelease()`

```
SPINNAKERC_API spinInterfaceRelease (
    spinInterface hInterface )
```

Releases an interface.

See also

[spinError](#)

Parameters

<i>hInterface</i>	The interface to release
-------------------	--------------------------

Returns

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

11.11.2.9 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

<i>iDeviceKey</i>	The Action Command's device key
<i>iGroupKey</i>	The Action Command's group key
<i>iGroupMask</i>	The Action Command's group mask
<i>iActionTime</i>	(Optional) Time when to assert a future action. Zero means immediate action.
<i>piResultSize</i>	(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.
<i>results</i>	(Optional) An Array with <i>*piResultSize</i> 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 <i>piResultSize</i> is 0. Thus this parameter can be NULL if <i>piResultSize</i> is 0 or NULL.

Returns

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

11.11.2.10 spinInterfaceUnregisterDeviceArrivalEventHandler()

```
SPINNAKERC_API spinInterfaceUnregisterDeviceArrivalEventHandler (
    spinInterface hInterface,
    spinDeviceArrivalEventHandler hDeviceArrivalEventHandler )
```

Unregisters a device arrival event handler from an interface.

See also

[spinError](#)

Parameters

<i>hInterface</i>	The interface from which to unregister the device arrival event handler
<i>hDeviceArrivalEventHandler</i>	The device arrival event handler to unregister

Returns

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

11.11.2.11 `spinInterfaceUnregisterDeviceRemovalEventHandler()`

```
SPINNAKERC_API spinInterfaceUnregisterDeviceRemovalEventHandler (
    spinInterface hInterface,
    spinDeviceRemovalEventHandler hDeviceRemovalEventHandler )
```

Unregisters a device removal event handler from an interface.

See also

[spinError](#)

Parameters

<i>hInterface</i>	The interface from which to unregister the device removal event handler
<i>hDeviceRemovalEventHandler</i>	The device removal event handler to unregister

Returns

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

11.11.2.12 `spinInterfaceUnregisterInterfaceEventHandler()`

```
SPINNAKERC_API spinInterfaceUnregisterInterfaceEventHandler (
    spinInterface hInterface,
    spinInterfaceEventHandler hInterfaceEventHandler )
```

Unregisters an interface event handler from an interface.

See also

[spinError](#)

Parameters

<i>hInterface</i>	The interface from which to unregister the interface event handler
<i>hInterfaceEventHandler</i>	The interface event handler to unregister

Returns

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

11.11.2.13 `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

<i>hInterface</i>	The interface of the list of attached cameras to update
<i>pbChanged</i>	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

11.12 Camera Access

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

Collaboration diagram for Camera Access:



Functions

- [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 \(spinCamera hCamera, spinNodeMapHandle *phNodeMap\)](#)
Retrieves the transport layer device nodemap from a camera.
- [SPINNAKERC_API spinCameraGetTLStreamNodeMap \(spinCamera hCamera, spinNodeMapHandle *phNodeMap\)](#)
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 spinCameraGetUniqueID \(spinCamera hCamera, char *pBuf, size_t *pBufLen\)](#)
Retrieves a unique identifier for a camera.
- [SPINNAKERC_API spinCameraIsStreaming \(spinCamera hCamera, bool8_t *pIsStreaming\)](#)
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, [spinDeviceEvent↵Handler](#) hDeviceEventHandler)
Registers a universal device event handler (every device event type) to a camera.
- [SPINNAKERC_API spinCameraRegisterDeviceEventHandlerEx](#) ([spinCamera](#) hCamera, [spinDeviceEvent↵Handler](#) hDeviceEventHandler, const char *pName)
Registers a specific device event handler (only one device event type) to a camera.
- [SPINNAKERC_API spinCameraUnregisterDeviceEventHandler](#) ([spinCamera](#) hCamera, [spinDeviceEvent↵Handler](#) hDeviceEventHandler)
Unregisters a device event handler from a camera.
- [SPINNAKERC_API spinCameraRegisterImageEventHandler](#) ([spinCamera](#) hCamera, [spinImageEvent↵Handler](#) hImageEventHandler)
Registers an image event handler to a camera.
- [SPINNAKERC_API spinCameraUnregisterImageEventHandler](#) ([spinCamera](#) hCamera, [spinImageEvent↵Handler](#) hImageEventHandler)
Unregisters an image event handler from a camera.
- [SPINNAKERC_API spinCameraRelease](#) ([spinCamera](#) hCamera)
Releases a camera.
- [SPINNAKERC_API spinCamerasValid](#) ([spinCamera](#) hCamera, [bool8_t](#) *pbValid)
Checks whether a camera is still valid for use.
- [SPINNAKERC_API spinCamerasInitialized](#) ([spinCamera](#) hCamera, [bool8_t](#) *pbInit)
Checks whether a camera is currently initialized.

11.12.1 Detailed Description

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.

11.12.2 Function Documentation

11.12.2.1 spinCameraBeginAcquisition()

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

Has a camera start acquiring images.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera to begin acquiring images
----------------	--------------------------------------

Returns

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

11.12.2.2 spinCameraDeInit()

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

Deinitializes a camera, greatly reducing functionality.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera to deinitialize
----------------	----------------------------

Returns

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

11.12.2.3 spinCameraEndAcquisition()

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

Has a camera stop acquiring images.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera to stop acquiring images
----------------	-------------------------------------

Returns

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

11.12.2.4 `spinCameraGetAccessMode()`

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

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

See also

[spinError](#)
[spinAccessMode](#)

Parameters

<i>hCamera</i>	The camera of the access mode to retrieve
<i>pAccessMode</i>	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

11.12.2.5 `spinCameraGetGuiXml()`

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

Retrieves the GUI XML from a camera.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera of the GUI XML to retrieve
<i>pBuf</i>	The c-string character buffer in which the GUI XML is returned
<i>pBufLen</i>	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

11.12.2.6 spinCameraGetNextImage()

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

Retrieves an image from a camera.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera of the image to retrieve
<i>phImage</i>	The image handle pointer in which the image is returned

Returns

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

11.12.2.7 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

<i>hCamera</i>	The camera of the image to retrieve
<i>grabTimeout</i>	The timeout value for returned an image
<i>phImage</i>	The image handle pointer in which the image is returned

Returns

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

11.12.2.8 spinCameraGetNodeMap()

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

Retrieves the GenICam nodemap from a camera.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera from which the nodemap is retrieved
<i>phNodeMap</i>	The nodemap handle pointer in which the nodemap is returned

Returns

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

11.12.2.9 spinCameraGetTLDeviceNodeMap()

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

Retrieves the transport layer device nodemap from a camera.

See also

[spinError](#)

Parameters

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

Returns

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

11.12.2.10 spinCameraGetTLStreamNodeMap()

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

Retrieves the transport layer stream nodemap from a camera.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera from which the transport layer streaming nodemap is retrieved
<i>pNodeMap</i>	The nodemap handle pointer in which the nodemap is returned

Returns

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

11.12.2.11 spinCameraGetUniqueID()

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

Retrieves a unique identifier for a camera.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera of the unique identifier
<i>pBuf</i>	The c-string character buffer in which the unique identifier is returned
<i>pBufLen</i>	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

11.12.2.12 spinCameraInit()

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

Initializes a camera, allowing for much more interaction.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera to initialize
----------------	--------------------------

Returns

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

11.12.2.13 spinCameraIsInitialized()

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

Checks whether a camera is currently initialized.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera to check
<i>pbInit</i>	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

11.12.2.14 spinCameraIsStreaming()

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

Checks whether a camera is currently acquiring images.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera to check
<i>pbIsStreaming</i>	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

11.12.2.15 `spinCamerasValid()`

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

Checks whether a camera is still valid for use.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera to check
<i>pbValid</i>	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

11.12.2.16 `spinCameraReadPort()`

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

11.12.2.17 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

<i>hCamera</i>	The camera on which to register the universal device event handler
<i>hDeviceEventHandler</i>	The device event handler to register

Returns

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

11.12.2.18 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

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

Returns

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

11.12.2.19 spinCameraRegisterImageEventHandler()

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

Registers an image event handler to a camera.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera on which to register the image event handler
<i>hImageEventHandler</i>	The image event handler to register

Returns

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

11.12.2.20 spinCameraRelease()

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

Releases a camera.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera to release
----------------	-----------------------

Returns

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

11.12.2.21 spinCameraUnregisterDeviceEventHandler()

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

Unregisters a device event handler from a camera.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera from which to unregister the device event handler
<i>hDeviceEventHandler</i>	The device event handler to unregister

Returns

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

11.12.2.22 `spinCameraUnregisterImageEventHandler()`

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

Unregisters an image event handler from a camera.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera from which to unregister the image event handler
<i>hImageEventHandler</i>	The image event handler to unregister

Returns

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

11.12.2.23 `spinCameraWritePort()`

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

11.13 Image Access

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

Functions

- [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, [spinPayloadTypeInfoIds](#) 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 spinImageSetDefaultColorProcessing](#) ([spinColorProcessingAlgorithm](#) algorithm)
Sets the default color processing algorithm of all images (if not otherwise set)
- [SPINNAKERC_API spinImageGetDefaultColorProcessing](#) ([spinColorProcessingAlgorithm](#) *pAlgorithm)
Retrieves the default color processing algorithm.
- [SPINNAKERC_API spinImageGetColorProcessing](#) ([spinImage](#) hImage, [spinColorProcessingAlgorithm](#) *pAlgorithm)
Retrieves the color processing algorithm of a specific image.
- [SPINNAKERC_API spinImageSetNumDecompressionThreads](#) (unsigned int numThreads)
Sets the default number of threads used for image decompression during [spinImageConvert\(\)](#).
- [SPINNAKERC_API spinImageGetNumDecompressionThreads](#) (unsigned int *pNumThreads)
Gets the number of threads used for image decompression during [Convert\(\)](#).
- [SPINNAKERC_API spinImageConvert](#) ([spinImage](#) hSrcImage, [spinPixelFormatEnums](#) pixelFormat, [spinImage](#) hDestImage)
Converts the pixel format of one image into a new image.
- [SPINNAKERC_API spinImageConvertEx](#) ([spinImage](#) hSrcImage, [spinPixelFormatEnums](#) pixelFormat, [spinColorProcessingAlgorithm](#) algorithm, [spinImage](#) hDestImage)
Converts the pixel format and color processing algorithm of one image into a new 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, [spinPayloadTypeInfolDs](#))
 - [SPINNAKERC_API spinImageGetTLPayloadType](#) ([spinImage](#) hImage, [spinPayloadTypeInfoDs](#) *pPayloadType)

Retrieves the transport layer payload type of an image (as an enum, [spinPayloadTypeInfolDs](#))
 - [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, [spinPixelFormatNamespaceID](#) *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](#) *pIsIncomplete)

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.

11.13.1 Detailed Description

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.

11.13.2 Function Documentation

11.13.2.1 spinImageCalculateStatistics()

```
SPINNAKERC_API spinImageCalculateStatistics (
    spinImage hImage,
    const spinImageStatistics hStatistics )
```

Calculates the image statistics of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>hStatistics</i>	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

11.13.2.2 `spinImageCheckCRC()`

```
SPINNAKER_API spinImageCheckCRC (  
    spinImage hImage,  
    bool8_t * pbCheckCRC )
```

Checks whether the CRC of an image is correct.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pbCheckCRC</i>	The boolean pointer to return whether the image CRC passes

Returns

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

11.13.2.3 `spinImageConvert()`

```
SPINNAKER_API spinImageConvert (  
    spinImage hSrcImage,  
    spinPixelFormatEnums pixelFormat,  
    spinImage hDestImage )
```

Converts the pixel format of one image into a new image.

See also

[spinError](#)

Parameters

<i>hSrcImage</i>	The image to be converted
<i>pixelFormat</i>	The pixel format to be converted to
<i>hDestImage</i>	The image handle pointer in which the converted image is returned

Returns

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

11.13.2.4 `spinImageConvertEx()`

```
SPINNAKERC_API spinImageConvertEx (
    spinImage hSrcImage,
    spinPixelFormatEnums pixelFormat,
    spinColorProcessingAlgorithm algorithm,
    spinImage hDestImage )
```

Converts the pixel format and color processing algorithm of one image into a new image.

See also

[spinError](#)

Parameters

<i>hSrcImage</i>	The image to be converted
<i>pixelFormat</i>	The pixel format to be converted to
<i>algorithm</i>	The color processing algorithm to use for conversion
<i>hDestImage</i>	The image handle pointer in which the converted image is returned

Returns

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

11.13.2.5 `spinImageCreate()`

```
SPINNAKERC_API spinImageCreate (
    spinImage hSrcImage,
    spinImage * phDestImage )
```

Creates an image from another; images created this way must be destroyed.

See also

[spinError](#)

Parameters

<i>hSrcImage</i>	The image to be copied
<i>phDestImage</i>	The image handle pointer of the image to be created

Returns

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

11.13.2.6 `spinImageCreateEmpty()`

```
SPINNAKERC_API spinImageCreateEmpty (  
    spinImage * phImage )
```

Creates an empty image; images created this way must be destroyed.

See also

[spinError](#)

Parameters

<i>phImage</i>	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

11.13.2.7 `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

<i>phImage</i>	The image handle pointer in which the image is returned
<i>width</i>	The width to set
<i>height</i>	The height to set
<i>offsetX</i>	The offset along the X axis to set
<i>offsetY</i>	The offset along the Y axis to set
<i>pixelFormat</i>	The pixel format to set
<i>pData</i>	The image data to set; can be set to null

Returns

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

11.13.2.8 `spinImageCreateEx2()`

```
SPINNAKERC_API spinImageCreateEx2 (
    spinImage * phImage,
    size_t width,
    size_t height,
    size_t offsetX,
    size_t offsetY,
    spinPixelFormatEnums pixelFormat,
    void * pData,
    spinPayloadTypeInfoIds dataPayloadType,
    size_t dataSize )
```

Creates an image with some set properties; images created this way must be destroyed.

See also

[spinError](#)
[spinImageGetTLPayloadType](#)

Parameters

<i>phImage</i>	The image handle pointer in which the image is returned
<i>width</i>	The width to set
<i>height</i>	The height to set
<i>offsetX</i>	The offset along the X axis to set
<i>offsetY</i>	The offset along the Y axis to set
<i>pixelFormat</i>	The pixel format to set
<i>pData</i>	The image data to set; can be set to null
<i>dataPayloadType</i>	The payload type of the data. This value can be retrieved from an existing image by using the spinImageGetTLPayloadType() function call.
<i>dataSize</i>	The size of the provided data in bytes

Returns

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

11.13.2.9 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

<i>hSrcImage</i>	The image to be copied
<i>hDestImage</i>	The image handle in which the image is copied

Returns

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

11.13.2.10 spinImageDestroy()

```
SPINNAKERC_API spinImageDestroy (
    spinImage hImage )
```

Destroys an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to destroy
---------------	----------------------

Returns

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

11.13.2.11 spinImageGetBitsPerPixel()

```
SPINNAKERC_API spinImageGetBitsPerPixel (
    spinImage hImage,
    size_t * pBitsPerPixel )
```

Retrieves the number of bits per pixel of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pBitsPerPixel</i>	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

11.13.2.12 spinImageGetBufferSize()

```
SPINNAKERC_API spinImageGetBufferSize (
    spinImage hImage,
    size_t * pSize )
```

Retrieves the buffer size of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of image data buffer to retrieve
<i>pSize</i>	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

11.13.2.13 spinImageGetChunkLayoutID()

```
SPINNAKERC_API spinImageGetChunkLayoutID (
    spinImage hImage,
    uint64_t * pId )
```

Retrieves the chunk layout ID of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pId</i>	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

11.13.2.14 spinImageGetColorProcessing()

```
SPINNAKERC_API spinImageGetColorProcessing (
    spinImage hImage,
    spinColorProcessingAlgorithm * pAlgorithm )
```

Retrieves the color processing algorithm of a specific image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of the color processing algorithm to retrieve
<i>pAlgorithm</i>	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

11.13.2.15 spinImageGetData()

```
SPINNAKERC_API spinImageGetData (
    spinImage hImage,
    void ** ppData )
```

Retrieves the image data of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of the image data to retrieve
<i>ppData</i>	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

11.13.2.16 spinImageGetDefaultColorProcessing()

```
SPINNAKERC_API spinImageGetDefaultColorProcessing (
    spinColorProcessingAlgorithm * pAlgorithm )
```

Retrieves the default color processing algorithm.

See also

[spinError](#)

Parameters

<i>pAlgorithm</i>	The color processing algorithm enum pointer in which the color processing algorithm is returned
-------------------	---

Returns

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

11.13.2.17 spinImageGetFrameID()

```
SPINNAKERC_API spinImageGetFrameID (
    spinImage hImage,
    uint64_t * pFrameID )
```

Retrieves the frame ID of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of the frame ID to retrieve
<i>pFrameID</i>	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

11.13.2.18 `spinImageGetHeight()`

```
SPINNAKERC_API spinImageGetHeight (
    spinImage hImage,
    size_t * pHeight )
```

Retrieves the height of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of the height to retrieve
<i>pHeight</i>	The unsigned integer pointer in which the height is returned

Returns

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

11.13.2.19 `spinImageGetID()`

```
SPINNAKERC_API spinImageGetID (
    spinImage hImage,
    uint64_t * pId )
```

Retrieves the ID of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of the ID to retrieve
<i>pId</i>	The unsigned integer pointer in which the ID is returned

Returns

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

11.13.2.20 spinImageGetNumDecompressionThreads()

```
SPINNAKERC_API spinImageGetNumDecompressionThreads (
    unsigned int * pNumThreads )
```

Gets the number of threads used for image decompression during `Convert()`.

Parameters

<i>pNumThreads</i>	The pointer indicating the number of parallel image decompression threads set to run
--------------------	--

See also

[spinImageSetNumDecompressionThreads\(\)](#)

Returns

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

11.13.2.21 spinImageGetOffsetX()

```
SPINNAKERC_API spinImageGetOffsetX (
    spinImage hImage,
    size_t * pOffsetX )
```

Retrieves the offset of an image along its X axis.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of the offset along the X axis to retrieve
<i>pOffsetX</i>	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

11.13.2.22 spinImageGetOffsetY()

```
SPINNAKERC_API spinImageGetOffsetY (
    spinImage hImage,
    size_t * pOffsetY )
```

Retrieves the offset of an image along its Y axis.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of the offset along the Y axis to retrieve
<i>pOffsetY</i>	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

11.13.2.23 spinImageGetPaddingX()

```
SPINNAKERC_API spinImageGetPaddingX (
    spinImage hImage,
    size_t * pPaddingX )
```

Retrieves the padding of an image along its X axis.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of the padding along the X axis to retrieve
<i>pPaddingX</i>	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

11.13.2.24 spinImageGetPaddingY()

```
SPINNAKERC_API spinImageGetPaddingY (
    spinImage hImage,
    size_t * pPaddingY )
```

Retrieves the padding of an image along its Y axis.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of the padding along the Y axis to retrieve
<i>pPaddingY</i>	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

11.13.2.25 spinImageGetPayloadType()

```
SPINNAKERC_API spinImageGetPayloadType (
    spinImage hImage,
    size_t * pPayloadType )
```

Retrieves the payload type of an image (as an enum, `spinPayloadTypeIn folds`)

See also

[spinError](#)

`spinPayloadTypeIn folds`

Parameters

<i>hImage</i>	The image of the payload type to retrieve
<i>pPayloadType</i>	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

11.13.2.26 spinImageGetPixelFormat()

```
SPINNAKERC_API spinImageGetPixelFormat (
    spinImage hImage,
    spinPixelFormatEnums * pPixelFormat )
```

Retrieves the pixel format of an image (as an enum, `spinPixelFormatEnums`)

See also

[spinError](#)
[spinPixelFormatEnums](#)

Parameters

<i>hImage</i>	The image of the pixel format to retrieve
<i>pPixelFormat</i>	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

11.13.2.27 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

<i>hImage</i>	The image of the pixel format to retrieve
<i>pBuf</i>	The c-string character buffer in which the pixel format symbolic is returned
<i>pBufLen</i>	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

11.13.2.28 spinImageGetPrivateData()

```
SPINNAKERC_API spinImageGetPrivateData (
    spinImage hImage,
    void ** ppData )
```

Retrieves the private data of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of the private image data to retrieve
<i>ppData</i>	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

11.13.2.29 spinImageGetSize()

```
SPINNAKERC_API spinImageGetSize (
    spinImage hImage,
    size_t * pImageSize )
```

Retrieves the size of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pImageSize</i>	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

11.13.2.30 `spinImageGetStatus()`

```
SPINNAKERC_API spinImageGetStatus (
    spinImage hImage,
    spinImageStatus * pStatus )
```

Retrieves the image status of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pStatus</i>	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

11.13.2.31 `spinImageGetStatusDescription()`

```
SPINNAKERC_API spinImageGetStatusDescription (
    spinImageStatus status,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the description of image status.

See also

[spinError](#)

Parameters

<i>status</i>	The status enum
<i>pBuf</i>	The c-string character buffer in which the explanation of image status enum is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length; if <code>pBuf</code> is NULL, minimum length of string buffer is returned

Returns

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

11.13.2.32 spinImageGetStride()

```
SPINNAKERC_API spinImageGetStride (
    spinImage hImage,
    size_t * pStride )
```

Retrieves the stride of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pStride</i>	The unsigned integer pointer in which the stride is returned

Returns

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

11.13.2.33 spinImageGetTimeStamp()

```
SPINNAKERC_API spinImageGetTimeStamp (
    spinImage hImage,
    uint64_t * pTimeStamp )
```

Retrieves the timestamp of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of the timestamp to retrieve
<i>pTimeStamp</i>	The unsigned integer pointer om which the timestamp is returned

Returns

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

11.13.2.34 spinImageGetTLPayloadType()

```
SPINNAKERC_API spinImageGetTLPayloadType (
    spinImage hImage,
    spinPayloadTypeInfoIDs * pPayloadType )
```

Retrieves the transport layer payload type of an image (as an enum, `spinPayloadTypeInfolDs`)

See also

[spinError](#)
`spinPayloadTypeInfolDs`

Parameters

<i>hImage</i>	The image of the TL payload type to retrieve
<i>pPayloadType</i>	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

11.13.2.35 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

<i>hImage</i>	The image of the TL pixel format to retrieve
<i>pPixelFormat</i>	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

11.13.2.36 spinImageGetTLPixelFormatNamespace()

```
SPINNAKERC_API spinImageGetTLPixelFormatNamespace (
    spinImage hImage,
    spinPixelFormatNamespaceID * pPixelFormatNamespace )
```

Retrieves the transport layer pixel format namespace of an image (as an enum, `spinPixelFormatNamespaceID`)

See also

[spinError](#)
[spinPixelFormatNamespaceID](#)

Parameters

<i>hImage</i>	The image of the TL pixel format namespace to retrieve
<i>pPixelFormatNamespace</i>	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

11.13.2.37 spinImageGetValidPayloadSize()

```
SPINNAKERC_API spinImageGetValidPayloadSize (
    spinImage hImage,
    size_t * pSize )
```

Retrieves the valid payload size of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of the payload size to retrieve
<i>pSize</i>	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

11.13.2.38 spinImageGetWidth()

```
SPINNAKERC_API spinImageGetWidth (
    spinImage hImage,
    size_t * pWidth )
```

Retrieves the width of an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image of the width to retrieve
<i>pWidth</i>	The unsigned integer pointer in which the width is returned

Returns

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

11.13.2.39 spinImageHasCRC()

```
SPINNAKERC_API spinImageHasCRC (
    spinImage hImage,
    bool8_t * pbHasCRC )
```

Checks whether an image has CRC.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pbHasCRC</i>	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

11.13.2.40 spinImageIsIncomplete()

```
SPINNAKERC_API spinImageIsIncomplete (
    spinImage hImage,
    bool8_t * pbIsIncomplete )
```

Checks whether an image is incomplete.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to check
<i>pbIsIncomplete</i>	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

11.13.2.41 spinImageRelease()

```
SPINNAKERC_API spinImageRelease (
    spinImage hImage )
```

Releases an image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
---------------	-----------------------

Returns

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

11.13.2.42 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

<i>hImage</i>	The image to be reset
<i>width</i>	The width to be reset to
<i>height</i>	The height to be reset to
<i>offsetX</i>	The offset to be reset to along the X axis
<i>offsetY</i>	The offset to be reset to along the Y axis
<i>pixelFormat</i>	The pixel format to be reset to

Returns

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

11.13.2.43 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

<i>hImage</i>	The image to reset
---------------	--------------------

Parameters

<i>width</i>	The width to be reset to
<i>height</i>	The height to be reset to
<i>offsetX</i>	The offset to be reset to along the X axis
<i>offsetY</i>	The offset to be reset to along the Y axis
<i>pixelFormat</i>	The pixel format to be reset to
<i>pData</i>	The image data to reset to

Returns

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

11.13.2.44 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

<i>hImage</i>	The image to be saved
<i>pFilename</i>	The filename to use to save the image (with or without the appropriate file extension) format The file format to use to save the image

Returns

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

11.13.2.45 spinImageSaveBmp()

```
SPINNAKERC_API spinImageSaveBmp (
    spinImage hImage,
    const char * pFilename,
    const spinBMPOption * pOption )
```

Saves an image as a BMP image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pFilename</i>	The filename to use to save the image (with or without the appropriate file extension)
<i>pOption</i>	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

11.13.2.46 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

<i>hImage</i>	The image to be saved
<i>pFilename</i>	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

11.13.2.47 spinImageSaveJpeg()

```
SPINNAKERC_API spinImageSaveJpeg (
    spinImage hImage,
    const char * pFilename,
    const spinJPEGOption * pOption )
```

Saves an image as a JPEG image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pFilename</i>	The filename to use to save the image (with or without the appropriate file extension)
<i>pOption</i>	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

11.13.2.48 `spinImageSaveJpg2()`

```
SPINNAKERC_API spinImageSaveJpg2 (
    spinImage hImage,
    const char * pFilename,
    const spinJPG2Option * pOption )
```

Saves an image as a JPEG 2000 image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pFilename</i>	The filename to use to save the image (with or without the appropriate file extension)
<i>pOption</i>	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

11.13.2.49 `spinImageSavePgm()`

```
SPINNAKERC_API spinImageSavePgm (
    spinImage hImage,
    const char * pFilename,
    const spinPGMOption * pOption )
```

Saves an image as an PGM image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pFilename</i>	The filename to use to save the image (with or without the appropriate file extension)
<i>pOption</i>	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

11.13.2.50 `spinImageSavePng()`

```
SPINNAKERC_API spinImageSavePng (  
    spinImage hImage,  
    const char * pFilename,  
    const spinPNGOption * pOption )
```

Saves an image as a PNG image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pFilename</i>	The filename to use to save the image (with or without the appropriate file extension)
<i>pOption</i>	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

11.13.2.51 `spinImageSavePpm()`

```
SPINNAKERC_API spinImageSavePpm (  
    spinImage hImage,  
    const char * pFilename,  
    const spinPPMOption * pOption )
```

Saves an image as a PPM image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pFilename</i>	The filename to use to save the image (with or without the appropriate file extension)
<i>pOption</i>	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

11.13.2.52 `spinImageSaveTiff()`

```
SPINNAKER_API spinImageSaveTiff (
    spinImage hImage,
    const char * pFilename,
    const spinTIFFOption * pOption )
```

Saves an image as a TIFF image.

See also

[spinError](#)

Parameters

<i>hImage</i>	The image to be saved
<i>pFilename</i>	The filename to use to save the image (with or without the appropriate file extension)
<i>pOption</i>	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

11.13.2.53 `spinImageSetDefaultColorProcessing()`

```
SPINNAKER_API spinImageSetDefaultColorProcessing (
    spinColorProcessingAlgorithm algorithm )
```

Sets the default color processing algorithm of all images (if not otherwise set)

See also

[spinError](#)

Parameters

<i>algorithm</i>	The color processing algorithm used by default
------------------	--

Returns

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

11.13.2.54 spinImageSetNumDecompressionThreads()

```
SPINNAKERC_API spinImageSetNumDecompressionThreads (
    unsigned int numThreads )
```

Sets the default number of threads used for image decompression during [spinImageConvert\(\)](#).

The number of threads used is defaulted to be equal to one less than the number of concurrent threads supported by the system.

Parameters

<i>numThreads</i>	Number of parallel image decompression threads set to run
-------------------	---

See also

[spinImageConvert\(\)](#)

Returns

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

11.14 Event Access

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

Functions

- [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 spinDeviceArrivalEventHandlerCreate](#) ([spinDeviceArrivalEventHandler](#) *phDeviceArrivalEventHandler, [spinArrivalEventFunction](#) pFunction, void *pUserData)
Creates a device arrival event handler.
- [SPINNAKERC_API spinDeviceArrivalEventHandlerDestroy](#) ([spinDeviceArrivalEventHandler](#) hDeviceArrivalEventHandler)
Destroys a device arrival event handler.
- [SPINNAKERC_API spinDeviceRemovalEventHandlerCreate](#) ([spinDeviceRemovalEventHandler](#) *phDeviceRemovalEventHandler, [spinRemovalEventFunction](#) pFunction, void *pUserData)
Creates a device removal event handler.
- [SPINNAKERC_API spinDeviceRemovalEventHandlerDestroy](#) ([spinDeviceRemovalEventHandler](#) hDeviceRemovalEventHandler)
Destroys a device removal event handler.
- [SPINNAKERC_API spinInterfaceEventHandlerCreate](#) ([spinInterfaceEventHandler](#) *phInterfaceEventHandler, [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.

11.14.1 Detailed Description

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

11.14.2 Function Documentation

11.14.2.1 spinDeviceArrivalEventHandlerCreate()

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

Creates a device arrival event handler.

See also

[spinError](#)

Parameters

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

Returns

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

11.14.2.2 spinDeviceArrivalEventHandlerDestroy()

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

Destroys a device arrival event handler.

See also

[spinError](#)

Parameters

<i>hDeviceArrivalEventHandler</i>	The device arrival event handler to destroy
-----------------------------------	---

Returns

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

11.14.2.3 spinDeviceEventHandlerCreate()

```
SPINNAKERC_API spinDeviceEventHandlerCreate (
    spinDeviceEventHandler * phDeviceEventHandler,
    spinDeviceEventFunction pFunction,
    void * pUserData )
```

Creates a device event handler.

See also

[spinError](#)

Parameters

<i>phDeviceEventHandler</i>	The device event handler pointer in which the device event context is created
<i>pFunction</i>	The function to be called at device event occurrences; signature to match: void(spinDeviceEventFunction)(const spinDeviceEventData hEventData, const char pEventName, void* pUserData)
<i>pUserData</i>	Properties that can be passed into the event function

Returns

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

11.14.2.4 spinDeviceEventHandlerDestroy()

```
SPINNAKERC_API spinDeviceEventHandlerDestroy (
    spinDeviceEventHandler hDeviceEventHandler )
```

Destroys a device event handler.

See also

[spinError](#)

Parameters

<i>hDeviceEventHandler</i>	The device event handler to destroy
----------------------------	-------------------------------------

Returns

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

11.14.2.5 spinDeviceRemovalEventHandlerCreate()

```
SPINNAKERC_API spinDeviceRemovalEventHandlerCreate (
    spinDeviceRemovalEventHandler * phDeviceRemovalEventHandler,
    spinRemovalEventFunction pFunction,
    void * pUserData )
```

Creates a device removal event handler.

See also

[spinError](#)

Parameters

<i>phDeviceRemovalEventHandler</i>	The device removal event handler pointer in which the device removal event context is created
<i>pFunction</i>	The function to be called at device event occurrences; signature to match: void(spinRemovalEventFunction)(uint64_t deviceSerialNumber, void pUserData)
<i>pUserData</i>	Properties that can be passed into the event function

Returns

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

11.14.2.6 spinDeviceRemovalEventHandlerDestroy()

```
SPINNAKERC_API spinDeviceRemovalEventHandlerDestroy (
    spinDeviceRemovalEventHandler hDeviceRemovalEventHandler )
```

Destroys a device removal event handler.

See also

[spinError](#)

Parameters

<i>hDeviceRemovalEventHandler</i>	The device removal event handler to destroy
-----------------------------------	---

Returns

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

11.14.2.7 spinImageEventHandlerCreate()

```
SPINNAKERC_API spinImageEventHandlerCreate (
    spinImageEventHandler * phImageEventHandler,
    spinImageEventFunction pFunction,
    void * pUserData )
```

Creates an image event handler.

See also

[spinError](#)

Parameters

<i>phImageEventHandler</i>	The image event handler pointer in which the image event context is created
<i>pFunction</i>	The function to be called at image event occurrences; signature to match: void(spinImageEventFunction)(const spinImage hImage, void pUserData)
<i>pUserData</i>	Properties that can be passed into the event function

Returns

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

11.14.2.8 spinImageEventHandlerDestroy()

```
SPINNAKERC_API spinImageEventHandlerDestroy (
    spinImageEventHandler hImageEventHandler )
```

Destroys an image event handler.

See also

[spinError](#)

Parameters

<i>hImageEventHandler</i>	The image event handler to destroy
---------------------------	------------------------------------

Returns

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

11.14.2.9 `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

<i>phInterfaceEventHandler</i>	The interface event handler pointer in which the interface event context is created
<i>pArrivalFunction</i>	The function to be called at arrival event occurrences; signature to match: void(spinArrivalEventFunction)(void pUserData)
<i>hRemovalFunction</i>	The function to be called at removal event occurrences; signature to match: void(spinRemovalEventFunction)(uint64_t deviceSerialNumber, void pUserData)
<i>pUserData</i>	Properties that can be passed into the event function

Returns

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

11.14.2.10 `spinInterfaceEventHandlerDestroy()`

```
SPINNAKERC_API spinInterfaceEventHandlerDestroy (
    spinInterfaceEventHandler hInterfaceEventHandler )
```

Destroys an interface event handler (both device arrival and device removal)

See also

[spinError](#)

Parameters

<i>hInterfaceEventHandler</i>	The interface event handler to destroy
-------------------------------	--

Returns

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

11.14.2.11 spinLogEventHandlerCreate()

```
SPINNAKERC_API spinLogEventHandlerCreate (
    spinLogEventHandler * phLogEventHandler,
    spinLogEventFunction pFunction,
    void * pUserData )
```

Creates a log event handler.

See also

[spinError](#)

Parameters

<i>phLogEventHandler</i>	The log event handler pointer in which the log event context is created
<i>pFunction</i>	The function to be called at device event occurrences; signature to match: void(spinLogEventFunction)(const spinLogEventData hEventData, void pUserData)
<i>pUserData</i>	Properties that can be passed into the event function

Returns

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

11.14.2.12 spinLogEventHandlerDestroy()

```
SPINNAKERC_API spinLogEventHandlerDestroy (
    spinLogEventHandler hLogEventHandler )
```

Destroys a log event handler.

See also

[spinError](#)

Parameters

<i>hLogEventHandler</i>	The log event handler to destroy
-------------------------	----------------------------------

Returns

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

11.15 ImageStatistics Access

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

Functions

- [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 *pPixelValueMax, unsigned int *pNumPixelValues, float *pPixelValueMean, int **ppHistogram)
Retrieves all available information of an image statistics channel.

11.15.1 Detailed Description

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.

11.15.2 Function Documentation

11.15.2.1 spinImageStatisticsCreate()

```
SPINNAKERC_API spinImageStatisticsCreate (
    spinImageStatistics * phStatistics )
```

Creates an image statistics context.

Parameters

<i>phStatistics</i>	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

11.15.2.2 spinImageStatisticsDestroy()

```
SPINNAKERC_API spinImageStatisticsDestroy (
    spinImageStatistics hStatistics )
```

Destroys an image statistics context.

See also

[spinError](#)

Parameters

<i>hStatistics</i>	The image statistics context to destroy
--------------------	---

Returns

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

11.15.2.3 spinImageStatisticsDisableAll()

```
SPINNAKERC_API spinImageStatisticsDisableAll (
    spinImageStatistics hStatistics )
```

Disables all channels of an image statistics context.

See also

[spinError](#)

Parameters

<i>hStatistics</i>	The image statistics context to disable all channels
--------------------	--

Returns

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

11.15.2.4 spinImageStatisticsEnableAll()

```
SPINNAKERC_API spinImageStatisticsEnableAll (
    spinImageStatistics hStatistics )
```

Enables all channels of an image statistics context.

See also

[spinError](#)

Parameters

<i>hStatistics</i>	The image statistics context to enable all channels
--------------------	---

Returns

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

11.15.2.5 spinImageStatisticsEnableGreyOnly()

```
SPINNAKERC_API spinImageStatisticsEnableGreyOnly (
    spinImageStatistics hStatistics )
```

Disables all channels of an image statistics context except grey-scale.

See also

[spinError](#)

Parameters

<i>hStatistics</i>	The image statistics context to enable only grey
--------------------	--

Returns

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

11.15.2.6 spinImageStatisticsEnableHslOnly()

```
SPINNAKERC_API spinImageStatisticsEnableHslOnly (  
    spinImageStatistics hStatistics )
```

Disables all channels of an image statistics context except hue, saturation, and lightness.

See also

[spinError](#)

Parameters

<i>hStatistics</i>	The image statistics context to enable only HSL
--------------------	---

Returns

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

11.15.2.7 spinImageStatisticsEnableRgbOnly()

```
SPINNAKERC_API spinImageStatisticsEnableRgbOnly (  
    spinImageStatistics hStatistics )
```

Disables all channels of an image statistics context except red, blue, and green.

See also

[spinError](#)

Parameters

<i>hStatistics</i>	The image statistics context to enable only RGB
--------------------	---

Returns

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

11.15.2.8 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

<i>hStatistics</i>	The image statistics context of the channel
<i>channel</i>	The channel of the information to retrieve
<i>pRangeMin</i>	The unsigned integer pointer in which the minimum value of the range is returned
<i>pRangeMax</i>	The unsigned integer pointer in which the maximum value of the range is returned
<i>pPixelValueMin</i>	The unsigned integer pointer in which the minimum pixel value of the range is returned
<i>pPixelValueMax</i>	The unsigned integer pointer in which the maximum pixel value of the range is returned
<i>pNumPixelValues</i>	The unsigned integer pointer in which the number of pixel values is returned
<i>pPixelValueMean</i>	The float pointer in which the mean pixel value is returned
<i>ppiHistogram</i>	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

11.15.2.9 spinImageStatisticsGetChannelStatus()

```
SPINNAKERC_API spinImageStatisticsGetChannelStatus (
    spinImageStatistics hStatistics,
    spinStatisticsChannel channel,
    bool8_t * pbEnabled )
```

Checks whether an image statistics context is enabled.

See also

[spinError](#)

Parameters

<i>hStatistics</i>	The image statistics context of the channel
<i>channel</i>	The channel to check
<i>pbEnabled</i>	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

11.15.2.10 `spinImageStatisticsGetHistogram()`

```
SPINNAKERC_API spinImageStatisticsGetHistogram (
    spinImageStatistics hStatistics,
    spinStatisticsChannel channel,
    int ** ppHistogram )
```

Retrieves a histogram of an image statistics channel.

See also

[spinError](#)

Parameters

<i>hStatistics</i>	The image statistics context of the channel
<i>channel</i>	The channel of the histogram to be returned
<i>pHistogram</i>	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

11.15.2.11 `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

<i>hStatistics</i>	The image statistics context of the channel
<i>channel</i>	The channel of the mean pixel value to be retrieved
<i>pMean</i>	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

11.15.2.12 `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

<i>hStatistics</i>	The image statistics context of the channel
<i>channel</i>	The channel where the pixel values to be counted are
<i>iNumValues</i>	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

11.15.2.13 `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

<i>hStatistics</i>	The image statistics context of the channel
<i>channel</i>	The channel of the pixel value range to retrieve
<i>pMin</i>	The unsigned integer pointer in which the minimum value of the pixel value range is returned
<i>pMax</i>	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

11.15.2.14 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

<i>hStatistics</i>	The image statistics context of the channel
<i>channel</i>	The channel of the range to retrieve
<i>pMin</i>	The unsigned integer pointer in which the minimum value of the range is returned
<i>pMax</i>	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

11.15.2.15 spinImageStatisticsSetChannelStatus()

```
SPINNAKERC_API spinImageStatisticsSetChannelStatus (
    spinImageStatistics hStatistics,
    spinStatisticsChannel channel,
    bool8_t bEnable )
```

Sets the status of an image statistics channel.

See also

[spinError](#)

Parameters

<i>hStatistics</i>	The image statistics context of the channel
<i>channel</i>	The channel to enable/disable
<i>bEnable</i>	The boolean value to set; true enables, false disables

Returns

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

11.16 Logging Event Data Access

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

Functions

- [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 *pBufLen)
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.

11.16.1 Detailed Description

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

11.16.2 Function Documentation

11.16.2.1 spinLogDataGetCategoryName()

```
SPINNAKERC_API spinLogDataGetCategoryName (
    spinLogEventData hLogEventData,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the category name of a log event.

See also

[spinError](#)

Parameters

<i>hLogEventData</i>	The log event data received from the log event
<i>pBuf</i>	The c-string character buffer in which the category name of the log event is returned
<i>pBufLen</i>	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

11.16.2.2 `spinLogDataGetLogMessage()`

```
SPINNAKERC_API spinLogDataGetLogMessage (
    spinLogEventData hLogEventData,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the log message of a log event.

See also

[spinError](#)

Parameters

<i>hLogEventData</i>	The log event data received from the log event
<i>pBuf</i>	The c-string character buffer in which the log message of the log event is returned
<i>pBufLen</i>	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

11.16.2.3 `spinLogDataGetNDC()`

```
SPINNAKERC_API spinLogDataGetNDC (
    spinLogEventData hLogEventData,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the NDC of a log event.

See also

[spinError](#)

Parameters

<i>hLogEventData</i>	The log event data received from the log event
<i>pBuf</i>	The c-string character buffer in which the NDC of the log event is returned
<i>pBufLen</i>	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

11.16.2.4 `spinLogDataGetPriority()`

```
SPINNAKERC_API spinLogDataGetPriority (
    spinLogEventData hLogEventData,
    int64_t * pValue )
```

Retrieves the priority of a log event.

See also

[spinError](#)

Parameters

<i>hLogEventData</i>	The log event data received from the log event
<i>pValue</i>	The integer pointer in which the priority value is returned

Returns

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

11.16.2.5 `spinLogDataGetPriorityName()`

```
SPINNAKERC_API spinLogDataGetPriorityName (
    spinLogEventData hLogEventData,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the priority name of a log event.

See also

[spinError](#)

Parameters

<i>hLogEventData</i>	The log event data received from the log event
<i>pBuf</i>	The c-string character buffer in which the priority name of the log event is returned
<i>pBufLen</i>	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

11.16.2.6 `spinLogDataGetThreadName()`

```
SPINNAKERC_API spinLogDataGetThreadName (  
    spinLogEventData hLogEventData,  
    char * pBuf,  
    size_t * pBufLen )
```

Retrieves the thread name of a log event.

See also

[spinError](#)

Parameters

<i>hLogEventData</i>	The log event data received from the log event
<i>pBuf</i>	The c-string character buffer in which the thread name of the log event is returned
<i>pBufLen</i>	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

11.16.2.7 `spinLogDataGetTimestamp()`

```
SPINNAKERC_API spinLogDataGetTimestamp (  
    spinLogEventData hLogEventData,  
    char * pBuf,  
    size_t * pBufLen )
```

Retrieves the timestamp of a log event.

See also

[spinError](#)

Parameters

<i>hLogEventData</i>	The log event data received from the log event
<i>pBuf</i>	The c-string character buffer in which the timestamp of the log event is returned
<i>pBufLen</i>	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

11.17 Device Event Data Access

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

Functions

- [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.

11.17.1 Detailed Description

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

11.17.2 Function Documentation

11.17.2.1 spinDeviceEventGetId()

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

Retrieves the event ID of a device event.

See also

[spinError](#)

Parameters

<i>hDeviceEventData</i>	The log event data received from the log event
<i>pEventId</i>	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

11.17.2.2 spinDeviceEventGetName()

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

Retrieves the event name of a device event.

See also

[spinError](#)

Parameters

<i>hDeviceEventData</i>	The log event data received from the log event
<i>pBuf</i>	The c-string character buffer in which the name of the device event is returned
<i>pBufLen</i>	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

11.17.2.3 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

<i>hDeviceEventData</i>	The log event data received from the log event
<i>pBuf</i>	The unsigned integer pointer in which the event payload is returned
<i>pBufSize</i>	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

11.17.2.4 spinDeviceEventGetPayloadDataSize()

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

Retrieves the payload data size of a device event.

See also

[spinError](#)

Parameters

<i>hDeviceEventData</i>	The log event data received from the log event
<i>pBufSize</i>	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

11.18 Chunk data access

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

Functions

- [SPINNAKERC_API spinImageChunkDataGetIntValue](#) ([spinImage](#) hImage, const char *pName, int64_t *pValue)
- [SPINNAKERC_API spinImageChunkDataGetFloatValue](#) ([spinImage](#) hImage, const char *pName, double *pValue)

11.18.1 Detailed Description

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

11.18.2 Function Documentation

11.18.2.1 spinImageChunkDataGetFloatValue()

```
SPINNAKERC_API spinImageChunkDataGetFloatValue (  
    spinImage hImage,  
    const char * pName,  
    double * pValue )
```

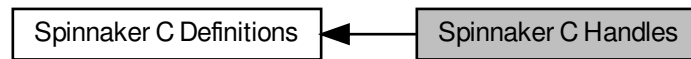
11.18.2.2 spinImageChunkDataGetIntValue()

```
SPINNAKERC_API spinImageChunkDataGetIntValue (  
    spinImage hImage,  
    const char * pName,  
    int64_t * pValue )
```

11.19 Spinnaker C Handles

Spinnaker C handle definitions.

Collaboration diagram for Spinnaker C Handles:



Typedefs

- 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 * [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 * [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.

11.19.1 Detailed Description

Spinnaker C handle definitions.

11.19.2 Typedef Documentation

11.19.2.1 spinCamera

```
typedef void* spinCamera
```

Handle for camera functionality.

Created by calling [spinCameraListGet\(\)](#), which requires a call to [spinCameraRelease\(\)](#) to release.

11.19.2.2 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.

11.19.2.3 spinDeviceArrivalEventHandler

```
typedef void* spinDeviceArrivalEventHandler
```

Handle for arrival event handler functionality.

Created by calling [spinArrivalEventCreate\(\)](#), which requires a call to [spinDeviceArrivalEventHandlerDestroy\(\)](#) to destroy.

11.19.2.4 spinDeviceEventData

```
typedef void* spinDeviceEventData
```

Handle for device event data functionality.

Received in device event function. No need to release, clear, or destroy.

11.19.2.5 spinDeviceEventHandler

```
typedef void* spinDeviceEventHandler
```

Handle for device event handler functionality.

Created by calling [spinDeviceEventHandlerCreate\(\)](#), which requires a call to [spinDeviceEventHandlerDestroy\(\)](#) to destroy.

11.19.2.6 spinDeviceRemovalEventHandler

```
typedef void* spinDeviceRemovalEventHandler
```

Handle for removal event handler functionality.

Created by calling [spinDeviceRemovalEventHandlerCreate\(\)](#), which requires a call to [spinDeviceRemovalEventHandlerDestroy\(\)](#) to destroy.

11.19.2.7 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.

11.19.2.8 spinImageEventHandler

```
typedef void* spinImageEventHandler
```

Handle for image event handler functionality.

Created by calling [spinImageEventHandlerCreate\(\)](#), which requires a call to [spinImageEventHandlerDestroy\(\)](#) to destroy.

11.19.2.9 spinImageStatistics

```
typedef void* spinImageStatistics
```

Handle for image statistics functionality.

Created by calling [spinImageStatisticsCreate\(\)](#), which requires a call to [spinImageStatisticsDestroy\(\)](#) to destroy.

11.19.2.10 spinInterface

```
typedef void* spinInterface
```

Handle for interface functionality.

Created by calling [spinInterfaceListGet\(\)](#), which requires a call to [spinInterfaceRelease\(\)](#) to release.

11.19.2.11 spinInterfaceEventHandler

```
typedef void* spinInterfaceEventHandler
```

Handle for interface event handler functionality.

Created by calling [spinInterfaceEventHandlerCreate\(\)](#), which requires a call to [spinInterfaceEventHandlerDestroy\(\)](#) to destroy.

11.19.2.12 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.

11.19.2.13 spinLogEventData

```
typedef void* spinLogEventData
```

Handle for logging event data functionality.

Received in log event function. No need to release, clear, or destroy.

11.19.2.14 spinLogEventHandler

```
typedef void* spinLogEventHandler
```

Handle for logging event handler functionality.

Created by calling [spinLogEventHandlerCreate\(\)](#), which requires a call to [spinLogEventHandlerDestroy\(\)](#) to destroy.

11.19.2.15 spinSystem

```
typedef void* spinSystem
```

Handle for system functionality.

Created by calling [spinSystemGetInstance\(\)](#), which requires a call to [spinSystemReleaseInstance\(\)](#) to release.

11.19.2.16 spinVideo

```
typedef void* spinVideo
```

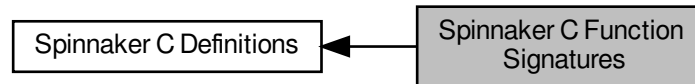
Handle for video recording functionality.

Created by calling [spinVideoOpenUncompressed\(\)](#), [spinVideoOpenMJPEG\(\)](#), and [spinVideoOpenH264\(\)](#), which require a call to [spinVideoClose\(\)](#) to destroy.

11.20 Spinnaker C Function Signatures

Spinnaker C function signature definitions.

Collaboration diagram for Spinnaker C Function Signatures:



Typedefs

- typedef void(* [spinDeviceEventFunction](#)) (const [spinDeviceEventData](#) hEventData, const char *pEventName, void *pUserData)
- Function signatures are used to create and trigger callbacks and events.*
- typedef void(* [spinImageEventFunction](#)) (const [spinImage](#) hImage, void *pUserData)
 - typedef void(* [spinArrivalEventFunction](#)) (uint64_t deviceSerialNumber, void *pUserData)
 - typedef void(* [spinRemovalEventFunction](#)) (uint64_t deviceSerialNumber, void *pUserData)
 - typedef void(* [spinLogEventFunction](#)) (const [spinLogEventData](#) hEventData, void *pUserData)

11.20.1 Detailed Description

Spinnaker C function signature definitions.

11.20.2 Typedef Documentation

11.20.2.1 spinArrivalEventFunction

```
typedef void(* spinArrivalEventFunction) (uint64_t deviceSerialNumber, void *pUserData)
```

11.20.2.2 spinDeviceEventFunction

```
typedef void(* spinDeviceEventFunction) (const spinDeviceEventData hEventData, const char *pEventName, void *pUserData)
```

Function signatures are used to create and trigger callbacks and events.

11.20.2.3 spinImageEventFunction

```
typedef void(* spinImageEventFunction) (const spinImage hImage, void *pUserData)
```

11.20.2.4 spinLogEventFunction

```
typedef void(* spinLogEventFunction) (const spinLogEventData hEventData, void *pUserData)
```

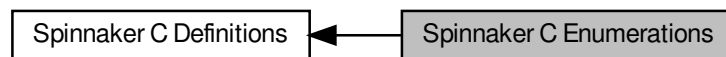
11.20.2.5 spinRemovalEventFunction

```
typedef void(* spinRemovalEventFunction) (uint64_t deviceSerialNumber, void *pUserData)
```

11.21 Spinnaker C Enumerations

Spinnaker C enumeration definitions.

Collaboration diagram for Spinnaker C Enumerations:



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,
    GENICAM_ERR_INVALID_ARGUMENT = -2001,
    GENICAM_ERR_OUT_OF_RANGE = -2002,
    GENICAM_ERR_PROPERTY = -2003,
    GENICAM_ERR_RUN_TIME = -2004,
    GENICAM_ERR_LOGICAL = -2005,
    GENICAM_ERR_ACCESS = -2006,
    GENICAM_ERR_TIMEOUT = -2007,
    GENICAM_ERR_DYNAMIC_CAST = -2008,
    GENICAM_ERR_GENERIC = -2009,
    GENICAM_ERR_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` {
`DEFAULT`,
`NO_COLOR_PROCESSING`,
`NEAREST_NEIGHBOR`,
`NEAREST_NEIGHBOR_AVG`,
`BILINEAR`,
`EDGE_SENSING`,
`HQ_LINEAR`,
`IPP`,
`DIRECTIONAL_FILTER`,
`RIGOROUS`,
`WEIGHTED_DIRECTIONAL_FILTER` }

Color processing algorithms.

- enum `spinStatisticsChannel` {
`GREY`,
`RED`,
`GREEN`,
`BLUE`,
`HUE`,
`SATURATION`,
`LIGHTNESS`,
`NUM_STATISTICS_CHANNELS` }

Channels that allow statistics to be calculated.

- enum `spinImageFileFormat` {
`FROM_FILE_EXT` = -1,
`PGM`,
`PPM`,
`BMP`,
`JPEG`,
`JPEG2000`,
`TIFF`,
`PNG`,
`RAW`,
`IMAGE_FILE_FORMAT_FORCE_32BITS` = 0x7FFFFFFF }

File formats to be used for saving images to disk.

- enum `spinPixelFormatNamespaceID` {
`SPINNAKER_PIXELFORMAT_NAMESPACE_UNKNOWN` = 0,
`SPINNAKER_PIXELFORMAT_NAMESPACE_GEV` = 1,
`SPINNAKER_PIXELFORMAT_NAMESPACE_IIDC` = 2,
`SPINNAKER_PIXELFORMAT_NAMESPACE_PFNC_16BIT` = 3,
`SPINNAKER_PIXELFORMAT_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` {
`IMAGE_UNKNOWN_ERROR` = -1,
`IMAGE_NO_ERROR` = 0,
`IMAGE_CRC_CHECK_FAILED` = 1,
`IMAGE_DATA_OVERFLOW` = 2,
`IMAGE_MISSING_PACKETS`,
`IMAGE_LEADER_BUFFER_SIZE_INCONSISTENT`,
`IMAGE_TRAILER_BUFFER_SIZE_INCONSISTENT`,

```

IMAGE_PACKETID_INCONSISTENT,
IMAGE_MISSING_LEADER = 7,
IMAGE_MISSING_TRAILER = 8,
IMAGE_DATA_INCOMPLETE = 9,
IMAGE_INFO_INCONSISTENT = 10,
IMAGE_CHUNK_DATA_INVALID = 11,
IMAGE_NO_SYSTEM_RESOURCES = 12 }

```

Status of images returned from `spinImageGetStatus()` call.

- enum `spinnakerLogLevel` {
`LOG_LEVEL_OFF` = -1,
`LOG_LEVEL_FATAL` = 0,
`LOG_LEVEL_ALERT` = 100,
`LOG_LEVEL_CRIT` = 200,
`LOG_LEVEL_ERROR` = 300,
`LOG_LEVEL_WARN` = 400,
`LOG_LEVEL_NOTICE` = 500,
`LOG_LEVEL_INFO` = 600,
`LOG_LEVEL_DEBUG` = 700,
`LOG_LEVEL_NOTSET` = 800 }

log levels

- enum `spinPayloadTypeInfoIds` {
`PAYLOAD_TYPE_UNKNOWN` = 0,
`PAYLOAD_TYPE_IMAGE` = 1,
`PAYLOAD_TYPE_RAW_DATA` = 2,
`PAYLOAD_TYPE_FILE` = 3,
`PAYLOAD_TYPE_CHUNK_DATA` = 4,
`PAYLOAD_TYPE_JPEG` = 5,
`PAYLOAD_TYPE_JPEG2000` = 6,
`PAYLOAD_TYPE_H264` = 7,
`PAYLOAD_TYPE_CHUNK_ONLY` = 8,
`PAYLOAD_TYPE_DEVICE_SPECIFIC` = 9,
`PAYLOAD_TYPE_MULTI_PART` = 10,
`PAYLOAD_TYPE_CUSTOM_ID` = 1000,
`PAYLOAD_TYPE_EXTENDED_CHUNK` = 1001,
`PAYLOAD_TYPE_LOSSLESS_COMPRESSED` = 1002,
`PAYLOAD_TYPE_LOSSY_COMPRESSED` = 1003,
`PAYLOAD_TYPE_JPEG_LOSSLESS_COMPRESSED` = 1004,
`PAYLOAD_TYPE_CHUNK_DATA_LOSSLESS_COMPRESSED` = 1005,
`PAYLOAD_TYPE_CHUNK_DATA_LOSSY_COMPRESSED` = 1006 }

11.21.1 Detailed Description

Spinnaker C enumeration definitions.

11.21.2 Enumeration Type Documentation

11.21.2.1 `spinColorProcessingAlgorithm`

```
enum spinColorProcessingAlgorithm
```

Color processing algorithms.

Please refer to our knowledge base at article at <https://www.flir.com/support-center/iis/machine-vision/kn> for complete details for each algorithm.

Enumerator

DEFAULT	Default method.
NO_COLOR_PROCESSING	No color processing.
NEAREST_NEIGHBOR	Fastest but lowest quality. Equivalent to FLYCAPTURE_NEAREST_NEIGHBOR_FAST in FlyCapture.
NEAREST_NEIGHBOR_AVG	Nearest Neighbor with averaged green pixels. Higher quality but slower compared to nearest neighbor without averaging.
BILINEAR	Weighted average of surrounding 4 pixels in a 2x2 neighborhood.
EDGE_SENSING	Weights surrounding pixels based on localized edge orientation.
HQ_LINEAR	Well-balanced speed and quality.
IPP	Multi-threaded with similar results to edge sensing.
DIRECTIONAL_FILTER	Best quality but much faster than rigorous.
RIGOROUS	Slowest but produces good results.
WEIGHTED_DIRECTIONAL_FILTER	Weighted pixel average from different directions.

11.21.2.2 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	

Enumerator

SPINNAKER_ERR_INVALID_VALUE	
SPINNAKER_ERR_RESOURCE_EXHAUSTED	
SPINNAKER_ERR_OUT_OF_MEMORY	
SPINNAKER_ERR_BUSY	
GENICAM_ERR_INVALID_ARGUMENT	The error codes in the range of -2000 to -2999 are reserved for Gen API related errors.
GENICAM_ERR_OUT_OF_RANGE	
GENICAM_ERR_PROPERTY	
GENICAM_ERR_RUN_TIME	
GENICAM_ERR_LOGICAL	
GENICAM_ERR_ACCESS	
GENICAM_ERR_TIMEOUT	
GENICAM_ERR_DYNAMIC_CAST	
GENICAM_ERR_GENERIC	
GENICAM_ERR_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	
SPINNAKER_ERR_CUSTOM_ID	Error codes less than -10000 are reserved for user-defined custom errors.

11.21.2.3 spinImageFileFormat

```
enum spinImageFileFormat
```

File formats to be used for saving images to disk.

Enumerator

FROM_FILE_EXT	Determine file format from file extension.
PGM	Portable gray map.
PPM	Portable pixmap.
BMP	Bitmap.
JPEG	JPEG.
JPEG2000	JPEG 2000.
TIFF	Tagged image file format.
PNG	Portable network graphics.
RAW	Raw data.
IMAGE_FILE_FORMAT_FORCE_32BITS	

11.21.2.4 spinImageStatus

enum `spinImageStatus`

Status of images returned from `spinImageGetStatus()` call.

Enumerator

IMAGE_UNKNOWN_ERROR	Image has an unknown error.
IMAGE_NO_ERROR	Image is returned from <code>GetNextImage()</code> call without any errors.
IMAGE_CRC_CHECK_FAILED	Image failed CRC check.
IMAGE_DATA_OVERFLOW	Received more data than the size of the image.
IMAGE_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
IMAGE_LEADER_BUFFER_SIZE_INCONSISTENT	Image leader is incomplete. Could be caused by missing packet(s). See link above.
IMAGE_TRAILER_BUFFER_SIZE_INCONSISTENT	Image trailer is incomplete. Could be caused by missing packet(s). See link above.
IMAGE_PACKETID_INCONSISTENT	Image has an inconsistent packet id. Could be caused by missing packet(s). See link above.
IMAGE_MISSING_LEADER	Image leader is missing. Could be caused by missing packet(s). See link above.
IMAGE_MISSING_TRAILER	Image trailer is missing. Could be caused by missing packet(s). See link above.
IMAGE_DATA_INCOMPLETE	Image data is incomplete. Could be caused by missing packet(s). See link above.
IMAGE_INFO_INCONSISTENT	Image info is corrupted. Could be caused by missing packet(s). See link above.
IMAGE_CHUNK_DATA_INVALID	Image chunk data is invalid.
IMAGE_NO_SYSTEM_RESOURCES	Image cannot be processed due to lack of system resources.

11.21.2.5 spinnakerLogLevel

enum `spinnakerLogLevel`

log levels

Enumerator

LOG_LEVEL_OFF	
LOG_LEVEL_FATAL	

Enumerator

LOG_LEVEL_ALERT	
LOG_LEVEL_CRIT	
LOG_LEVEL_ERROR	
LOG_LEVEL_WARN	
LOG_LEVEL_NOTICE	
LOG_LEVEL_INFO	
LOG_LEVEL_DEBUG	
LOG_LEVEL_NOTSET	

11.21.2.6 spinPayloadTypeInfoIDs

```
enum spinPayloadTypeInfoIDs
```

Enumerator

PAYLOAD_TYPE_UNKNOWN	
PAYLOAD_TYPE_IMAGE	
PAYLOAD_TYPE_RAW_DATA	
PAYLOAD_TYPE_FILE	
PAYLOAD_TYPE_CHUNK_DATA	
PAYLOAD_TYPE_JPEG	
PAYLOAD_TYPE_JPEG2000	
PAYLOAD_TYPE_H264	
PAYLOAD_TYPE_CHUNK_ONLY	
PAYLOAD_TYPE_DEVICE_SPECIFIC	
PAYLOAD_TYPE_MULTI_PART	
PAYLOAD_TYPE_CUSTOM_ID	
PAYLOAD_TYPE_EXTENDED_CHUNK	
PAYLOAD_TYPE_LOSSLESS_COMPRESSED	
PAYLOAD_TYPE_LOSSY_COMPRESSED	
PAYLOAD_TYPE_JPEG_LOSSLESS_COMPRESSED	
PAYLOAD_TYPE_CHUNK_DATA_LOSSLESS_COMPRESSED	
PAYLOAD_TYPE_CHUNK_DATA_LOSSY_COMPRESSED	

11.21.2.7 spinPixelFormatNamespaceID

```
enum spinPixelFormatNamespaceID
```

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_PIXELFORMAT_NAMESPACE_UNKNOWN	
SPINNAKER_PIXELFORMAT_NAMESPACE_GEV	
SPINNAKER_PIXELFORMAT_NAMESPACE_IIDC	
SPINNAKER_PIXELFORMAT_NAMESPACE_PFNC_16BIT	
SPINNAKER_PIXELFORMAT_NAMESPACE_PFNC_32BIT	
SPINNAKER_PIXELFORMAT_NAMESPACE_CUSTOM_ID	

11.21.2.8 spinStatisticsChannel

enum [spinStatisticsChannel](#)

Channels that allow statistics to be calculated.

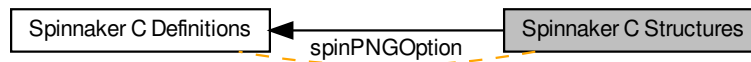
Enumerator

GREY	
RED	
GREEN	
BLUE	
HUE	
SATURATION	
LIGHTNESS	
NUM_STATISTICS_CHANNELS	

11.22 Spinnaker C Structures

Spinnaker C structure definitions.

Collaboration diagram for Spinnaker C Structures:



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.

Enumerations

- enum [spinCompressionMethod](#) {
[NONE](#) = 1,
[PACKBITS](#),
[DEFLATE](#),
[ADOBE_DEFLATE](#),
[CCITTFAX3](#),
[CCITTFAX4](#),
[LZW](#),
[JPG](#) }

Compression method used in saving TIFF images in the [spinTIFFOption](#) struct.

- enum [actionCommandStatus](#) {
[ACTION_COMMAND_STATUS_OK](#) = 0,
[ACTION_COMMAND_STATUS_NO_REF_TIME](#) = 0x8013,
[ACTION_COMMAND_STATUS_OVERFLOW](#) = 0x8015,
[ACTION_COMMAND_STATUS_ACTION_LATE](#) = 0x8016,
[ACTION_COMMAND_STATUS_ERROR](#) = 0x8FFF }

Possible Status Codes Returned from Action Command.

11.22.1 Detailed Description

Spinnaker C structure definitions.

11.22.2 Enumeration Type Documentation

11.22.2.1 actionCommandStatus

enum [actionCommandStatus](#)

Possible Status Codes Returned from Action Command.

Enumerator

ACTION_COMMAND_STATUS_OK	The device acknowledged the command.
ACTION_COMMAND_STATUS_NO_REF_TIME	
ACTION_COMMAND_STATUS_OVERFLOW	
ACTION_COMMAND_STATUS_ACTION_LATE	
ACTION_COMMAND_STATUS_ERROR	

11.22.2.2 spinCompressionMethod

enum [spinCompressionMethod](#)

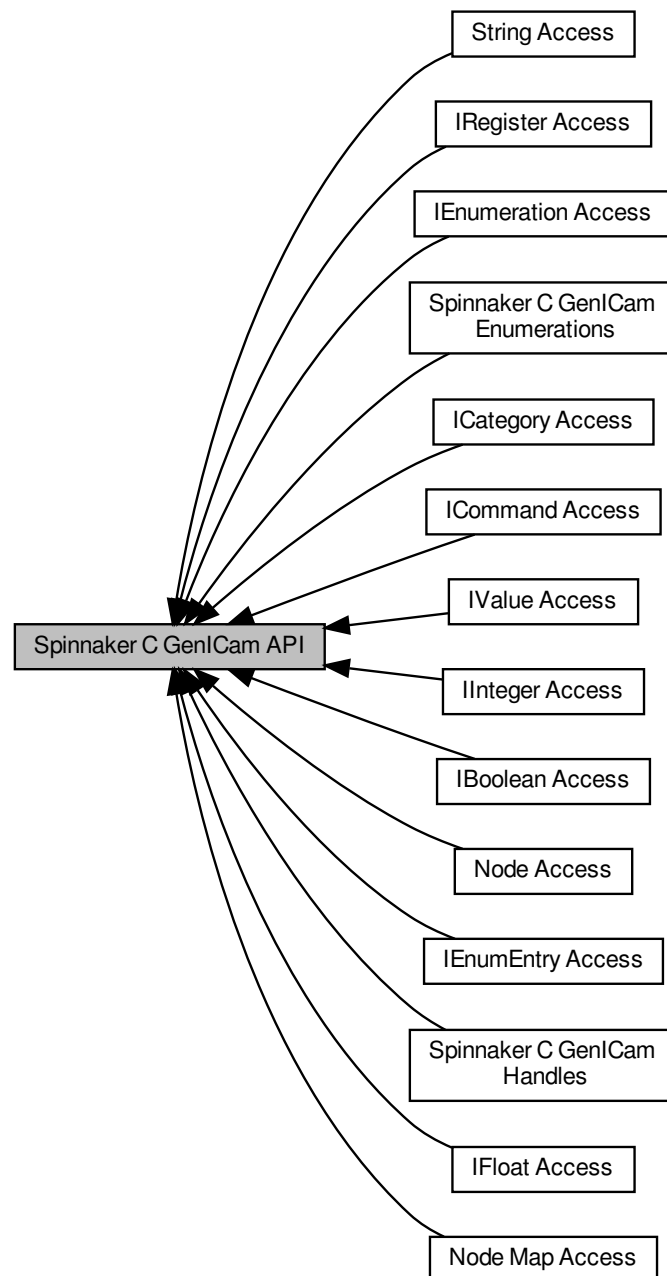
Compression method used in saving TIFF images in the [spinTIFFOption](#) struct.

Enumerator

NONE	
PACKBITS	
DEFLATE	
ADOBE_DEFLATE	
CCITTFAX3	
CCITTFAX4	
LZW	
JPG	

11.23 Spinnaker C GenICam API

Collaboration diagram for Spinnaker C GenICam API:



Modules

- [Node Map Access](#)

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

- [Node Access](#)

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

- [IValue Access](#)

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

- [String Access](#)

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

- [Integer Access](#)

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

- [IFloat Access](#)

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

- [IEnumeration Access](#)

The functions in this section provide access to enum nodes.

- [IEnumEntry Access](#)

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

- [IBoolean Access](#)

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

- [ICommand Access](#)

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

- [ICategory Access](#)

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

- [IRegister Access](#)

The functions in this section provide access to register nodes.

- [Spinnaker C GenICam Handles](#)

Handle definitions for Spinnaker C GenICam API.

- [Spinnaker C GenICam Enumerations](#)

Enumeration definitions for Spinnaker C GenICam API.

Functions

- [SPINNAKERC_API spinCategoryReleaseNode](#) ([spinNodeHandle](#) hCategoryNode, [spinNodeHandle](#) hFeature)

Releases the feature node from the category node.

11.23.1 Detailed Description

11.23.2 Function Documentation

11.23.2.1 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

<i>hCategoryNode</i>	The category node handle from which the feature node is retrieved
<i>hFeature</i>	The feature node handle to be released

Returns

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

11.24 Node Map Access

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

Collaboration diagram for Node Map Access:



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.

11.24.1 Detailed Description

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

This includes nodes, node counts, and polling.

11.24.2 Function Documentation

11.24.2.1 spinNodeMapGetNode()

```

SPINNAKERC_API spinNodeMapGetNode (
    spinNodeMapHandle hNodeMap,
    const char * pName,
    spinNodeHandle * phNode )
  
```

Retrieves a node from the nodemap by name.

See also

[spinError](#)

Parameters

<i>hNodeMap</i>	The node map where the node is
<i>pName</i>	The name of the node
<i>phNode</i>	The node handle pointer in which the node is returned

Returns

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

11.24.2.2 `spinNodeMapGetNodeByIndex()`

```
SPINNAKERC_API spinNodeMapGetNodeByIndex (
    spinNodeMapHandle hNodeMap,
    size_t index,
    spinNodeHandle * phNode )
```

Retrieves a node from the nodemap by index.

See also

[spinError](#)

Parameters

<i>hNodeMap</i>	The node map where the node is
<i>index</i>	The index of the node
<i>phNode</i>	The node handle pointer in which the node is returned

Returns

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

11.24.2.3 `spinNodeMapGetNumNodes()`

```
SPINNAKERC_API spinNodeMapGetNumNodes (
    spinNodeMapHandle hNodeMap,
    size_t * pValue )
```

Gets the number of nodes in the map.

See also

[spinError](#)

Parameters

<i>hNodeMap</i>	The node map where the nodes to be counted are
<i>pValue</i>	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

11.24.2.4 spinNodeMapPoll()

```
SPINNAKERC_API spinNodeMapPoll (
    spinNodeMapHandle hNodeMap,
    int64_t timestamp )
```

Fires nodes which have a polling time.

See also

[spinError](#)

Parameters

<i>hNodeMap</i>	The nodemap to poll
<i>timestamp</i>	The timestamp

Returns

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

11.24.2.5 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

<i>hNodeMap</i>	The node map from which the node handle is retrieved
<i>hNode</i>	The node handle to be released

Returns

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

11.25 Node Access

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

Collaboration diagram for Node Access:



Functions

- [SPINNAKERC_API spinNodesImplemented](#) ([spinNodeHandle](#) hNode, [bool8_t](#) *pbResult)
Checks whether a node is implemented.
- [SPINNAKERC_API spinNodesReadable](#) ([spinNodeHandle](#) hNode, [bool8_t](#) *pbResult)
Checks whether a node is readable.
- [SPINNAKERC_API spinNodesWritable](#) ([spinNodeHandle](#) hNode, [bool8_t](#) *pbResult)
Checks whether a node is writable.
- [SPINNAKERC_API spinNodesAvailable](#) ([spinNodeHandle](#) hNode, [bool8_t](#) *pbResult)
Checks whether a node is available.
- [SPINNAKERC_API spinNodesEqual](#) ([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](#) *pCachingMode)
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.

11.25.1 Detailed Description

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

This includes node properties and callback registration.

11.25.2 Function Documentation

11.25.2.1 spinNodeDeregisterCallback()

```
SPINNAKERC_API spinNodeDeregisterCallback (
    spinNodeHandle hNode,
    spinNodeCallbackHandle hCb )
```

Unregisters a callback from a node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The node from which to unregister the callback
<i>hCb</i>	The callback handle to unregister

Returns

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

11.25.2.2 spinNodeGetAccessMode()

```
SPINNAKERC_API spinNodeGetAccessMode (
    spinNodeHandle hNode,
    spinAccessMode * pAccessMode )
```

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

See also

[spinError](#)
[spinAccessMode](#)

Parameters

<i>hNode</i>	The node of the access mode to retrieve
<i>pAccessMode</i>	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

11.25.2.3 spinNodeGetCachingMode()

```
SPINNAKERC_API spinNodeGetCachingMode (
    spinNodeHandle hNode,
    spinCachingMode * pCachingMode )
```

Retrieves the caching mode of a node (as an enum, spinCachingMode)

See also

[spinError](#)
[spinCachingMode](#)

Parameters

<i>hNode</i>	The node of the caching mode to retrieve
<i>pCachingMode</i>	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

11.25.2.4 spinNodeGetDescription()

```
SPINNAKERC_API spinNodeGetDescription (
    spinNodeHandle hNode,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves a longer description of a node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The node of the description to retrieve
<i>pBuf</i>	The c-string character buffer in which the longer description of the node is returned
<i>pBufLen</i>	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

11.25.2.5 spinNodeGetDisplayName()

```
SPINNAKERC_API spinNodeGetDisplayName (
    spinNodeHandle hNode,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the display name of a node (whitespace possible)

See also

[spinError](#)

Parameters

<i>hNode</i>	The node of the display name to retrieve
<i>pBuf</i>	The c-string character buffer in which the display name of the node is returned
<i>pBufLen</i>	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

11.25.2.6 spinNodeGetImposedAccessMode()

```
SPINNAKERC_API spinNodeGetImposedAccessMode (
    spinNodeHandle hNode,
    spinAccessMode imposedAccessMode )
```

Retrieves the imposed access mode of a node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The node of the imposed access mode to retrieve
<i>imposedAccessMode</i>	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

11.25.2.7 spinNodeGetImposedVisibility()

```
SPINNAKERC_API spinNodeGetImposedVisibility (
    spinNodeHandle hNode,
    spinVisibility imposedVisibility )
```

Retrieves the imposed visibility of a node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The node of the visibility to impose
<i>imposedVisibility</i>	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

11.25.2.8 `spinNodeGetName()`

```
SPINNAKERC_API spinNodeGetName (
    spinNodeHandle hNode,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the name of a node (no whitespace)

See also

[spinError](#)

Parameters

<i>hNode</i>	The node of the name to retrieve
<i>pBuf</i>	The c-string character buffer in which the name of the node is returned
<i>pBufLen</i>	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

11.25.2.9 `spinNodeGetNamespace()`

```
SPINNAKERC_API spinNodeGetNamespace (
    spinNodeHandle hNode,
    spinNameSpace * pNamespace )
```

Retrieve the namespace of a node (as an enum, `spinNameSpace`)

See also

[spinError](#)
[spinNameSpace](#)

Parameters

<i>hNode</i>	The node of the namespace to retrieve
<i>pNamespace</i>	The namespace enum pointer in which the namespace is returned

Returns

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

11.25.2.10 spinNodeGetPollingTime()

```
SPINNAKERC_API spinNodeGetPollingTime (
    spinNodeHandle hNode,
    int64_t * pPollingTime )
```

Retrieve the polling time of a node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The node of the polling time to retrieve
<i>pPollingTime</i>	The integer pointer in which the polling time is returned

Returns

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

11.25.2.11 spinNodeGetToolTip()

```
SPINNAKERC_API spinNodeGetToolTip (
    spinNodeHandle hNode,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves a short description of a node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The node of the tooltip to retrieve
<i>pBuf</i>	The c-string character buffer in which the short description of the node is returned
<i>pBufLen</i>	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

11.25.2.12 spinNodeGetType()

```
SPINNAKERC_API spinNodeGetType (
    spinNodeHandle hNode,
    spinNodeType * pType )
```

Retrieves the type of a node (as an enum, spinNodeType)

See also

[spinError](#)
[spinNodeType](#)

Parameters

<i>hNode</i>	The node of the node type to retrieve
<i>pType</i>	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

11.25.2.13 spinNodeGetVisibility()

```
SPINNAKERC_API spinNodeGetVisibility (
    spinNodeHandle hNode,
    spinVisibility * pVisibility )
```

Retrieves the recommended visibility of a node (as an enum, spinVisibility)

See also

[spinError](#)
[spinVisibility](#)

Parameters

<i>hNode</i>	The node of the visibility to retrieve
<i>pVisibility</i>	The visibility enum pointer in which the visibility is returned

Returns

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

11.25.2.14 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

<i>hNode</i>	The node whose values may have changed
--------------	--

Returns

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

11.25.2.15 spinNodesAvailable()

```
SPINNAKERC_API spinNodeIsAvailable (
    spinNodeHandle hNode,
    bool8_t * pbResult )
```

Checks whether a node is available.

See also

[spinError](#)

Parameters

<i>hNode</i>	The node to check
<i>pbResult</i>	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

11.25.2.16 spinNodesEqual()

```
SPINNAKERC_API spinNodeIsEqual (
    spinNodeHandle hNodeFirst,
```

```
spinNodeHandle hNodeSecond,  
bool8_t * pbResult )
```

Checks whether two nodes are equal.

See also

[spinError](#)

Parameters

<i>hNodeFirst</i>	The first node to check
<i>hNodeSecond</i>	The second node to check
<i>pbResult</i>	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

11.25.2.17 spinNodeIsImplemented()

```
SPINNAKERC_API spinNodeIsImplemented (  
    spinNodeHandle hNode,  
    bool8_t * pbResult )
```

Checks whether a node is implemented.

See also

[spinError](#)

Parameters

<i>hNode</i>	The node to check
<i>pbResult</i>	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

11.25.2.18 spinNodeIsReadable()

```
SPINNAKERC_API spinNodeIsReadable (  
    spinNodeHandle hNode,  
    bool8_t * pbResult )
```

Checks whether a node is readable.

See also

[spinError](#)

Parameters

<i>hNode</i>	The node to check
<i>pbResult</i>	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

11.25.2.19 `spinNodesWritable()`

```
SPINNAKERC_API spinNodeIsWritable (
    spinNodeHandle hNode,
    bool8_t * pbResult )
```

Checks whether a node is writable.

See also

[spinError](#)

Parameters

<i>hNode</i>	The node to check
<i>pbResult</i>	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

11.25.2.20 `spinNodeRegisterCallback()`

```
SPINNAKERC_API spinNodeRegisterCallback (
    spinNodeHandle hNode,
    spinNodeCallbackFunction pCbFunction,
    spinNodeCallbackHandle * phCb )
```

Registers a callback to a node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The node on which to register the callback
<i>pCbFunction</i>	The function pointer of the function that will execute when the callback is triggered; must match signature "void spinNodeCallbackFunction(spinNodeHandle hNode)"
<i>phCb</i>	The callback handle pointer in which the callback is returned; used to unregister callbacks

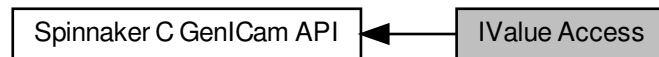
Returns

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

11.26 IValue Access

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

Collaboration diagram for IValue Access:



Functions

- [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 *pBufLen)
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.

11.26.1 Detailed Description

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.

11.26.2 Function Documentation

11.26.2.1 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

<i>hNode</i>	The node having its value changed
<i>pBuf</i>	The c-string of the value to set

Returns

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

11.26.2.2 `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

<i>hNode</i>	The node having its value changed
<i>bVerify</i>	The boolean of whether to verify the node
<i>pBuf</i>	The c-string of the value to set

Returns

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

11.26.2.3 `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

<i>hNode</i>	The node of the value to read
<i>pBuf</i>	The c-string character buffer in which the value of the node is returned
<i>pBufLen</i>	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

11.26.2.4 `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

<i>hNode</i>	The node of the value to read
<i>bVerify</i>	The boolean of whether to verify the node
<i>pBuf</i>	The c-string character buffer in which the value of the node is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

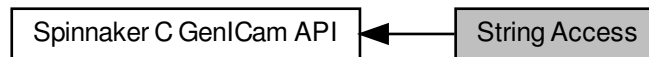
Returns

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

11.27 String Access

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

Collaboration diagram for String Access:



Functions

- [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](#) *pBufLen)
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.

11.27.1 Detailed Description

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.

11.27.2 Function Documentation

11.27.2.1 spinStringGetMaxLength()

```

SPINNAKERC_API spinStringGetMaxLength (
    spinNodeHandle hNode,
    int64_t * pValue )
  
```

Retrieves the maximum length of the c-string to be returned.

See also

[spinError](#)

Parameters

<i>hNode</i>	The string node of the length to retrieve
<i>pValue</i>	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

11.27.2.2 `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

<i>hNode</i>	The string node of the value to read
<i>pBuf</i>	The c-string character buffer in which the value of the node is returned
<i>pBufLen</i>	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

11.27.2.3 `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

<i>hNode</i>	The string node of the value to read
<i>bVerify</i>	The boolean of whether to verify the node
<i>pBuf</i>	The c-string character buffer in which the value of the node is returned
<i>pBufLen</i>	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

11.27.2.4 `spinStringSetValue()`

```
SPINNAKERC_API spinStringSetValue (  
    spinNodeHandle hNode,  
    const char * pBuf )
```

Sets the value of a string node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The string node having its value changed
<i>pBuf</i>	The c-string of the value to set

Returns

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

11.27.2.5 `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

<i>hNode</i>	The string node having its value changed
<i>bVerify</i>	The boolean of whether to verify the node
<i>pBuf</i>	The c-string of the value to set

Returns

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

11.28 Integer Access

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

Collaboration diagram for Integer Access:



Functions

- [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.

11.28.1 Detailed Description

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.

11.28.2 Function Documentation

11.28.2.1 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

<i>hNode</i>	The integer node of the increment to retrieve
<i>pValue</i>	The integer pointer in which the increment is returned

Returns

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

11.28.2.2 `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

<i>hNode</i>	The integer node of the maximum value to retrieve
<i>pValue</i>	The integer pointer in which the maximum value is returned

Returns

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

11.28.2.3 `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

<i>hNode</i>	The integer node of the minimum value to retrieve
<i>pValue</i>	The integer pointer in which the minimum value is returned

Returns

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

11.28.2.4 `spinIntegerGetRepresentation()`

```
SPINNAKERC_API spinIntegerGetRepresentation (
    spinNodeHandle hNode,
    spinRepresentation * pValue )
```

Retrieves the numerical representation of the value of a node; i.e.

linear, logarithmic, hexadecimal, MAC address, etc.

See also

[spinError](#)

Parameters

<i>hNode</i>	The integer node of the numerical representation to retrieve
<i>pValue</i>	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

11.28.2.5 `spinIntegerGetValue()`

```
SPINNAKERC_API spinIntegerGetValue (
    spinNodeHandle hNode,
    int64_t * pValue )
```

Retrieves the value of an integer node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The integer node of the value to read
<i>pValue</i>	The integer pointer in which the value is returned

Returns

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

11.28.2.6 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

<i>hNode</i>	The integer node of the value to read
<i>bVerify</i>	The boolean of whether to verify the node
<i>pValue</i>	The integer pointer in which the value is returned

Returns

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

11.28.2.7 spinIntegerSetValue()

```
SPINNAKERC_API spinIntegerSetValue (  
    spinNodeHandle hNode,  
    int64_t value )
```

Sets the value of an integer node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The integer node having its value changed
<i>value</i>	The integer value to set

Returns

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

11.28.2.8 `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

<i>hNode</i>	The integer node having its value changed
<i>bVerify</i>	The boolean of whether to verify the node
<i>value</i>	The integer value to set

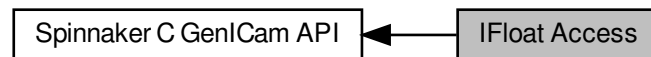
Returns

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

11.29 IFloat Access

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

Collaboration diagram for IFloat Access:



Functions

- [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.

11.29.1 Detailed Description

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.

11.29.2 Function Documentation

11.29.2.1 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

<i>hNode</i>	The float node of the maximum value to retrieve
<i>pValue</i>	The double pointer in which the maximum value is returned

Returns

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

11.29.2.2 `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

<i>hNode</i>	The float node of the minimum value to retrieve
<i>pValue</i>	The double pointer in which the minimum value is returned

Returns

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

11.29.2.3 `spinFloatGetRepresentation()`

```
SPINNAKERC_API spinFloatGetRepresentation (  
    spinNodeHandle hNode,  
    spinRepresentation * pValue )
```

Retrieves the numerical representation of the value of a node; i.e.

linear, logarithmic, hexadecimal, MAC address, etc.

See also

[spinError](#)

Parameters

<i>hNode</i>	The float node of the numerical representation to retrieve
<i>pValue</i>	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

11.29.2.4 `spinFloatGetUnit()`

```
SPINNAKERC_API spinFloatGetUnit (
    spinNodeHandle hNode,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the units of the float node value.

See also

[spinError](#)

Parameters

<i>hNode</i>	The float node of the units to retrieve
<i>pBuf</i>	The c-string character buffer in which the value units are returned
<i>pBufLen</i>	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

11.29.2.5 `spinFloatGetValue()`

```
SPINNAKERC_API spinFloatGetValue (
    spinNodeHandle hNode,
    double * pValue )
```

Retrieves the value of a float node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The float node of the value to read
<i>pValue</i>	The double pointer in which the value is returned

Returns

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

11.29.2.6 `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

<i>hNode</i>	The float node of the value to read
<i>pValue</i>	The double pointer in which the value is returned

Returns

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

11.29.2.7 `spinFloatSetValue()`

```
SPINNAKERC_API spinFloatSetValue (
    spinNodeHandle hNode,
    double value )
```

Sets the value of a float node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The float node having its value changed
<i>value</i>	The float value to set

Returns

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

11.29.2.8 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

<i>hNode</i>	The float node having its value changed
<i>bVerify</i>	The boolean of whether to verify the node
<i>value</i>	The float value to set

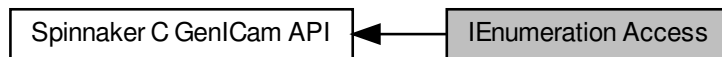
Returns

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

11.30 IEnumeration Access

The functions in this section provide access to enum nodes.

Collaboration diagram for IEnumeration Access:



Functions

- [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](#).

11.30.1 Detailed Description

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.

11.30.2 Function Documentation

11.30.2.1 spinEnumerationGetCurrentEntry()

```
SPINNAKERC_API spinEnumerationGetCurrentEntry (
    spinNodeHandle hEnumNode,
    spinNodeHandle * phEntry )
```

Retrieves the currently selected entry node from an enum node.

See also

[spinError](#)

Parameters

<i>hEnumNode</i>	The enum node from which the current entry node is retrieved
<i>phEntry</i>	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

11.30.2.2 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

<i>hEnumNode</i>	The enum node from which the entry node is retrieved
<i>index</i>	The index of the entry node
<i>phEntry</i>	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

11.30.2.3 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

<i>hEnumNode</i>	The enum node from which the entry node is retrieved
<i>pName</i>	The name of the entry node
<i>phEntry</i>	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

11.30.2.4 spinEnumerationGetNumEntries()

```
SPINNAKERC_API spinEnumerationGetNumEntries (
    spinNodeHandle hEnumNode,
    size_t * pValue )
```

Retrieves the number of entries of an enum node.

See also

[spinError](#)

Parameters

<i>hEnumNode</i>	The enum node where the entries to be counted are
<i>pValue</i>	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

11.30.2.5 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

<i>hEnumNode</i>	The enum node from which the current entry node is retrieved
<i>hEntry</i>	The entry node handle to be released

Returns

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

11.30.2.6 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

<i>hEnumNode</i>	The enum node have its entry changed
<i>value</i>	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

11.30.2.7 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 [Spinnaker↗](#) [DefsC.h](#).

See also

[spinEnumerationEntryGetIntValue\(\)](#)
[spinError](#)

Parameters

<i>hEnumNode</i>	The enum node having its entry changed
<i>value</i>	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

11.31 IEnumEntry Access

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

Collaboration diagram for IEnumEntry Access:



Functions

- [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](#) *pBufLen)
Retrieves the symbolic of an entry node as a c-string.

11.31.1 Detailed Description

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

11.31.2 Function Documentation

11.31.2.1 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

<i>hNode</i>	The entry node of the enum value to retrieve
<i>pValue</i>	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

11.31.2.2 `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

<i>hNode</i>	The entry node of the integer value to retrieve
<i>pValue</i>	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

11.31.2.3 `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

<i>hNode</i>	The entry node of the symbolic to retrieve
<i>pBuf</i>	The c-string character buffer in which the symbolic of the entry node is returned
<i>pBufLen</i>	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

11.32 IBoolean Access

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

Collaboration diagram for IBoolean Access:



Functions

- [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')

11.32.1 Detailed Description

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.

11.32.2 Function Documentation

11.32.2.1 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

<i>hNode</i>	The boolean node of the value to read
<i>pValue</i>	The boolean pointer in which the value is returned

Returns

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

11.32.2.2 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

<i>hNode</i>	The boolean node having its value changed
<i>value</i>	The boolean value to set

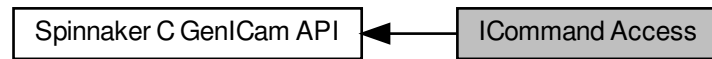
Returns

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

11.33 ICommand Access

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

Collaboration diagram for ICommand Access:



Functions

- [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.

11.33.1 Detailed Description

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

This includes node properties and callbacks.

11.33.2 Function Documentation

11.33.2.1 spinCommandExecute()

```
SPINNAKERC_API spinCommandExecute (
    spinNodeHandle hNode )
```

Executes the action associated to a command node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The command node to execute
--------------	-----------------------------

Returns

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

11.33.2.2 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

<i>hNode</i>	The command node to check
<i>pValue</i>	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

11.34 ICategory Access

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

Collaboration diagram for ICategory Access:



Functions

- [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.

11.34.1 Detailed Description

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

This includes node properties and callbacks.

11.34.2 Function Documentation

11.34.2.1 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

<i>hCategoryNode</i>	The category node of the node to retrieve
<i>index</i>	The index of the feature node
<i>phFeature</i>	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

11.34.2.2 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

<i>hCategoryNode</i>	The category node where the features to be counted are
<i>pValue</i>	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

11.35 IRegister Access

The functions in this section provide access to register nodes.

Collaboration diagram for IRegister Access:



Functions

- [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.

11.35.1 Detailed Description

The functions in this section provide access to register nodes.

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

11.35.2 Function Documentation

11.35.2.1 spinRegisterGet()

```
SPINNAKERC_API spinRegisterGet (
    spinNodeHandle hNode,
    uint8_t * pBuf,
    int64_t length )
```

Retrieves the value of a register node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The register node of the value to retrieve
<i>pBuf</i>	The unsigned integer buffer in which the value is returned
<i>length</i>	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

11.35.2.2 spinRegisterGetAddress()

```
SPINNAKERC_API spinRegisterGetAddress (
    spinNodeHandle hNode,
    int64_t * pAddress )
```

Retrieves the address of a register node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The register node of the address to retrieve
<i>pAddress</i>	The integer pointer in which the address is returned

Returns

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

11.35.2.3 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

<i>hNode</i>	The register node of the value to retrieve
<i>bVerify</i>	The boolean of whether to verify the node
<i>IgnoreCache</i>	The boolean of whether to ignore the cache
<i>pBuf</i>	The unsigned integer buffer in which the value is returned
<i>length</i>	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

11.35.2.4 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

<i>hNode</i>	The register node of the length to retrieve
<i>pLength</i>	The integer in which the number of bytes is returned

Returns

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

11.35.2.5 spinRegisterSet()

```
SPINNAKERC_API spinRegisterSet (  
    spinNodeHandle hNode,  
    const uint8_t * pBuf,  
    int64_t length )
```

Sets the value of a register node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The register node of the value to set
<i>pBuf</i>	The unsigned integer buffer of the value to set
<i>length</i>	The number of bytes of the value to set

Returns

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

11.35.2.6 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

<i>hNode</i>	The register node of the value to set
<i>bVerify</i>	The boolean of whether to verify the node
<i>pBuf</i>	The unsigned integer buffer of the value to set
<i>length</i>	The number of bytes of the value to set

Returns

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

11.35.2.7 spinRegisterSetReference()

```
SPINNAKERC_API spinRegisterSetReference (
    spinNodeHandle hNode,
    spinNodeHandle hRef )
```

Uses a second node as a reference for a register node.

See also

[spinError](#)

Parameters

<i>hNode</i>	The register node that houses the reference
<i>hRef</i>	The reference node

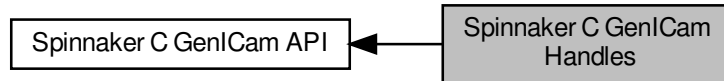
Returns

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

11.36 Spinnaker C GenICam Handles

Handle definitions for Spinnaker C GenICam API.

Collaboration diagram for Spinnaker C GenICam Handles:



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.

11.36.1 Detailed Description

Handle definitions for Spinnaker C GenICam API.

11.36.2 Typedef Documentation

11.36.2.1 spinNodeCallbackFunction

```
typedef void(* spinNodeCallbackFunction) (spinNodeHandle hNode)
```

Function signatures are used to create and trigger callbacks and events.

11.36.2.2 spinNodeCallbackHandle

```
typedef void* spinNodeCallbackHandle
```

Handle for callback functionality.

Created by calling [spinNodeRegisterCallback\(\)](#), which requires a call to [spinNodeUnregisterCallback\(\)](#) destroy.

11.36.2.3 spinNodeHandle

```
typedef void* spinNodeHandle
```

Handle for node functionality.

Created by calling [spinNodeMapGetNode\(\)](#). No need to release, clear, or destroy.

11.36.2.4 spinNodeMapHandle

```
typedef void* spinNodeMapHandle
```

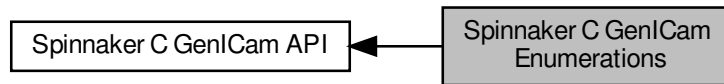
Handle for nodemap functionality.

Created by calling [spinCameraGetNodemap\(\)](#), [spinCameraGetTLDeviceNodeMap\(\)](#), [spinCameraGetTLStreamNodeMap\(\)](#) or [spinInterfaceGetTLNodeMap\(\)](#). No need to release, clear, or destroy.

11.37 Spinnaker C GenICam Enumerations

Enumeration definitions for Spinnaker C GenICam API.

Collaboration diagram for Spinnaker C GenICam Enumerations:



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 `spinEndianness` {
`BigEndian`,
`LittleEndian`,
`_UndefinedEndian` }
Endianness 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` = 0x80000000L }
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,
intfInteger,
intfBoolean,
intfCommand,
intfFloat,
intfString,
intfRegister,
intfCategory,
intfEnumeration,
intfEnumEntry,
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

11.37.1 Detailed Description

Enumeration definitions for Spinnaker C GenICam API.

11.37.2 Enumeration Type Documentation

11.37.2.1 spinAccessMode

```
enum spinAccessMode
```

Enumerator

NI	
NA	
WO	
RO	
RW	
_UndefinedAccesMode	
_CycleDetectAccesMode	

11.37.2.2 spinCachingMode

```
enum spinCachingMode
```

Enumerator

NoCache	
WriteThrough	
WriteAround	
_UndefinedCachingMode	

11.37.2.3 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

11.37.2.4 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.

11.37.2.5 spinIncMode

enum `spinIncMode`

typedef for increment mode

Enumerator

noIncrement	
fixedIncrement	
listIncrement	

11.37.2.6 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

11.37.2.7 spinInterfaceType

enum `spinInterfaceType`

typedef for interface type

Enumerator

intfIValue	
intfIBase	IValue interface

Enumerator

intfInteger	IBase interface
intfBoolean	IInteger interface
intfCommand	IBoolean interface
intfFloat	ICommand interface
intfString	IFloat interface
intfRegister	IString interface
intfCategory	IRegister interface
intfEnumeration	ICategory interface
intfEnumEntry	IEnumeration interface
intfIPort	IEnumEntry interface IPort interface

11.37.2.8 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

Enumerator

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

11.37.2.9 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.

11.37.2.10 spinNodeType

enum [spinNodeType](#)

Enumerator

ValueNode	
BaseNode	
IntegerNode	
BooleanNode	
FloatNode	
CommandNode	
StringNode	
RegisterNode	
EnumerationNode	
EnumEntryNode	
CategoryNode	
PortNode	
UnknownNode	

11.37.2.11 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	

11.37.2.12 spinSign

enum `spinSign`

Enumerator

Signed	
Unsigned	
_UndefinedSign	

11.37.2.13 spinSlope

enum `spinSlope`

typedef for fomula type

Enumerator

Increasing	
Decreasing	strictly monotonous increasing
Varying	strictly monotonous decreasing

Enumerator

Automatic	slope changes, e.g. at run-time
_UndefinedESlope	slope is determined automatically by probing the function Object is not yet initialized

11.37.2.14 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.

11.37.2.15 spinVisibility

enum `spinVisibility`

Enumerator

Beginner	
Expert	
Guru	
Invisible	
_UndefinedVisibility	

11.37.2.16 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

11.37.2.17 spinYesNo

enum `spinYesNo`

Defines the chices of a Yes/No alternaive.

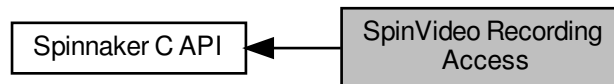
Enumerator

Yes	yes
No	no
_UndefinedYesNo	Object is not yet initialized.

11.38 SpinVideo Recording Access

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

Collaboration diagram for SpinVideo Recording Access:



Functions

- [SPINNAKERC_API spinVideoOpenUncompressed](#) ([spinVideo](#) *phSpinVideo, const char *pName, [spinAVIOOption](#) option)
- [SPINNAKERC_API spinVideoOpenMJPEG](#) ([spinVideo](#) *phSpinVideo, const char *pName, [spinMJPEGOption](#) 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)

11.38.1 Detailed Description

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

11.38.2 Function Documentation

11.38.2.1 spinVideoAppend()

```

SPINNAKERC_API spinVideoAppend (
    spinVideo hSpinVideo,
    spinImage hImage )
  
```

11.38.2.2 spinVideoClose()

```
SPINNAKERC_API spinVideoClose (
    spinVideo hSpinVideo )
```

11.38.2.3 spinVideoOpenH264()

```
SPINNAKERC_API spinVideoOpenH264 (
    spinVideo * phSpinVideo,
    const char * pName,
    spinH264Option option )
```

11.38.2.4 spinVideoOpenMJPG()

```
SPINNAKERC_API spinVideoOpenMJPG (
    spinVideo * phSpinVideo,
    const char * pName,
    spinMJPGOption option )
```

11.38.2.5 spinVideoOpenUncompressed()

```
SPINNAKERC_API spinVideoOpenUncompressed (
    spinVideo * phSpinVideo,
    const char * pName,
    spinAVIOption option )
```

11.38.2.6 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

<i>hSpinVideo</i>	The spin video recorder to append the image to
<i>size</i>	The maximum video file size in MB.

Returns

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

11.39 Transport Layer Enumerations

Collaboration diagram for Transport Layer Enumerations:



Enumerations

- enum `spinTLStreamTypeEnums` {
`StreamType_GigEVision`,
`StreamType_CameraLink`,
`StreamType_CameraLinkHS`,
`StreamType_CoaXPress`,
`StreamType_USB3Vision`,
`StreamType_Custom`,
`NUMSTREAMTYPE` }
- The enumeration definitions for transport layer nodes.*
- enum `spinTLStreamBufferCountModeEnums` {
`StreamBufferCountMode_Manual`,
`StreamBufferCountMode_Auto`,
`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 `spinTLGevCCPEnums` {
`GevCCP_EnumEntry_GevCCP_OpenAccess`,
`GevCCP_EnumEntry_GevCCP_ExclusiveAccess`,
`GevCCP_EnumEntry_GevCCP_ControlAccess`,
`NUMGEVCCP` }

- enum `spinTLGUIXMLLocationEnums` {
 `GUIXMLLocation_Device`,
 `GUIXMLLocation_Host`,
 `NUMGUIXMLLOCATION` }
- enum `spinTLGenICamXMLLocationEnums` {
 `GenICamXMLLocation_Device`,
 `GenICamXMLLocation_Host`,
 `NUMGENICAMXMLLOCATION` }
- enum `spinTLDeviceEndiannessMechanismEnums` {
 `DeviceEndiannessMechanism_Legacy`,
 `DeviceEndiannessMechanism_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` }

11.39.1 Detailed Description

11.39.2 Enumeration Type Documentation

11.39.2.1 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	

11.39.2.2 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	

11.39.2.3 spinTLDeviceEndiannessMechanismEnums

```
enum spinTLDeviceEndiannessMechanismEnums
```

< Identifies the endianness handling mode.

Enumerator

DeviceEndiannessMechanism_Legacy	Handling the device endianness according to GenICam Schema 1.0
DeviceEndiannessMechanism_Standard	Handling the device endianness according to GenICam Schema 1.1 and later
NUMDEVICEENDIANESSMECHANISM	

11.39.2.4 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	

11.39.2.5 spinTLFilterDriverStatusEnums

enum [spinTLFilterDriverStatusEnums](#)

< Reports whether FLIR Light Weight Filter Driver is enabled or not.

Enumerator

FilterDriverStatus_NotSupported	Not Supported
FilterDriverStatus_Disabled	FLIR Light Weight Filter Driver is disabled
FilterDriverStatus_Enabled	FLIR Light Weight Filter Driver is enabled
NUMFILTERDRIVERSTATUS	

11.39.2.6 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	

11.39.2.7 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	

11.39.2.8 spinTLGUIXMLLocationEnums

enum `spinTLGUIXMLLocationEnums`

< Sets the location to load GUI XML.

Enumerator

GUIXMLLocation_Device	Load XML from device
GUIXMLLocation_Host	Load XML from host
NUMGUIXMLLOCATION	

11.39.2.9 spinTLInterfaceTypeEnums

enum `spinTLInterfaceTypeEnums`

< Transport layer type of the interface.

Enumerator

InterfaceType_GigEVision	GigE Vision
InterfaceType_CameraLink	Camera Link
InterfaceType_CameraLinkHS	Camera Link High Speed
InterfaceType_CoaXPress	CoaXPress
InterfaceType_USB3Vision	USB3 Vision
InterfaceType_Custom	Custom transport layer
NUMINTERFACETYPE	

11.39.2.10 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	

11.39.2.11 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 are set by the user.
StreamBufferCountMode_Auto	DEPRECATED. The number of buffers used for the stream is automatically calculated based on the device frame rate.
NUMSTREAMBUFFERCOUNTMODE	

11.39.2.12 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).

Enumerator

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	

11.39.2.13 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	

11.39.2.14 spinTLTLTypeEnums

enum `spinTLTLTypeEnums`

< Transport layer type of the GenTL Producer implementation.

Enumerator

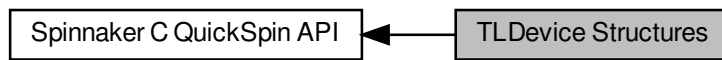
TLType_GigEVision	GigE Vision
TLType_CameraLink	Camera Link
TLType_CameraLinkHS	Camera Link High Speed
TLType_CoaXPRESS	CoaXPRESS

Enumerator

TLType_USB3Vision	USB3 Vision
TLType_Mixed	Different Interface modules of the GenTL Producer are of different types
TLType_Custom	Custom transport layer
NUMTLTYPE	

11.40 TLDevice Structures

Collaboration diagram for TLDevice Structures:



Data Structures

- struct [quickSpinTLDevice](#)

11.40.1 Detailed Description

11.41 TLInterface Structures

Collaboration diagram for TLInterface Structures:



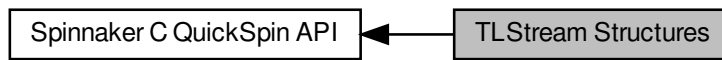
Data Structures

- struct [quickSpinTLInterface](#)

11.41.1 Detailed Description

11.42 TLStream Structures

Collaboration diagram for TLStream Structures:



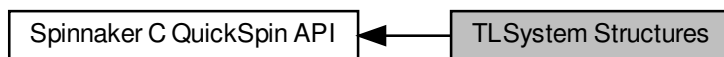
Data Structures

- struct [quickSpinTLStream](#)

11.42.1 Detailed Description

11.43 TLSystem Structures

Collaboration diagram for TLSystem Structures:



Data Structures

- struct [quickSpinTLSystem](#)

11.43.1 Detailed Description

Chapter 12

Data Structure Documentation

12.1 actionCommandResult Struct Reference

Action Command Result.

Data Fields

- unsigned int [DeviceAddress](#)
- [actionCommandStatus](#) 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

```
actionCommandStatus 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 CIConfiguration](#)
- [quickSpinEnumerationNode CITimeSlotsCount](#)
- [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 DeviceVendorName
- quickSpinStringNode DeviceModelName
- quickSpinEnumerationNode DeviceType
- quickSpinStringNode DeviceDisplayName
- quickSpinEnumerationNode DeviceAccessStatus
- quickSpinStringNode DeviceVersion
- quickSpinStringNode DeviceUserID
- quickSpinStringNode DeviceDriverVersion
- quickSpinBooleanNode DevicesUpdater
- quickSpinEnumerationNode GevCCP
- quickSpinEnumerationNode GUIXMLLocation
- quickSpinStringNode GUIXMLPath
- quickSpinEnumerationNode GenICamXMLLocation
- quickSpinStringNode GenICamXMLPath
- quickSpinIntegerNode GevDeviceIPAddress
- quickSpinIntegerNode GevDeviceSubnetMask
- quickSpinIntegerNode GevDeviceMACAddress
- quickSpinIntegerNode GevDeviceGateway
- quickSpinIntegerNode DeviceLinkSpeed
- quickSpinIntegerNode GevVersionMajor
- quickSpinIntegerNode GevVersionMinor
- quickSpinBooleanNode GevDeviceModelsBigEndian
- quickSpinIntegerNode GevDeviceReadAndWriteTimeout
- quickSpinIntegerNode GevDeviceMaximumRetryCount
- quickSpinIntegerNode GevDevicePort
- quickSpinCommandNode GevDeviceDiscoverMaximumPacketSize
- quickSpinIntegerNode GevDeviceMaximumPacketSize
- quickSpinBooleanNode GevDevicesWrongSubnet
- quickSpinCommandNode GevDeviceAutoForceIP
- quickSpinCommandNode GevDeviceForceIP
- quickSpinIntegerNode GevDeviceForceIPAddress
- quickSpinIntegerNode GevDeviceForceSubnetMask
- quickSpinIntegerNode GevDeviceForceGateway
- quickSpinBooleanNode DeviceMulticastMonitorMode
- quickSpinEnumerationNode DeviceEndiannessMechanism
- 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 DeviceCurrentSpeed

[quickSpinEnumerationNode](#) DeviceCurrentSpeed

12.3.1.3 DeviceDisplayName

[quickSpinStringNode](#) DeviceDisplayName

12.3.1.4 DeviceDriverVersion

[quickSpinStringNode](#) DeviceDriverVersion

12.3.1.5 DeviceEndiannessMechanism

[quickSpinEnumerationNode](#) DeviceEndiannessMechanism

12.3.1.6 DeviceID

[quickSpinStringNode](#) DeviceID

12.3.1.7 DeviceInstanceId

[quickSpinStringNode](#) DeviceInstanceId

12.3.1.8 DeviceIsUpdater

[quickSpinBooleanNode](#) DeviceIsUpdater

12.3.1.9 DeviceLinkSpeed

[quickSpinIntegerNode](#) DeviceLinkSpeed

12.3.1.10 DeviceLocation

[quickSpinStringNode](#) DeviceLocation

12.3.1.11 DeviceModelName

[quickSpinStringNode](#) DeviceModelName

12.3.1.12 DeviceMulticastMonitorMode

[quickSpinBooleanNode](#) DeviceMulticastMonitorMode

12.3.1.13 DevicePortId

[quickSpinStringNode](#) DevicePortId

12.3.1.14 DeviceSerialNumber

[quickSpinStringNode](#) DeviceSerialNumber

12.3.1.15 DeviceType

[quickSpinEnumerationNode](#) DeviceType

12.3.1.16 DeviceU3VProtocol

[quickSpinBooleanNode](#) DeviceU3VProtocol

12.3.1.17 DeviceUserID

[quickSpinStringNode](#) DeviceUserID

12.3.1.18 DeviceVendorName

[quickSpinStringNode](#) DeviceVendorName

12.3.1.19 DeviceVersion

[quickSpinStringNode](#) DeviceVersion

12.3.1.20 GenICamXMLLocation

[quickSpinEnumerationNode](#) GenICamXMLLocation

12.3.1.21 GenICamXMLPath

[quickSpinStringNode](#) GenICamXMLPath

12.3.1.22 GevCCP

[quickSpinEnumerationNode](#) GevCCP

12.3.1.23 GevDeviceAutoForceIP

[quickSpinCommandNode](#) GevDeviceAutoForceIP

12.3.1.24 GevDeviceDiscoverMaximumPacketSize

[quickSpinCommandNode](#) GevDeviceDiscoverMaximumPacketSize

12.3.1.25 GevDeviceForceGateway

[quickSpinIntegerNode](#) GevDeviceForceGateway

12.3.1.26 GevDeviceForceIP

[quickSpinCommandNode](#) GevDeviceForceIP

12.3.1.27 GevDeviceForceIPAddress

[quickSpinIntegerNode](#) GevDeviceForceIPAddress

12.3.1.28 GevDeviceForceSubnetMask

[quickSpinIntegerNode](#) GevDeviceForceSubnetMask

12.3.1.29 GevDeviceGateway

[quickSpinIntegerNode](#) GevDeviceGateway

12.3.1.30 GevDeviceIPAddress

[quickSpinIntegerNode](#) GevDeviceIPAddress

12.3.1.31 GevDeviceIsWrongSubnet

[quickSpinBooleanNode](#) GevDeviceIsWrongSubnet

12.3.1.32 GevDeviceMACAddress

[quickSpinIntegerNode](#) GevDeviceMACAddress

12.3.1.33 GevDeviceMaximumPacketSize

[quickSpinIntegerNode](#) GevDeviceMaximumPacketSize

12.3.1.34 GevDeviceMaximumRetryCount

[quickSpinIntegerNode](#) GevDeviceMaximumRetryCount

12.3.1.35 GevDeviceModelsBigEndian

[quickSpinBooleanNode](#) GevDeviceModeIsBigEndian

12.3.1.36 GevDevicePort

[quickSpinIntegerNode](#) GevDevicePort

12.3.1.37 GevDeviceReadAndWriteTimeout

[quickSpinIntegerNode](#) GevDeviceReadAndWriteTimeout

12.3.1.38 GevDeviceSubnetMask

[quickSpinIntegerNode](#) GevDeviceSubnetMask

12.3.1.39 GevVersionMajor

[quickSpinIntegerNode](#) GevVersionMajor

12.3.1.40 GevVersionMinor

[quickSpinIntegerNode](#) GevVersionMinor

12.3.1.41 GUIXMLLocation

[quickSpinEnumerationNode](#) [GUIXMLLocation](#)

12.3.1.42 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](#)
- [quickSpinIntegerNode](#) [StreamBufferCountManual](#)
- [quickSpinIntegerNode](#) [StreamBufferCountResult](#)
- [quickSpinIntegerNode](#) [StreamBufferCountMax](#)
- [quickSpinEnumerationNode](#) [StreamBufferCountMode](#)
- [quickSpinEnumerationNode](#) [StreamBufferHandlingMode](#)
- [quickSpinIntegerNode](#) [StreamAnnounceBufferMinimum](#)
- [quickSpinIntegerNode](#) [StreamAnnouncedBufferCount](#)
- [quickSpinIntegerNode](#) [StreamStartedFrameCount](#)
- [quickSpinIntegerNode](#) [StreamDeliveredFrameCount](#)
- [quickSpinIntegerNode](#) [StreamLostFrameCount](#)
- [quickSpinIntegerNode](#) [StreamInputBufferCount](#)
- [quickSpinIntegerNode](#) [StreamOutputBufferCount](#)
- [quickSpinBooleanNode](#) [StreamCRCCheckEnable](#)
- [quickSpinBooleanNode](#) [GevPacketResendMode](#)
- [quickSpinIntegerNode](#) [GevMaximumNumberResendRequests](#)
- [quickSpinIntegerNode](#) [GevPacketResendTimeout](#)
- [quickSpinBooleanNode](#) [StreamIsGrabbing](#)
- [quickSpinIntegerNode](#) [StreamChunkCountMaximum](#)
- [quickSpinIntegerNode](#) [StreamBufferAlignment](#)
- [quickSpinIntegerNode](#) [StreamDroppedFrameCount](#)
- [quickSpinIntegerNode](#) [GevTotalPacketCount](#)
- [quickSpinIntegerNode](#) [GevFailedPacketCount](#)
- [quickSpinIntegerNode](#) [GevResendPacketCount](#)
- [quickSpinIntegerNode](#) [StreamFailedBufferCount](#)
- [quickSpinIntegerNode](#) [GevResendRequestCount](#)
- [quickSpinIntegerNode](#) [StreamBlockTransferSize](#)

12.5.1 Field Documentation

12.5.1.1 `GevFailedPacketCount`

`quickSpinIntegerNode` `GevFailedPacketCount`

12.5.1.2 `GevMaximumNumberResendRequests`

`quickSpinIntegerNode` `GevMaximumNumberResendRequests`

12.5.1.3 `GevPacketResendMode`

`quickSpinBooleanNode` `GevPacketResendMode`

12.5.1.4 `GevPacketResendTimeout`

`quickSpinIntegerNode` `GevPacketResendTimeout`

12.5.1.5 `GevResendPacketCount`

`quickSpinIntegerNode` `GevResendPacketCount`

12.5.1.6 `GevResendRequestCount`

`quickSpinIntegerNode` `GevResendRequestCount`

12.5.1.7 `GevTotalPacketCount`

`quickSpinIntegerNode` `GevTotalPacketCount`

12.5.1.8 StreamAnnounceBufferMinimum

[quickSpinIntegerNode](#) StreamAnnounceBufferMinimum

12.5.1.9 StreamAnnouncedBufferCount

[quickSpinIntegerNode](#) StreamAnnouncedBufferCount

12.5.1.10 StreamBlockTransferSize

[quickSpinIntegerNode](#) StreamBlockTransferSize

12.5.1.11 StreamBufferAlignment

[quickSpinIntegerNode](#) StreamBufferAlignment

12.5.1.12 StreamBufferCountManual

[quickSpinIntegerNode](#) StreamBufferCountManual

12.5.1.13 StreamBufferCountMax

[quickSpinIntegerNode](#) StreamBufferCountMax

12.5.1.14 StreamBufferCountMode

[quickSpinEnumerationNode](#) StreamBufferCountMode

12.5.1.15 StreamBufferCountResult

[quickSpinIntegerNode](#) StreamBufferCountResult

12.5.1.16 StreamBufferHandlingMode

`quickSpinEnumerationNode` StreamBufferHandlingMode

12.5.1.17 StreamChunkCountMaximum

`quickSpinIntegerNode` StreamChunkCountMaximum

12.5.1.18 StreamCRCCheckEnable

`quickSpinBooleanNode` StreamCRCCheckEnable

12.5.1.19 StreamDeliveredFrameCount

`quickSpinIntegerNode` StreamDeliveredFrameCount

12.5.1.20 StreamDroppedFrameCount

`quickSpinIntegerNode` StreamDroppedFrameCount

12.5.1.21 StreamFailedBufferCount

`quickSpinIntegerNode` StreamFailedBufferCount

12.5.1.22 StreamID

`quickSpinStringNode` StreamID

12.5.1.23 StreamInputBufferCount

`quickSpinIntegerNode` StreamInputBufferCount

12.5.1.24 StreamIsGrabbing

[quickSpinBooleanNode](#) StreamIsGrabbing

12.5.1.25 StreamLostFrameCount

[quickSpinIntegerNode](#) StreamLostFrameCount

12.5.1.26 StreamOutputBufferCount

[quickSpinIntegerNode](#) StreamOutputBufferCount

12.5.1.27 StreamStartedFrameCount

[quickSpinIntegerNode](#) StreamStartedFrameCount

12.5.1.28 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](#) GenTLFNCVersionMajor

- [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](#)

12.6.1 Field Documentation

12.6.1.1 EnumerateGEVInterfaces

[quickSpinBooleanNode](#) [EnumerateGEVInterfaces](#)

12.6.1.2 EnumerateUSBInterfaces

[quickSpinBooleanNode](#) [EnumerateUSBInterfaces](#)

12.6.1.3 GenTLSFNCVersionMajor

[quickSpinIntegerNode](#) [GenTLSFNCVersionMajor](#)

12.6.1.4 GenTLSFNCVersionMinor

[quickSpinIntegerNode](#) [GenTLSFNCVersionMinor](#)

12.6.1.5 GenTLSFNCVersionSubMinor

[quickSpinIntegerNode](#) [GenTLSFNCVersionSubMinor](#)

12.6.1.6 GenTLVersionMajor

`quickSpinIntegerNode` GenTLVersionMajor

12.6.1.7 GenTLVersionMinor

`quickSpinIntegerNode` GenTLVersionMinor

12.6.1.8 GevInterfaceDefaultGateway

`quickSpinIntegerNode` GevInterfaceDefaultGateway

12.6.1.9 GevInterfaceDefaultIPAddress

`quickSpinIntegerNode` GevInterfaceDefaultIPAddress

12.6.1.10 GevInterfaceDefaultSubnetMask

`quickSpinIntegerNode` GevInterfaceDefaultSubnetMask

12.6.1.11 GevInterfaceMACAddress

`quickSpinIntegerNode` GevInterfaceMACAddress

12.6.1.12 GevVersionMajor

`quickSpinIntegerNode` GevVersionMajor

12.6.1.13 GevVersionMinor

`quickSpinIntegerNode` GevVersionMinor

12.6.1.14 InterfaceDisplayName

[quickSpinStringNode](#) InterfaceDisplayName

12.6.1.15 InterfaceID

[quickSpinStringNode](#) InterfaceID

12.6.1.16 InterfaceSelector

[quickSpinIntegerNode](#) InterfaceSelector

12.6.1.17 InterfaceUpdateList

[quickSpinCommandNode](#) InterfaceUpdateList

12.6.1.18 TLDisplayName

[quickSpinStringNode](#) TLDisplayName

12.6.1.19 TLFileName

[quickSpinStringNode](#) TLFileName

12.6.1.20 TLID

[quickSpinStringNode](#) TLID

12.6.1.21 TLModelName

[quickSpinStringNode](#) TLModelName

12.6.1.22 TLPath

`quickSpinStringNode` TLPath

12.6.1.23 TLType

`quickSpinEnumerationNode` TLType

12.6.1.24 TLVendorName

`quickSpinStringNode` TLVendorName

12.6.1.25 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 `reserved` [256]
Reserved for future use.

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 reserved

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

Reserved for future use.

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 spinMJPEGOption 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 [reserved](#) [256]

12.14.1 Detailed Description

Options for saving MJPG videos.

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

12.14.2 Field Documentation

12.14.2.1 frameRate

float frameRate

Frame rate of the stream.

12.14.2.2 quality

unsigned int quality

Image quality (1-100)

12.14.2.3 reserved

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

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

- [spinCompressionMethod](#) *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](#)

[spinCompressionMethod](#) *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:

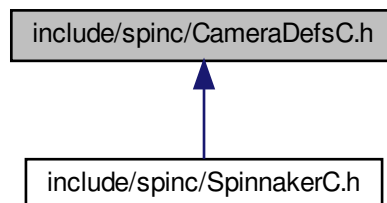
- include/spinc/[SpinnakerDefsC.h](#)

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_DEVICEREGISTERSENDIANNES` }
- 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_HistogramPeak](#),
[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,
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 spinCIConfigurationEnums {
    CIConfiguration_Base,
    CIConfiguration_Medium,

```

```

CIConfiguration_Full,
CIConfiguration_DualBase,
CIConfiguration_EightyBit,
NUM_CLCONFIGURATION }

• enum spinCITimeSlotsCountEnums {
    CITimeSlotsCount_One,
    CITimeSlotsCount_Two,
    CITimeSlotsCount_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 {`

```

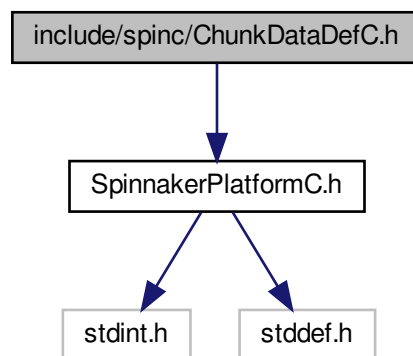
CxpPoCxpStatus_Auto,
CxpPoCxpStatus_Off,
CxpPoCxpStatus_Tripped,

```

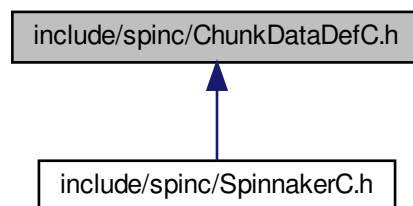
```
NUM_CXPPOCXPSTATUS }
```

### 13.9 include/spinc/ChunkDataDefC.h File Reference

Include dependency graph for ChunkDataDefC.h:



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



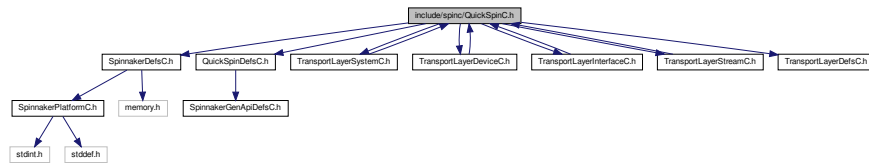
#### Data Structures

- struct [spinChunkData](#)

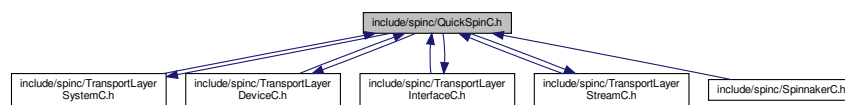
*The type of information that can be obtained from image chunk data.*

## 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:

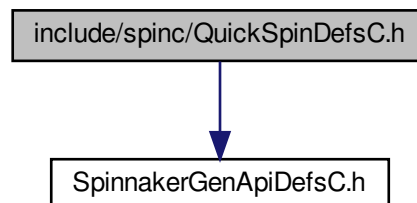


### Functions

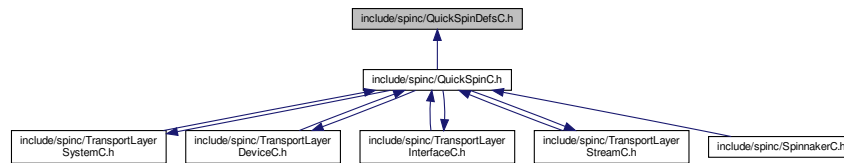
- [SPINNAKERC\\_API quickSpinInit](#) ([spinCamera](#) hCamera, [quickSpin](#) \*pQuickSpin)
- [SPINNAKERC\\_API quickSpinInitEx](#) ([spinCamera](#) hCamera, [quickSpin](#) \*pQuickSpin, [quickSpinTLDevice](#) \*pQuickSpinTLDevice, [quickSpinTLStream](#) \*pQuickSpinTLStream)
- [SPINNAKERC\\_API quickSpinTLDeviceInit](#) ([spinCamera](#) hCamera, [quickSpinTLDevice](#) \*pQuickSpinTLDevice)
- [SPINNAKERC\\_API quickSpinTLStreamInit](#) ([spinCamera](#) hCamera, [quickSpinTLStream](#) \*pQuickSpinTLStream)
- [SPINNAKERC\\_API quickSpinTLInterfaceInit](#) ([spinInterface](#) hInterface, [quickSpinTLInterface](#) \*pQuickSpinTLInterface)
- [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:



## 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

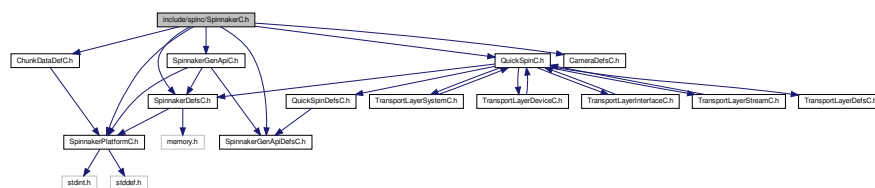
```
typedef spinNodeHandle quickSpinRegisterNode
```

## 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, [spinDevice](#)↔ArrivalEventHandler 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, [spinDevice](#)↔RemovalEventHandler 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, [spinDeviceRemovalEventHandler](#) 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](#) iGroupKey, [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](#) \*hLibraryVersion)

*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](#) \*phInterface)

*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](#) \*phCamera)

*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, [spin↔Camera](#) \*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 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, [spin↔CameraList](#) 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, [spinDevice↔ArrivalEventHandler](#) hDeviceArrivalEventHandler)

*Registers a device arrival event handler on an interface (event handlers registered in this way must be unregistered)*

  - [SPINNAKERC\\_API spinInterfaceRegisterDeviceRemovalEventHandler](#) ([spinInterface](#) hInterface, [spin↔DeviceRemovalEventHandler](#) hDeviceRemovalEventHandler)

*Registers a device removal event handler on an interface (event handlers registered in this way must be unregistered)*

  - [SPINNAKERC\\_API spinInterfaceUnregisterDeviceArrivalEventHandler](#) ([spinInterface](#) hInterface, [spin↔DeviceArrivalEventHandler](#) hDeviceArrivalEventHandler)

*Unregisters a device arrival event handler from an interface.*

  - [SPINNAKERC\\_API spinInterfaceUnregisterDeviceRemovalEventHandler](#) ([spinInterface](#) hInterface, [spin↔DeviceRemovalEventHandler](#) hDeviceRemovalEventHandler)

*Unregisters a device removal event handler from an interface.*

  - [SPINNAKERC\\_API spinInterfaceRegisterInterfaceEventHandler](#) ([spinInterface](#) hInterface, [spinInterface↔EventHandler](#) hInterfaceEventHandler)

*Registers an interface event handler (both device arrival and device removal) on an interface.*

  - [SPINNAKERC\\_API spinInterfaceUnregisterInterfaceEventHandler](#) ([spinInterface](#) hInterface, [spinInterface↔EventHandler](#) 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](#) \*phNodeMap)
- 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 spinCameraGetUniqueID](#) ([spinCamera](#) hCamera, char \*pBuf, size\_t \*pBufLen)
- Retrieves a unique identifier for a camera.*
- [SPINNAKERC\\_API spinCameralStreaming](#) ([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 spinCameraUnregisterImageEventHandler](#) ([spinCamera](#) hCamera, [spinImageEventHandler](#) hImageEventHandler)
- Unregisters an image event handler from a camera.*
- [SPINNAKERC\\_API spinCameraRelease](#) ([spinCamera](#) hCamera)
- Releases a camera.*
- [SPINNAKERC\\_API spinCameralValid](#) ([spinCamera](#) hCamera, bool8\_t \*pbValid)
- Checks whether a camera is still valid for use.*
- [SPINNAKERC\\_API spinCameralInitialized](#) ([spinCamera](#) hCamera, bool8\_t \*pbInit)
- Checks whether a camera is currently initialized.*
- [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.*
- [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, [spinPayloadTypeInfoIds](#) 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 spinImageSetDefaultColorProcessing](#) ([spinColorProcessingAlgorithm](#) algorithm)

*Sets the default color processing algorithm of all images (if not otherwise set)*

  - [SPINNAKERC\\_API spinImageGetDefaultColorProcessing](#) ([spinColorProcessingAlgorithm](#) \*pAlgorithm)

*Retrieves the default color processing algorithm.*

  - [SPINNAKERC\\_API spinImageGetColorProcessing](#) ([spinImage](#) hImage, [spinColorProcessingAlgorithm](#) \*pAlgorithm)

*Retrieves the color processing algorithm of a specific image.*

  - [SPINNAKERC\\_API spinImageSetNumDecompressionThreads](#) (unsigned int numThreads)

*Sets the default number of threads used for image decompression during [spinImageConvert\(\)](#).*

  - [SPINNAKERC\\_API spinImageGetNumDecompressionThreads](#) (unsigned int \*pNumThreads)

*Gets the number of threads used for image decompression during [Convert\(\)](#).*

  - [SPINNAKERC\\_API spinImageConvert](#) ([spinImage](#) hSrcImage, [spinPixelFormatEnums](#) pixelFormat, [spinImage](#) hDestImage)

*Converts the pixel format of one image into a new image.*

  - [SPINNAKERC\\_API spinImageConvertEx](#) ([spinImage](#) hSrcImage, [spinPixelFormatEnums](#) pixelFormat, [spinColorProcessingAlgorithm](#) algorithm, [spinImage](#) hDestImage)

*Converts the pixel format and color processing algorithm of one image into a new 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, [spinPayloadTypeInfolDs](#))*

  - [SPINNAKERC\\_API spinImageGetTLPayloadType](#) ([spinImage](#) hImage, [spinPayloadTypeInfolDs](#) \*pPayloadType)

*Retrieves the transport layer payload type of an image (as an enum, [spinPayloadTypeInfolDs](#))*

  - [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, [spinPixelFormatNamespaceID](#) \*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](#) \*pIsIncomplete)

*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 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 spinDeviceArrivalEventHandlerCreate](#) ([spinDeviceArrivalEventHandler](#) \*phDeviceArrivalEventHandler, [spinArrivalEventFunction](#) pFunction, [void](#) \*pUserData)

*Creates a device arrival event handler.*
- [SPINNAKERC\\_API spinDeviceArrivalEventHandlerDestroy](#) ([spinDeviceArrivalEventHandler](#) hDeviceArrivalEventHandler)

*Destroys a device arrival event handler.*
- [SPINNAKERC\\_API spinDeviceRemovalEventHandlerCreate](#) ([spinDeviceRemovalEventHandler](#) \*phDeviceRemovalEventHandler, [spinRemovalEventFunction](#) pFunction, [void](#) \*pUserData)

*Creates a device removal event handler.*
- [SPINNAKERC\\_API spinDeviceRemovalEventHandlerDestroy](#) ([spinDeviceRemovalEventHandler](#) hDeviceRemovalEventHandler)

*Destroys a device removal event handler.*
- [SPINNAKERC\\_API spinInterfaceEventHandlerCreate](#) ([spinInterfaceEventHandler](#) \*phInterfaceEventHandler, [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 \*pPixelValueMax, 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](#) \*pBufLen)  
*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 \*pValue)  
*Retrieves the integer value of a device event.*
- [SPINNAKERC\\_API spinImageChunkDataGetFloatValue](#) ([spinImage](#) hImage, const char \*pName, double \*pValue)  
*Retrieves the float value of a device event.*

## 13.12.1 Function Documentation

### 13.12.1.1 spinCameraForceIP()

[SPINNAKERC\\_API](#) spinCameraForceIP ( )

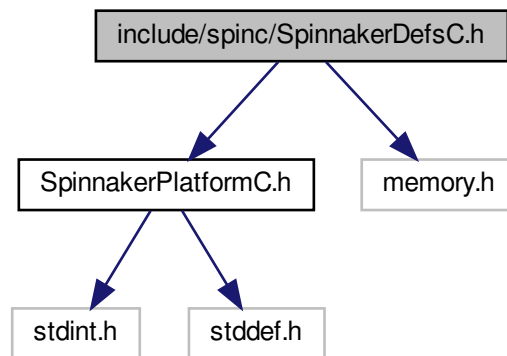
Forces the camera to be on the same subnet as its corresponding interface.

#### Returns

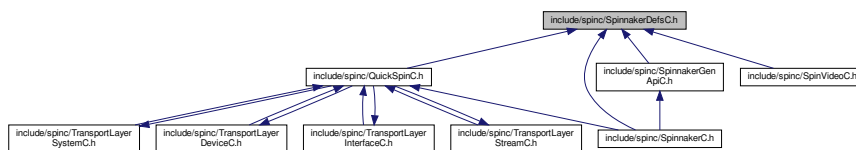
spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

## 13.13 include/spinc/SpinnakerDefsC.h File Reference

Include dependency graph for SpinnakerDefsC.h:



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



### 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 \* [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 \* [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 \*pEventName, void \*pUserData)  
*Function signatures are used to create and trigger callbacks and events.*
- typedef void(\* [spinImageEventFunction](#)) (const [spinImage](#) hImage, void \*pUserData)
- typedef void(\* [spinArrivalEventFunction](#)) (uint64\_t deviceSerialNumber, void \*pUserData)
- typedef void(\* [spinRemovalEventFunction](#)) (uint64\_t deviceSerialNumber, 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,
  - `GENICAM_ERR_INVALID_ARGUMENT` = -2001,
  - `GENICAM_ERR_OUT_OF_RANGE` = -2002,
  - `GENICAM_ERR_PROPERTY` = -2003,
  - `GENICAM_ERR_RUN_TIME` = -2004,
  - `GENICAM_ERR_LOGICAL` = -2005,
  - `GENICAM_ERR_ACCESS` = -2006,
  - `GENICAM_ERR_TIMEOUT` = -2007,
  - `GENICAM_ERR_DYNAMIC_CAST` = -2008,
  - `GENICAM_ERR_GENERIC` = -2009,
  - `GENICAM_ERR_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` {
  - `DEFAULT`,
  - `NO_COLOR_PROCESSING`,
  - `NEAREST_NEIGHBOR`,
  - `NEAREST_NEIGHBOR_AVG`,
  - `BILINEAR`,
  - `EDGE_SENSING`,
  - `HQ_LINEAR`,
  - `IPP`,
  - `DIRECTIONAL_FILTER`,
  - `RIGOROUS`,
  - `WEIGHTED_DIRECTIONAL_FILTER` }

*Color processing algorithms.*

- enum `spinStatisticsChannel` {  
`GREY`,  
`RED`,  
`GREEN`,  
`BLUE`,  
`HUE`,  
`SATURATION`,  
`LIGHTNESS`,  
`NUM_STATISTICS_CHANNELS` }

*Channels that allow statistics to be calculated.*

- enum `spinImageFileFormat` {  
`FROM_FILE_EXT` = -1,  
`PGM`,  
`PPM`,  
`BMP`,  
`JPEG`,  
`JPEG2000`,  
`TIFF`,  
`PNG`,  
`RAW`,  
`IMAGE_FILE_FORMAT_FORCE_32BITS` = 0x7FFFFFFF }

*File formats to be used for saving images to disk.*

- enum `spinPixelFormatNamespaceID` {  
`SPINNAKER_PIXELFORMAT_NAMESPACE_UNKNOWN` = 0,  
`SPINNAKER_PIXELFORMAT_NAMESPACE_GEV` = 1,  
`SPINNAKER_PIXELFORMAT_NAMESPACE_IIDC` = 2,  
`SPINNAKER_PIXELFORMAT_NAMESPACE_PFNC_16BIT` = 3,  
`SPINNAKER_PIXELFORMAT_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` {  
`IMAGE_UNKNOWN_ERROR` = -1,  
`IMAGE_NO_ERROR` = 0,  
`IMAGE_CRC_CHECK_FAILED` = 1,  
`IMAGE_DATA_OVERFLOW` = 2,  
`IMAGE_MISSING_PACKETS`,  
`IMAGE_LEADER_BUFFER_SIZE_INCONSISTENT`,  
`IMAGE_TRAILER_BUFFER_SIZE_INCONSISTENT`,  
`IMAGE_PACKETID_INCONSISTENT`,  
`IMAGE_MISSING_LEADER` = 7,  
`IMAGE_MISSING_TRAILER` = 8,  
`IMAGE_DATA_INCOMPLETE` = 9,  
`IMAGE_INFO_INCONSISTENT` = 10,  
`IMAGE_CHUNK_DATA_INVALID` = 11,  
`IMAGE_NO_SYSTEM_RESOURCES` = 12 }

*Status of images returned from `spinImageGetStatus()` call.*

- enum `spinnakerLogLevel` {  
`LOG_LEVEL_OFF` = -1,  
`LOG_LEVEL_FATAL` = 0,  
`LOG_LEVEL_ALERT` = 100,  
`LOG_LEVEL_CRIT` = 200,  
`LOG_LEVEL_ERROR` = 300,  
`LOG_LEVEL_WARN` = 400,  
`LOG_LEVEL_NOTICE` = 500,  
`LOG_LEVEL_INFO` = 600,  
`LOG_LEVEL_DEBUG` = 700,  
`LOG_LEVEL_NOTSET` = 800 }

*log levels*

- enum [spinPayloadTypeInfoIds](#) {  
[PAYLOAD\\_TYPE\\_UNKNOWN](#) = 0,  
[PAYLOAD\\_TYPE\\_IMAGE](#) = 1,  
[PAYLOAD\\_TYPE\\_RAW\\_DATA](#) = 2,  
[PAYLOAD\\_TYPE\\_FILE](#) = 3,  
[PAYLOAD\\_TYPE\\_CHUNK\\_DATA](#) = 4,  
[PAYLOAD\\_TYPE\\_JPEG](#) = 5,  
[PAYLOAD\\_TYPE\\_JPEG2000](#) = 6,  
[PAYLOAD\\_TYPE\\_H264](#) = 7,  
[PAYLOAD\\_TYPE\\_CHUNK\\_ONLY](#) = 8,  
[PAYLOAD\\_TYPE\\_DEVICE\\_SPECIFIC](#) = 9,  
[PAYLOAD\\_TYPE\\_MULTI\\_PART](#) = 10,  
[PAYLOAD\\_TYPE\\_CUSTOM\\_ID](#) = 1000,  
[PAYLOAD\\_TYPE\\_EXTENDED\\_CHUNK](#) = 1001,  
[PAYLOAD\\_TYPE\\_LOSSLESS\\_COMPRESSED](#) = 1002,  
[PAYLOAD\\_TYPE\\_LOSSY\\_COMPRESSED](#) = 1003,  
[PAYLOAD\\_TYPE\\_JPEG\\_LOSSLESS\\_COMPRESSED](#) = 1004,  
[PAYLOAD\\_TYPE\\_CHUNK\\_DATA\\_LOSSLESS\\_COMPRESSED](#) = 1005,  
[PAYLOAD\\_TYPE\\_CHUNK\\_DATA\\_LOSSY\\_COMPRESSED](#) = 1006 }
- enum [spinCompressionMethod](#) {  
[NONE](#) = 1,  
[PACKBITS](#),  
[DEFLATE](#),  
[ADOBE\\_DEFLATE](#),  
[CCITTFAX3](#),  
[CCITTFAX4](#),  
[LZW](#),  
[JPG](#) }

*Compression method used in saving TIFF images in the [spinTIFFOption](#) struct.*

- enum [actionCommandStatus](#) {  
[ACTION\\_COMMAND\\_STATUS\\_OK](#) = 0,  
[ACTION\\_COMMAND\\_STATUS\\_NO\\_REF\\_TIME](#) = 0x8013,  
[ACTION\\_COMMAND\\_STATUS\\_OVERFLOW](#) = 0x8015,  
[ACTION\\_COMMAND\\_STATUS\\_ACTION\\_LATE](#) = 0x8016,  
[ACTION\\_COMMAND\\_STATUS\\_ERROR](#) = 0x8FFF }

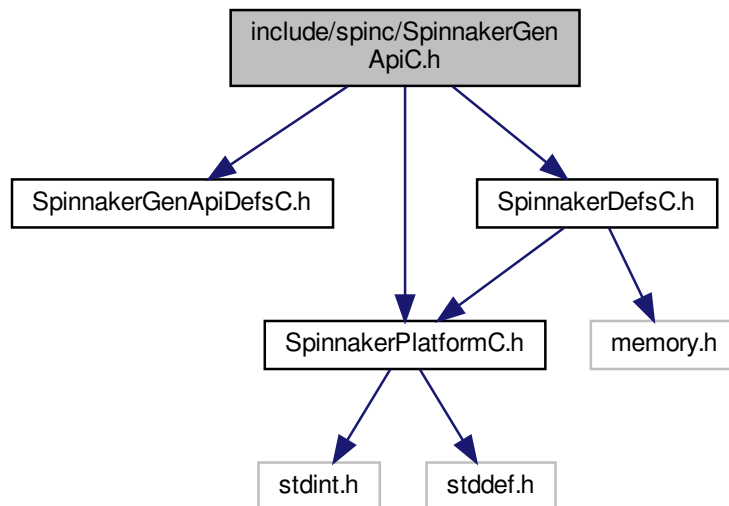
*Possible Status Codes Returned from Action Command.*

## Variables

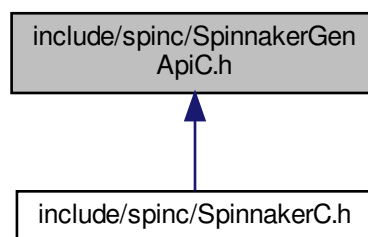
- static const [bool8\\_t False](#) = 0
- static const [bool8\\_t True](#) = 1

## 13.14 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 spinNodesImplemented](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbResult)

*Checks whether a node is implemented.*
- [SPINNAKERC\\_API spinNodesReadable](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbResult)

*Checks whether a node is readable.*
- [SPINNAKERC\\_API spinNodesWritable](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbResult)

*Checks whether a node is writable.*
- [SPINNAKERC\\_API spinNodesAvailable](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbResult)

*Checks whether a node is available.*
- [SPINNAKERC\\_API spinNodesEqual](#) ([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](#) \*pCachingMode)

*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](#) pCbFunction, [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](#) \*pBufLen)
 

*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](#) \*pBufLen)
 

*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 enumeration 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](#) \*pBufLen)

*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) of 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](#) hFeature)

*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)

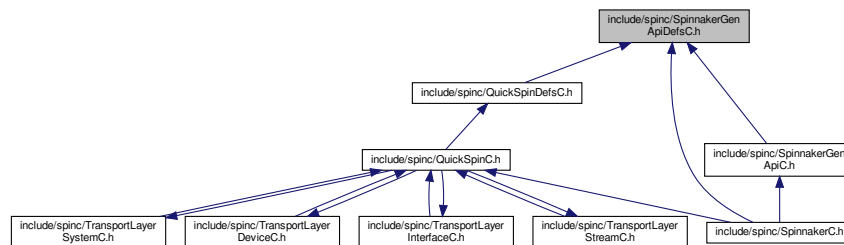
*Sets the value of a register node; manually set whether to verify the node.*

- [SPINNAKERC\\_API spinRegisterSetReference](#) ([spinNodeHandle](#) hNode, [spinNodeHandle](#) hRef)

*Uses a second node as a reference for a register node.*

## 13.15 include/spinc/SpinnakerGenApiDefsC.h File Reference

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



### 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

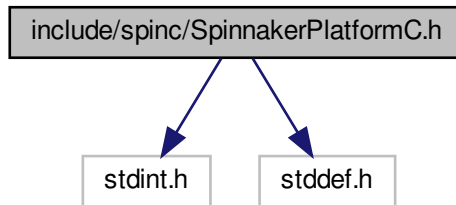
- enum [spinNodeType](#) {  
[ValueNode](#),  
[BaseNode](#),  
[IntegerNode](#),  
[BooleanNode](#),  
[FloatNode](#),  
[CommandNode](#),  
[StringNode](#),  
[RegisterNode](#),  
[EnumerationNode](#),  
[EnumEntryNode](#),  
[CategoryNode](#),  
[PortNode](#),  
[UnknownNode](#) = -1 }

- 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 `spinEndianness` {  
`BigEndian`,  
`LittleEndian`,  
`_UndefinedEndian` }  
*Endianness 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.*

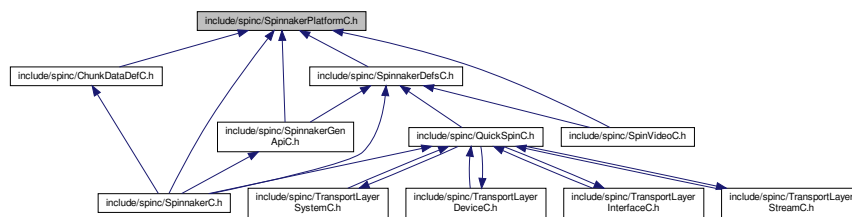
- enum `spinSlope` {  
`Increasing`,  
`Decreasing`,  
`Varying`,  
`Automatic`,  
`_UndefinedESlope` }  
*typedef for fomula type*
- enum `spinXMLValidation` {  
`xvLoad` = 0x00000001L,  
`xvCycles` = 0x00000002L,  
`xvSFNC` = 0x00000004L,  
`xvDefault` = 0x00000000L,  
`xvAll` = 0xffffffffL,  
`_UndefinedEXMLValidation` = 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` {  
`intflValue`,  
`intflBase`,  
`intflInteger`,  
`intflBoolean`,  
`intflCommand`,  
`intflFloat`,  
`intflString`,  
`intflRegister`,  
`intflCategory`,  
`intflEnumeration`,  
`intflEnumEntry`,  
`intflIPort` }  
*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*

## 13.16 include/spinc/SpinnakerPlatformC.h File Reference

Include dependency graph for SpinnakerPlatformC.h:



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



### Macros

- `#define SPINNAKERC_API SPINC_IMPORT_EXPORT spinError SPINC_CALLTYPE`

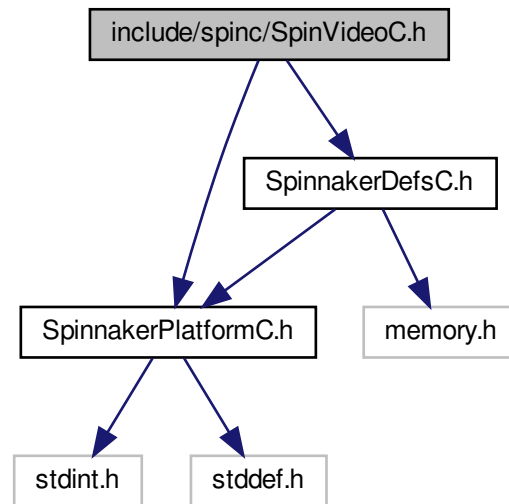
### 13.16.1 Macro Definition Documentation

#### 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:

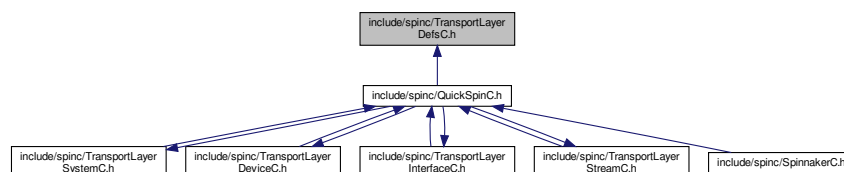


#### Functions

- [SPINNAKERC\\_API spinVideoOpenUncompressed](#) ([spinVideo](#) \*phSpinVideo, const char \*pName, [spinAVIOOption](#) option)
- [SPINNAKERC\\_API spinVideoOpenMJPEG](#) ([spinVideo](#) \*phSpinVideo, const char \*pName, [spinMJPEGOption](#) 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)

### 13.18 include/spinc/TransportLayerDefsC.h File Reference

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



## Enumerations

- enum [spinTLStreamTypeEnums](#) {  
[StreamType\\_GigEVision](#),  
[StreamType\\_CameraLink](#),  
[StreamType\\_CameraLinkHS](#),  
[StreamType\\_CoaXPRESS](#),  
[StreamType\\_USB3Vision](#),  
[StreamType\\_Custom](#),  
[NUMSTREAMTYPE](#) }

*The enumeration definitions for transport layer nodes.*

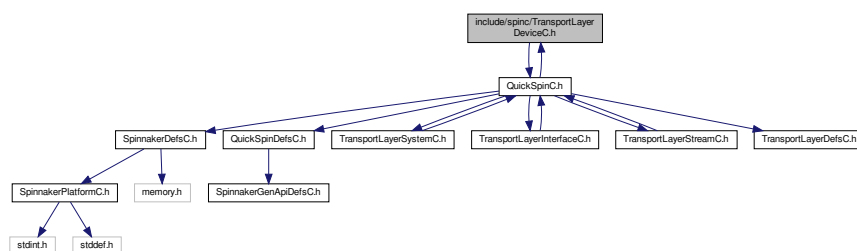
- enum [spinTLStreamBufferCountModeEnums](#) {  
[StreamBufferCountMode\\_Manual](#),  
[StreamBufferCountMode\\_Auto](#),  
[NUMSTREAMBUFFERCOUNTMODE](#) }
- enum [spinTLStreamBufferHandlingModeEnums](#) {  
[StreamBufferHandlingMode\\_OldestFirst](#),  
[StreamBufferHandlingMode\\_OldestFirstOverwrite](#),  
[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 [spinTLGevCCPEnums](#) {  
[GevCCP\\_EnumEntry\\_GevCCP\\_OpenAccess](#),  
[GevCCP\\_EnumEntry\\_GevCCP\\_ExclusiveAccess](#),  
[GevCCP\\_EnumEntry\\_GevCCP\\_ControlAccess](#),  
[NUMGEVCCP](#) }
- enum [spinTLGUIXMLLocationEnums](#) {  
[GUIXMLLocation\\_Device](#),  
[GUIXMLLocation\\_Host](#),  
[NUMGUIXMLLOCATION](#) }
- enum [spinTLGenICamXMLLocationEnums](#) {  
[GenICamXMLLocation\\_Device](#),  
[GenICamXMLLocation\\_Host](#),  
[NUMGENICAMXMLLOCATION](#) }
- enum [spinTLDeviceEndiannessMechanismEnums](#) {  
[DeviceEndiannessMechanism\\_Legacy](#),  
[DeviceEndiannessMechanism\\_Standard](#),  
[NUMDEVICEENDIANESSMECHANISM](#) }
- enum [spinTLDeviceCurrentSpeedEnums](#) {  
[DeviceCurrentSpeed\\_UnknownSpeed](#),  
[DeviceCurrentSpeed\\_LowSpeed](#),

```
DeviceCurrentSpeed_FullSpeed,
DeviceCurrentSpeed_HighSpeed,
DeviceCurrentSpeed_SuperSpeed,
NUMDEVICECURRENTSPEED }
```

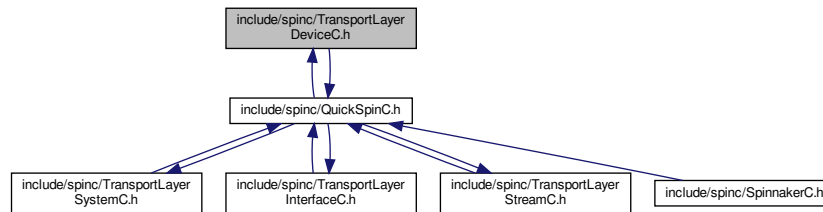
- enum `spinTLInterfaceTypeEnums` {  
`InterfaceType_GigEVision`,  
`InterfaceType_CameraLink`,  
`InterfaceType_CameraLinkHS`,  
`InterfaceType_CoaXPress`,  
`InterfaceType_USB3Vision`,  
`InterfaceType_Custom`,  
`NUMINTERFACETYPE` }
- enum `spinTLPOEStatusEnums` {  
`POEStatus_NotSupported`,  
`POEStatus_PowerOff`,  
`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` }

## 13.19 include/spinc/TransportLayerDeviceC.h File Reference

Include dependency graph for `TransportLayerDeviceC.h`:



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

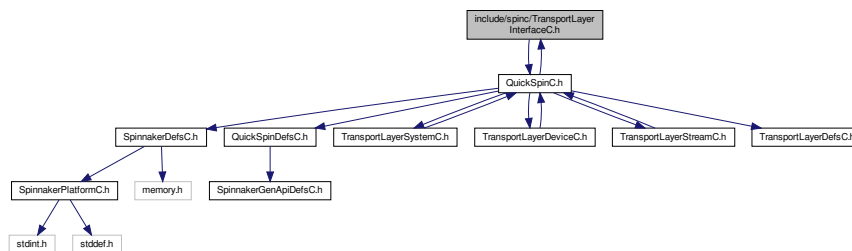


## Data Structures

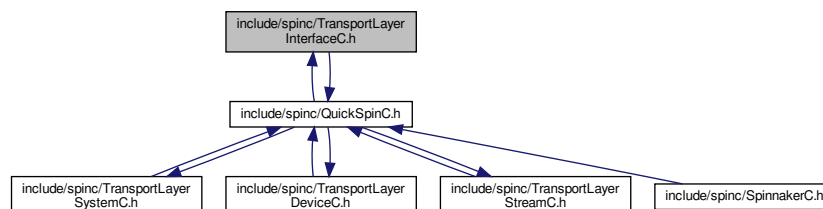
- struct [quickSpinTLDevice](#)

## 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:

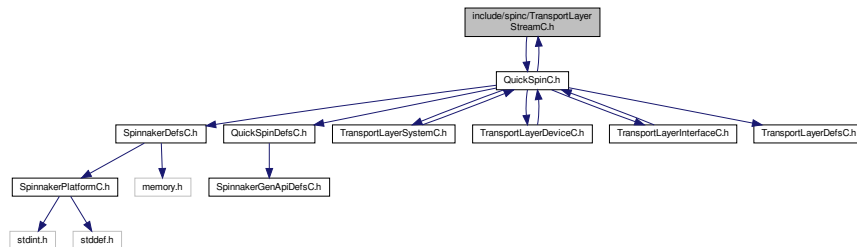


## Data Structures

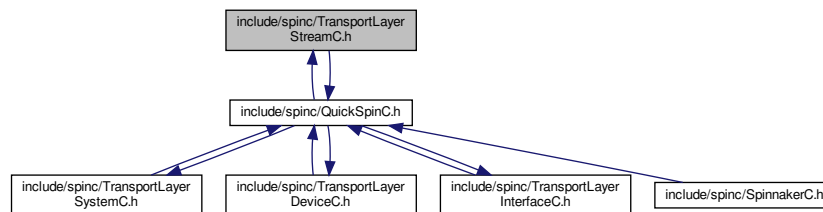
- struct [quickSpinTLInterface](#)

## 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:

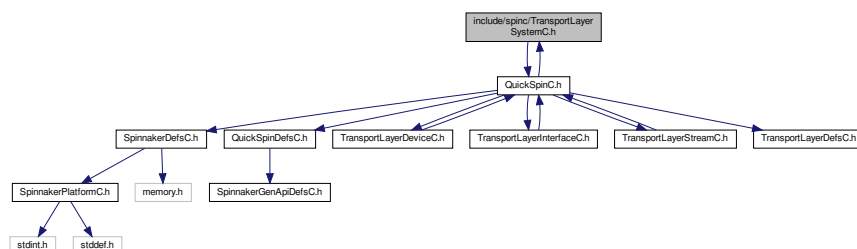


### Data Structures

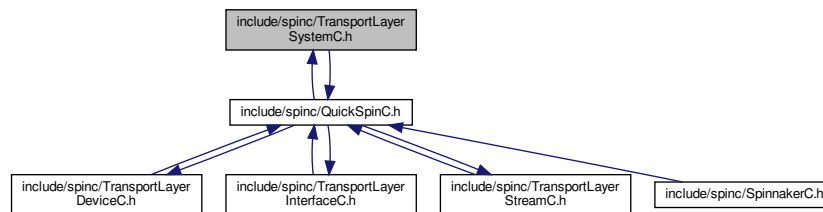
- struct [quickSpinTLStream](#)

## 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:



## Data Structures

- struct [quickSpinTLSystem](#)



# Index

- aPAUSEMACCtrlFramesReceived
  - quickSpin, [365](#)
- aPAUSEMACCtrlFramesTransmitted
  - quickSpin, [365](#)
- AasRoiEnable
  - quickSpin, [362](#)
- AasRoiHeight
  - quickSpin, [362](#)
- AasRoiOffsetX
  - quickSpin, [362](#)
- AasRoiOffsetY
  - quickSpin, [362](#)
- AasRoiWidth
  - quickSpin, [362](#)
- AcquisitionAbort
  - quickSpin, [363](#)
- AcquisitionArm
  - quickSpin, [363](#)
- AcquisitionBurstFrameCount
  - quickSpin, [363](#)
- AcquisitionFrameCount
  - quickSpin, [363](#)
- AcquisitionFrameRate
  - quickSpin, [363](#)
- AcquisitionFrameRateEnable
  - quickSpin, [363](#)
- AcquisitionLineRate
  - quickSpin, [363](#)
- AcquisitionMode
  - quickSpin, [363](#)
- AcquisitionResultingFrameRate
  - quickSpin, [364](#)
- AcquisitionStart
  - quickSpin, [364](#)
- AcquisitionStatus
  - quickSpin, [364](#)
- AcquisitionStatusSelector
  - quickSpin, [364](#)
- AcquisitionStop
  - quickSpin, [364](#)
- ActionCommand
  - quickSpinTLInterface, [454](#)
- actionCommandResult, [349](#)
  - DeviceAddress, [349](#)
  - Status, [349](#)
- actionCommandStatus
  - Spinnaker C Structures, [264](#)
- ActionDeviceKey
  - quickSpin, [364](#)
- ActionGroupKey
  - quickSpin, [364](#)
- ActionGroupMask
  - quickSpin, [364](#)
- ActionQueueSize
  - quickSpin, [365](#)
- ActionSelector
  - quickSpin, [365](#)
- ActionUnconditionalMode
  - quickSpin, [365](#)
- AdaptiveCompressionEnable
  - quickSpin, [365](#)
- AdcBitDepth
  - quickSpin, [365](#)
- AutoAlgorithmSelector
  - quickSpin, [365](#)
- AutoExposureControlLoopDamping
  - quickSpin, [366](#)
- AutoExposureControlPriority
  - quickSpin, [366](#)
- AutoExposureEVCompensation
  - quickSpin, [366](#)
- AutoExposureExposureTimeLowerLimit
  - quickSpin, [366](#)
- AutoExposureExposureTimeUpperLimit
  - quickSpin, [366](#)
- AutoExposureGainLowerLimit
  - quickSpin, [366](#)
- AutoExposureGainUpperLimit
  - quickSpin, [366](#)
- AutoExposureGreyValueLowerLimit
  - quickSpin, [366](#)
- AutoExposureGreyValueUpperLimit
  - quickSpin, [367](#)
- AutoExposureLightingMode
  - quickSpin, [367](#)
- AutoExposureMeteringMode
  - quickSpin, [367](#)
- AutoExposureTargetGreyValue
  - quickSpin, [367](#)
- AutoExposureTargetGreyValueAuto
  - quickSpin, [367](#)
- BalanceRatio
  - quickSpin, [367](#)
- BalanceRatioSelector
  - quickSpin, [367](#)
- BalanceWhiteAuto
  - quickSpin, [367](#)
- BalanceWhiteAutoDamping

- quickSpin, 368
- BalanceWhiteAutoLowerLimit
  - quickSpin, 368
- BalanceWhiteAutoProfile
  - quickSpin, 368
- BalanceWhiteAutoUpperLimit
  - quickSpin, 368
- binaryFile
  - spinPGMOption, 482
  - spinPPMOption, 484
- BinningHorizontal
  - quickSpin, 368
- BinningHorizontalMode
  - quickSpin, 368
- BinningSelector
  - quickSpin, 368
- BinningVertical
  - quickSpin, 368
- BinningVerticalMode
  - quickSpin, 369
- bitrate
  - spinH264Option, 476
- BlackLevel
  - quickSpin, 369
- BlackLevelAuto
  - quickSpin, 369
- BlackLevelAutoBalance
  - quickSpin, 369
- BlackLevelClampingEnable
  - quickSpin, 369
- BlackLevelRaw
  - quickSpin, 369
- BlackLevelSelector
  - quickSpin, 369
- bool8\_t
  - Spinnaker C Definitions, 22
- build
  - spinLibraryVersion, 480
- Camera Access, 182
  - spinCameraBeginAcquisition, 183
  - spinCameraDelInit, 184
  - spinCameraEndAcquisition, 184
  - spinCameraGetAccessMode, 184
  - spinCameraGetGuiXml, 185
  - spinCameraGetNextImage, 185
  - spinCameraGetNextImageEx, 186
  - spinCameraGetNodeMap, 186
  - spinCameraGetTLDeviceNodeMap, 187
  - spinCameraGetTLStreamNodeMap, 187
  - spinCameraGetUniqueID, 188
  - spinCameraInit, 188
  - spinCamerasInitialized, 189
  - spinCamerasStreaming, 189
  - spinCamerasValid, 190
  - spinCameraReadPort, 190
  - spinCameraRegisterDeviceEventHandler, 190
  - spinCameraRegisterDeviceEventHandlerEx, 191
  - spinCameraRegisterImageEventHandler, 191
  - spinCameraRelease, 192
  - spinCameraUnregisterDeviceEventHandler, 192
  - spinCameraUnregisterImageEventHandler, 193
  - spinCameraWritePort, 193
- Camera Enumerations, 23
  - spinAcquisitionModeEnums, 55
  - spinAcquisitionStatusSelectorEnums, 55
  - spinActionUnconditionalModeEnums, 56
  - spinAdcBitDepthEnums, 56
  - spinAutoAlgorithmSelectorEnums, 56
  - spinAutoExposureControlPriorityEnums, 57
  - spinAutoExposureLightingModeEnums, 57
  - spinAutoExposureMeteringModeEnums, 58
  - spinAutoExposureTargetGreyValueAutoEnums, 58
  - spinBalanceRatioSelectorEnums, 58
  - spinBalanceWhiteAutoEnums, 59
  - spinBalanceWhiteAutoProfileEnums, 59
  - spinBinningHorizontalModeEnums, 59
  - spinBinningSelectorEnums, 60
  - spinBinningVerticalModeEnums, 60
  - spinBlackLevelAutoBalanceEnums, 60
  - spinBlackLevelAutoEnums, 61
  - spinBlackLevelSelectorEnums, 61
  - spinChunkBlackLevelSelectorEnums, 61
  - spinChunkCounterSelectorEnums, 62
  - spinChunkEncoderSelectorEnums, 62
  - spinChunkEncoderStatusEnums, 62
  - spinChunkExposureTimeSelectorEnums, 63
  - spinChunkGainSelectorEnums, 63
  - spinChunkImageComponentEnums, 63
  - spinChunkPixelFormatEnums, 64
  - spinChunkRegionIDEnums, 64
  - spinChunkScan3dCoordinateReferenceSelector↔  
Enums, 65
  - spinChunkScan3dCoordinateSelectorEnums, 65
  - spinChunkScan3dCoordinateSystemEnums, 65
  - spinChunkScan3dCoordinateSystemReference↔  
Enums, 66
  - spinChunkScan3dCoordinateTransformSelector↔  
Enums, 66
  - spinChunkScan3dDistanceUnitEnums, 66
  - spinChunkScan3dOutputModeEnums, 67
  - spinChunkSelectorEnums, 67
  - spinChunkSourceIDEnums, 68
  - spinChunkTimerSelectorEnums, 68
  - spinChunkTransferStreamIDEnums, 69
  - spinCIConfigurationEnums, 69
  - spinCITimeSlotsCountEnums, 69
  - spinColorTransformationSelectorEnums, 70
  - spinColorTransformationValueSelectorEnums, 70
  - spinCompressionSaturationPriorityEnums, 71
  - spinCounterEventActivationEnums, 71
  - spinCounterEventSourceEnums, 71
  - spinCounterResetActivationEnums, 72
  - spinCounterResetSourceEnums, 72
  - spinCounterSelectorEnums, 73
  - spinCounterStatusEnums, 73
  - spinCounterTriggerActivationEnums, 74

- spinCounterTriggerSourceEnums, 74
- spinCxpConnectionTestModeEnums, 75
- spinCxpLinkConfigurationEnums, 75
- spinCxpLinkConfigurationPreferredEnums, 76
- spinCxpLinkConfigurationStatusEnums, 77
- spinCxpPoCxpStatusEnums, 78
- spinDecimationHorizontalModeEnums, 78
- spinDecimationSelectorEnums, 78
- spinDecimationVerticalModeEnums, 79
- spinDefectCorrectionModeEnums, 79
- spinDeinterlacingEnums, 79
- spinDeviceCharacterSetEnums, 80
- spinDeviceClockSelectorEnums, 80
- spinDeviceConnectionStatusEnums, 80
- spinDeviceIndicatorModeEnums, 81
- spinDeviceLinkHeartbeatModeEnums, 81
- spinDeviceLinkThroughputLimitModeEnums, 81
- spinDevicePowerSupplySelectorEnums, 81
- spinDeviceRegistersEndiannessEnums, 82
- spinDeviceScanTypeEnums, 82
- spinDeviceSerialPortBaudRateEnums, 82
- spinDeviceSerialPortSelectorEnums, 83
- spinDeviceStreamChannelEndiannessEnums, 83
- spinDeviceStreamChannelTypeEnums, 83
- spinDeviceTLTypeEnums, 85
- spinDeviceTapGeometryEnums, 84
- spinDeviceTemperatureSelectorEnums, 85
- spinDeviceTypeEnums, 86
- spinEncoderModeEnums, 86
- spinEncoderOutputModeEnums, 86
- spinEncoderResetActivationEnums, 87
- spinEncoderResetSourceEnums, 87
- spinEncoderSelectorEnums, 88
- spinEncoderSourceAEnums, 89
- spinEncoderSourceBEnums, 89
- spinEncoderStatusEnums, 89
- spinEventNotificationEnums, 90
- spinEventSelectorEnums, 90
- spinExposureActiveModeEnums, 90
- spinExposureAutoEnums, 90
- spinExposureModeEnums, 91
- spinExposureTimeModeEnums, 91
- spinExposureTimeSelectorEnums, 92
- spinFileOpenModeEnums, 92
- spinFileOperationSelectorEnums, 92
- spinFileOperationStatusEnums, 93
- spinFileSelectorEnums, 93
- spinGainAutoBalanceEnums, 93
- spinGainAutoEnums, 95
- spinGainSelectorEnums, 95
- spinGevCCPEnums, 95
- spinGevCurrentPhysicalLinkConfigurationEnums, 96
- spinGevGVCPExtendedStatusCodesSelector↔  
Enums, 96
- spinGevGVSPExtentedIDModeEnums, 96
- spinGevIEEE1588ClockAccuracyEnums, 97
- spinGevIEEE1588ModeEnums, 97
- spinGevIEEE1588StatusEnums, 97
- spinGevIPConfigurationStatusEnums, 98
- spinGevPhysicalLinkConfigurationEnums, 98
- spinGevSupportedOptionSelectorEnums, 98
- spinImageComponentSelectorEnums, 99
- spinImageCompressionJPEGFormatOption↔  
Enums, 100
- spinImageCompressionModeEnums, 100
- spinImageCompressionRateOptionEnums, 101
- spinLUTSelectorEnums, 105
- spinLineFormatEnums, 101
- spinLineInputFilterSelectorEnums, 101
- spinLineModeEnums, 102
- spinLineSelectorEnums, 102
- spinLineSourceEnums, 102
- spinLogicBlockLUTInputActivationEnums, 103
- spinLogicBlockLUTInputSelectorEnums, 103
- spinLogicBlockLUTInputSourceEnums, 104
- spinLogicBlockLUTSelectorEnums, 104
- spinLogicBlockSelectorEnums, 105
- spinPixelColorFilterEnums, 105
- spinPixelFormatEnums, 106
- spinPixelFormatInfoSelectorEnums, 111
- spinPixelSizeEnums, 117
- spinRegionDestinationEnums, 118
- spinRegionModeEnums, 118
- spinRegionSelectorEnums, 118
- spinRgbTransformLightSourceEnums, 118
- spinScan3dCoordinateReferenceSelectorEnums, 119
- spinScan3dCoordinateSelectorEnums, 119
- spinScan3dCoordinateSystemEnums, 120
- spinScan3dCoordinateSystemReferenceEnums, 120
- spinScan3dCoordinateTransformSelectorEnums, 120
- spinScan3dDistanceUnitEnums, 121
- spinScan3dOutputModeEnums, 121
- spinSensorDigitizationTapsEnums, 122
- spinSensorShutterModeEnums, 122
- spinSensorTapsEnums, 123
- spinSequencerConfigurationModeEnums, 123
- spinSequencerConfigurationValidEnums, 124
- spinSequencerModeEnums, 124
- spinSequencerSetValidEnums, 124
- spinSequencerTriggerActivationEnums, 124
- spinSequencerTriggerSourceEnums, 125
- spinSerialPortBaudRateEnums, 125
- spinSerialPortParityEnums, 126
- spinSerialPortSelectorEnums, 126
- spinSerialPortSourceEnums, 126
- spinSerialPortStopBitsEnums, 127
- spinSoftwareSignalSelectorEnums, 127
- spinSourceSelectorEnums, 127
- spinTestPatternEnums, 127
- spinTestPatternGeneratorSelectorEnums, 128
- spinTimerSelectorEnums, 128
- spinTimerStatusEnums, 128

- spinTimerTriggerActivationEnums, [129](#)
- spinTimerTriggerSourceEnums, [129](#)
- spinTransferComponentSelectorEnums, [131](#)
- spinTransferControlModeEnums, [131](#)
- spinTransferOperationModeEnums, [131](#)
- spinTransferQueueModeEnums, [132](#)
- spinTransferSelectorEnums, [132](#)
- spinTransferStatusSelectorEnums, [132](#)
- spinTransferTriggerActivationEnums, [133](#)
- spinTransferTriggerModeEnums, [133](#)
- spinTransferTriggerSelectorEnums, [133](#)
- spinTransferTriggerSourceEnums, [134](#)
- spinTriggerActivationEnums, [135](#)
- spinTriggerModeEnums, [135](#)
- spinTriggerOverlapEnums, [136](#)
- spinTriggerSelectorEnums, [136](#)
- spinTriggerSourceEnums, [136](#)
- spinUserOutputSelectorEnums, [137](#)
- spinUserSetDefaultEnums, [137](#)
- spinUserSetSelectorEnums, [138](#)
- spinWhiteClipSelectorEnums, [138](#)
- CameraList Access, [168](#)
  - spinCameraListAppend, [168](#)
  - spinCameraListClear, [169](#)
  - spinCameraListCreateEmpty, [169](#)
  - spinCameraListDestroy, [170](#)
  - spinCameraListGet, [170](#)
  - spinCameraListGetBySerial, [171](#)
  - spinCameraListGetSize, [171](#)
  - spinCameraListRemove, [172](#)
  - spinCameraListRemoveBySerial, [172](#)
- Chunk data access, [248](#)
  - spinImageChunkDataGetFloatValue, [248](#)
  - spinImageChunkDataGetIntValue, [248](#)
- Chunk Data Structures, [139](#)
- ChunkBlackLevel
  - quickSpin, [369](#)
- ChunkBlackLevelSelector
  - quickSpin, [370](#)
- ChunkCRC
  - quickSpin, [370](#)
- ChunkCompressionMode
  - quickSpin, [370](#)
- ChunkCompressionRatio
  - quickSpin, [370](#)
- ChunkCounterSelector
  - quickSpin, [370](#)
- ChunkCounterValue
  - quickSpin, [370](#)
- ChunkEnable
  - quickSpin, [370](#)
- ChunkEncoderSelector
  - quickSpin, [370](#)
- ChunkEncoderStatus
  - quickSpin, [371](#)
- ChunkEncoderValue
  - quickSpin, [371](#)
- ChunkExposureEndLineStatusAll
  - quickSpin, [371](#)
- ChunkExposureTime
  - quickSpin, [371](#)
- ChunkExposureTimeSelector
  - quickSpin, [371](#)
- ChunkFrameID
  - quickSpin, [371](#)
- ChunkGain
  - quickSpin, [371](#)
- ChunkGainSelector
  - quickSpin, [371](#)
- ChunkHeight
  - quickSpin, [372](#)
- ChunkImage
  - quickSpin, [372](#)
- ChunkImageComponent
  - quickSpin, [372](#)
- ChunkInferenceBoundingBoxResult
  - quickSpin, [372](#)
- ChunkInferenceConfidence
  - quickSpin, [372](#)
- ChunkInferenceFrameId
  - quickSpin, [372](#)
- ChunkInferenceResult
  - quickSpin, [372](#)
- ChunkLinePitch
  - quickSpin, [372](#)
- ChunkLineStatusAll
  - quickSpin, [373](#)
- ChunkModeActive
  - quickSpin, [373](#)
- ChunkOffsetX
  - quickSpin, [373](#)
- ChunkOffsetY
  - quickSpin, [373](#)
- ChunkPartSelector
  - quickSpin, [373](#)
- ChunkPixelDynamicRangeMax
  - quickSpin, [373](#)
- ChunkPixelDynamicRangeMin
  - quickSpin, [373](#)
- ChunkPixelFormat
  - quickSpin, [373](#)
- ChunkRegionID
  - quickSpin, [374](#)
- ChunkScan3dAxisMax
  - quickSpin, [374](#)
- ChunkScan3dAxisMin
  - quickSpin, [374](#)
- ChunkScan3dCoordinateOffset
  - quickSpin, [374](#)
- ChunkScan3dCoordinateReferenceSelector
  - quickSpin, [374](#)
- ChunkScan3dCoordinateReferenceValue
  - quickSpin, [374](#)
- ChunkScan3dCoordinateScale
  - quickSpin, [374](#)
- ChunkScan3dCoordinateSelector

- quickSpin, [374](#)
- ChunkScan3dCoordinateSystem
  - quickSpin, [375](#)
- ChunkScan3dCoordinateSystemReference
  - quickSpin, [375](#)
- ChunkScan3dCoordinateTransformSelector
  - quickSpin, [375](#)
- ChunkScan3dDistanceUnit
  - quickSpin, [375](#)
- ChunkScan3dInvalidDataFlag
  - quickSpin, [375](#)
- ChunkScan3dInvalidDataValue
  - quickSpin, [375](#)
- ChunkScan3dOutputMode
  - quickSpin, [375](#)
- ChunkScan3dTransformValue
  - quickSpin, [375](#)
- ChunkScanLineSelector
  - quickSpin, [376](#)
- ChunkSelector
  - quickSpin, [376](#)
- ChunkSequencerSetActive
  - quickSpin, [376](#)
- ChunkSerialData
  - quickSpin, [376](#)
- ChunkSerialDataLength
  - quickSpin, [376](#)
- ChunkSerialReceiveOverflow
  - quickSpin, [376](#)
- ChunkSourceID
  - quickSpin, [376](#)
- ChunkStreamChannelID
  - quickSpin, [376](#)
- ChunkTimerSelector
  - quickSpin, [377](#)
- ChunkTimerValue
  - quickSpin, [377](#)
- ChunkTimestamp
  - quickSpin, [377](#)
- ChunkTimestampLatchValue
  - quickSpin, [377](#)
- ChunkTransferBlockID
  - quickSpin, [377](#)
- ChunkTransferQueueCurrentBlockCount
  - quickSpin, [377](#)
- ChunkTransferStreamID
  - quickSpin, [377](#)
- ChunkWidth
  - quickSpin, [377](#)
- CIConfiguration
  - quickSpin, [378](#)
- CITimeSlotsCount
  - quickSpin, [378](#)
- ColorTransformationEnable
  - quickSpin, [378](#)
- ColorTransformationSelector
  - quickSpin, [378](#)
- ColorTransformationValue
  - quickSpin, [378](#)
- ColorTransformationValueSelector
  - quickSpin, [378](#)
- compression
  - spinTIFFOption, [485](#)
- compressionLevel
  - spinPNGOption, [483](#)
- CompressionRatio
  - quickSpin, [378](#)
- CompressionSaturationPriority
  - quickSpin, [378](#)
- CounterDelay
  - quickSpin, [379](#)
- CounterDuration
  - quickSpin, [379](#)
- CounterEventActivation
  - quickSpin, [379](#)
- CounterEventSource
  - quickSpin, [379](#)
- CounterReset
  - quickSpin, [379](#)
- CounterResetActivation
  - quickSpin, [379](#)
- CounterResetSource
  - quickSpin, [379](#)
- CounterSelector
  - quickSpin, [379](#)
- CounterStatus
  - quickSpin, [380](#)
- CounterTriggerActivation
  - quickSpin, [380](#)
- CounterTriggerSource
  - quickSpin, [380](#)
- CounterValue
  - quickSpin, [380](#)
- CounterValueAtReset
  - quickSpin, [380](#)
- CxpConnectionSelector
  - quickSpin, [380](#)
- CxpConnectionTestErrorCount
  - quickSpin, [380](#)
- CxpConnectionTestMode
  - quickSpin, [380](#)
- CxpConnectionTestPacketCount
  - quickSpin, [381](#)
- CxpLinkConfiguration
  - quickSpin, [381](#)
- CxpLinkConfigurationPreferred
  - quickSpin, [381](#)
- CxpLinkConfigurationStatus
  - quickSpin, [381](#)
- CxpPoCxpAuto
  - quickSpin, [381](#)
- CxpPoCxpStatus
  - quickSpin, [381](#)
- CxpPoCxpTripReset
  - quickSpin, [381](#)
- CxpPoCxpTurnOff

- quickSpin, [381](#)
- DecimationHorizontal
  - quickSpin, [382](#)
- DecimationHorizontalMode
  - quickSpin, [382](#)
- DecimationSelector
  - quickSpin, [382](#)
- DecimationVertical
  - quickSpin, [382](#)
- DecimationVerticalMode
  - quickSpin, [382](#)
- DefectCorrectStaticEnable
  - quickSpin, [382](#)
- DefectCorrectionMode
  - quickSpin, [382](#)
- DefectTableApply
  - quickSpin, [382](#)
- DefectTableCoordinateX
  - quickSpin, [383](#)
- DefectTableCoordinateY
  - quickSpin, [383](#)
- DefectTableFactoryRestore
  - quickSpin, [383](#)
- DefectTableIndex
  - quickSpin, [383](#)
- DefectTablePixelCount
  - quickSpin, [383](#)
- DefectTableSave
  - quickSpin, [383](#)
- Deinterlacing
  - quickSpin, [383](#)
- Device Event Data Access, [245](#)
  - spinDeviceEventGetId, [245](#)
  - spinDeviceEventGetName, [246](#)
  - spinDeviceEventGetPayloadData, [246](#)
  - spinDeviceEventGetPayloadDataSize, [247](#)
- DeviceAccessStatus
  - quickSpinTLDevice, [447](#)
  - quickSpinTLInterface, [454](#)
- DeviceAddress
  - actionCommandResult, [349](#)
- DeviceCharacterSet
  - quickSpin, [383](#)
- DeviceClockFrequency
  - quickSpin, [384](#)
- DeviceClockSelector
  - quickSpin, [384](#)
- DeviceConnectionSelector
  - quickSpin, [384](#)
- DeviceConnectionSpeed
  - quickSpin, [384](#)
- DeviceConnectionStatus
  - quickSpin, [384](#)
- DeviceCount
  - quickSpinTLInterface, [454](#)
- DeviceCurrentSpeed
  - quickSpinTLDevice, [448](#)
- DeviceDisplayName
  - quickSpinTLDevice, [448](#)
- DeviceDriverVersion
  - quickSpinTLDevice, [448](#)
- DeviceEndianessMechanism
  - quickSpinTLDevice, [448](#)
- DeviceEventChannelCount
  - quickSpin, [384](#)
- DeviceFamilyName
  - quickSpin, [384](#)
- DeviceFeaturePersistenceEnd
  - quickSpin, [384](#)
- DeviceFeaturePersistenceStart
  - quickSpin, [385](#)
- DeviceFirmwareVersion
  - quickSpin, [385](#)
- DeviceGenCPVersionMajor
  - quickSpin, [385](#)
- DeviceGenCPVersionMinor
  - quickSpin, [385](#)
- DeviceId
  - quickSpin, [385](#)
  - quickSpinTLDevice, [448](#)
  - quickSpinTLInterface, [454](#)
- DeviceIndicatorMode
  - quickSpin, [385](#)
- DeviceInstanceId
  - quickSpinTLDevice, [448](#)
- DevicesUpdater
  - quickSpinTLDevice, [448](#)
- DeviceLinkBandwidthReserve
  - quickSpin, [385](#)
- DeviceLinkCommandTimeout
  - quickSpin, [385](#)
- DeviceLinkConnectionCount
  - quickSpin, [386](#)
- DeviceLinkCurrentThroughput
  - quickSpin, [386](#)
- DeviceLinkHeartbeatMode
  - quickSpin, [386](#)
- DeviceLinkHeartbeatTimeout
  - quickSpin, [386](#)
- DeviceLinkSelector
  - quickSpin, [386](#)
- DeviceLinkSpeed
  - quickSpin, [386](#)
  - quickSpinTLDevice, [448](#)
- DeviceLinkThroughputLimit
  - quickSpin, [386](#)
- DeviceLinkThroughputLimitMode
  - quickSpin, [386](#)
- DeviceLocation
  - quickSpinTLDevice, [449](#)
- DeviceManifestEntrySelector
  - quickSpin, [387](#)
- DeviceManifestPrimaryURL
  - quickSpin, [387](#)
- DeviceManifestSchemaMajorVersion
  - quickSpin, [387](#)

- DeviceManifestSchemaMinorVersion
  - [quickSpin, 387](#)
- DeviceManifestSecondaryURL
  - [quickSpin, 387](#)
- DeviceManifestXMLMajorVersion
  - [quickSpin, 387](#)
- DeviceManifestXMLMinorVersion
  - [quickSpin, 387](#)
- DeviceManifestXMLSubMinorVersion
  - [quickSpin, 387](#)
- DeviceManufacturerInfo
  - [quickSpin, 388](#)
- DeviceMaxThroughput
  - [quickSpin, 388](#)
- DeviceModelName
  - [quickSpin, 388](#)
  - [quickSpinTLDevice, 449](#)
  - [quickSpinTLInterface, 454](#)
- DeviceMulticastMonitorMode
  - [quickSpinTLDevice, 449](#)
- DevicePortId
  - [quickSpinTLDevice, 449](#)
- DevicePowerSupplySelector
  - [quickSpin, 388](#)
- DeviceRegistersCheck
  - [quickSpin, 388](#)
- DeviceRegistersEndianness
  - [quickSpin, 388](#)
- DeviceRegistersStreamingEnd
  - [quickSpin, 388](#)
- DeviceRegistersStreamingStart
  - [quickSpin, 388](#)
- DeviceRegistersValid
  - [quickSpin, 389](#)
- DeviceReset
  - [quickSpin, 389](#)
- DeviceSFNCVersionMajor
  - [quickSpin, 389](#)
- DeviceSFNCVersionMinor
  - [quickSpin, 389](#)
- DeviceSFNCVersionSubMinor
  - [quickSpin, 390](#)
- DeviceScanType
  - [quickSpin, 389](#)
- DeviceSelector
  - [quickSpinTLInterface, 455](#)
- DeviceSerialNumber
  - [quickSpin, 389](#)
  - [quickSpinTLDevice, 449](#)
  - [quickSpinTLInterface, 455](#)
- DeviceSerialPortBaudRate
  - [quickSpin, 389](#)
- DeviceSerialPortSelector
  - [quickSpin, 389](#)
- DeviceStreamChannelCount
  - [quickSpin, 390](#)
- DeviceStreamChannelEndianness
  - [quickSpin, 390](#)
- DeviceStreamChannelLink
  - [quickSpin, 390](#)
- DeviceStreamChannelPacketSize
  - [quickSpin, 390](#)
- DeviceStreamChannelSelector
  - [quickSpin, 390](#)
- DeviceStreamChannelType
  - [quickSpin, 390](#)
- DeviceTLType
  - [quickSpin, 391](#)
- DeviceTLVersionMajor
  - [quickSpin, 391](#)
- DeviceTLVersionMinor
  - [quickSpin, 391](#)
- DeviceTLVersionSubMinor
  - [quickSpin, 391](#)
- DeviceTapGeometry
  - [quickSpin, 390](#)
- DeviceTemperature
  - [quickSpin, 391](#)
- DeviceTemperatureSelector
  - [quickSpin, 391](#)
- DeviceType
  - [quickSpin, 391](#)
  - [quickSpinTLDevice, 449](#)
- DeviceU3VProtocol
  - [quickSpinTLDevice, 449](#)
- DeviceUnlock
  - [quickSpinTLInterface, 455](#)
- DeviceUpdateList
  - [quickSpinTLInterface, 455](#)
- DeviceUptime
  - [quickSpin, 391](#)
- DeviceUserID
  - [quickSpin, 392](#)
  - [quickSpinTLDevice, 449](#)
- DeviceVendorName
  - [quickSpin, 392](#)
  - [quickSpinTLDevice, 450](#)
  - [quickSpinTLInterface, 455](#)
- DeviceVersion
  - [quickSpin, 392](#)
  - [quickSpinTLDevice, 450](#)
- [doc/spindocs/C/GettingStarted.dox, 487](#)
- [doc/spindocs/C/ProgrammerGuide.dox, 487](#)
- [doc/spindocs/shared/Benefits.dox, 487](#)
- [doc/spindocs/shared/FlyCapture2Comparison.dox, 487](#)
- [doc/spindocs/shared/GenICamGenTL.dox, 487](#)
- [doc/spindocs/shared/Licensing.dox, 487](#)
- [doc/spindocs/shared/Maintenance.dox, 487](#)
- EncoderDivider
  - [quickSpin, 392](#)
- EncoderMode
  - [quickSpin, 392](#)
- EncoderOutputMode
  - [quickSpin, 392](#)
- EncoderReset
  - [quickSpin, 392](#)

- EncoderResetActivation
  - quickSpin, [392](#)
- EncoderResetSource
  - quickSpin, [393](#)
- EncoderSelector
  - quickSpin, [393](#)
- EncoderSourceA
  - quickSpin, [393](#)
- EncoderSourceB
  - quickSpin, [393](#)
- EncoderStatus
  - quickSpin, [393](#)
- EncoderTimeout
  - quickSpin, [393](#)
- EncoderValue
  - quickSpin, [393](#)
- EncoderValueAtReset
  - quickSpin, [393](#)
- EnumerateGEVInterfaces
  - quickSpinTLSystem, [465](#)
- EnumerateUSBInterfaces
  - quickSpinTLSystem, [465](#)
- EnumerationCount
  - quickSpin, [394](#)
- Error Handling, [145](#)
  - spinErrorGetLast, [145](#)
  - spinErrorGetLastBuildDate, [146](#)
  - spinErrorGetLastBuildTime, [146](#)
  - spinErrorGetLastFileName, [147](#)
  - spinErrorGetLastFullMessage, [147](#)
  - spinErrorGetLastFunctionName, [148](#)
  - spinErrorGetLastLineNumber, [148](#)
  - spinErrorGetLastMessage, [149](#)
- Event Access, [224](#)
  - spinDeviceArrivalEventHandlerCreate, [224](#)
  - spinDeviceArrivalEventHandlerDestroy, [225](#)
  - spinDeviceEventHandlerCreate, [225](#)
  - spinDeviceEventHandlerDestroy, [226](#)
  - spinDeviceRemovalEventHandlerCreate, [226](#)
  - spinDeviceRemovalEventHandlerDestroy, [227](#)
  - spinImageEventHandlerCreate, [227](#)
  - spinImageEventHandlerDestroy, [228](#)
  - spinInterfaceEventHandlerCreate, [228](#)
  - spinInterfaceEventHandlerDestroy, [229](#)
  - spinLogEventHandlerCreate, [229](#)
  - spinLogEventHandlerDestroy, [230](#)
- EventAcquisitionEnd
  - quickSpin, [394](#)
- EventAcquisitionEndFrameID
  - quickSpin, [394](#)
- EventAcquisitionEndTimestamp
  - quickSpin, [394](#)
- EventAcquisitionError
  - quickSpin, [394](#)
- EventAcquisitionErrorFrameID
  - quickSpin, [394](#)
- EventAcquisitionErrorTimestamp
  - quickSpin, [394](#)
- EventAcquisitionStart
  - quickSpin, [394](#)
- EventAcquisitionStartFrameID
  - quickSpin, [395](#)
- EventAcquisitionStartTimestamp
  - quickSpin, [395](#)
- EventAcquisitionTransferEnd
  - quickSpin, [395](#)
- EventAcquisitionTransferEndFrameID
  - quickSpin, [395](#)
- EventAcquisitionTransferEndTimestamp
  - quickSpin, [395](#)
- EventAcquisitionTransferStart
  - quickSpin, [395](#)
- EventAcquisitionTransferStartFrameID
  - quickSpin, [395](#)
- EventAcquisitionTransferStartTimestamp
  - quickSpin, [395](#)
- EventAcquisitionTrigger
  - quickSpin, [396](#)
- EventAcquisitionTriggerFrameID
  - quickSpin, [396](#)
- EventAcquisitionTriggerTimestamp
  - quickSpin, [396](#)
- EventActionLate
  - quickSpin, [396](#)
- EventActionLateFrameID
  - quickSpin, [396](#)
- EventActionLateTimestamp
  - quickSpin, [396](#)
- EventCounter0End
  - quickSpin, [396](#)
- EventCounter0EndFrameID
  - quickSpin, [396](#)
- EventCounter0EndTimestamp
  - quickSpin, [397](#)
- EventCounter0Start
  - quickSpin, [397](#)
- EventCounter0StartFrameID
  - quickSpin, [397](#)
- EventCounter0StartTimestamp
  - quickSpin, [397](#)
- EventCounter1End
  - quickSpin, [397](#)
- EventCounter1EndFrameID
  - quickSpin, [397](#)
- EventCounter1EndTimestamp
  - quickSpin, [397](#)
- EventCounter1Start
  - quickSpin, [397](#)
- EventCounter1StartFrameID
  - quickSpin, [398](#)
- EventCounter1StartTimestamp
  - quickSpin, [398](#)
- EventEncoder0Restarted
  - quickSpin, [398](#)
- EventEncoder0RestartedFrameID
  - quickSpin, [398](#)

EventEncoder0RestartedTimestamp  
    [quickSpin, 398](#)

EventEncoder0Stopped  
    [quickSpin, 398](#)

EventEncoder0StoppedFrameID  
    [quickSpin, 398](#)

EventEncoder0StoppedTimestamp  
    [quickSpin, 398](#)

EventEncoder1Restarted  
    [quickSpin, 399](#)

EventEncoder1RestartedFrameID  
    [quickSpin, 399](#)

EventEncoder1RestartedTimestamp  
    [quickSpin, 399](#)

EventEncoder1Stopped  
    [quickSpin, 399](#)

EventEncoder1StoppedFrameID  
    [quickSpin, 399](#)

EventEncoder1StoppedTimestamp  
    [quickSpin, 399](#)

EventError  
    [quickSpin, 399](#)

EventErrorCode  
    [quickSpin, 399](#)

EventErrorFrameID  
    [quickSpin, 400](#)

EventErrorTimestamp  
    [quickSpin, 400](#)

EventExposureEnd  
    [quickSpin, 400](#)

EventExposureEndFrameID  
    [quickSpin, 400](#)

EventExposureEndTimestamp  
    [quickSpin, 400](#)

EventExposureStart  
    [quickSpin, 400](#)

EventExposureStartFrameID  
    [quickSpin, 400](#)

EventExposureStartTimestamp  
    [quickSpin, 400](#)

EventFrameBurstEnd  
    [quickSpin, 401](#)

EventFrameBurstEndFrameID  
    [quickSpin, 401](#)

EventFrameBurstEndTimestamp  
    [quickSpin, 401](#)

EventFrameBurstStart  
    [quickSpin, 401](#)

EventFrameBurstStartFrameID  
    [quickSpin, 401](#)

EventFrameBurstStartTimestamp  
    [quickSpin, 401](#)

EventFrameEnd  
    [quickSpin, 401](#)

EventFrameEndFrameID  
    [quickSpin, 401](#)

EventFrameEndTimestamp  
    [quickSpin, 402](#)

EventFrameStart  
    [quickSpin, 402](#)

EventFrameStartFrameID  
    [quickSpin, 402](#)

EventFrameStartTimestamp  
    [quickSpin, 402](#)

EventFrameTransferEnd  
    [quickSpin, 402](#)

EventFrameTransferEndFrameID  
    [quickSpin, 402](#)

EventFrameTransferEndTimestamp  
    [quickSpin, 402](#)

EventFrameTransferStart  
    [quickSpin, 402](#)

EventFrameTransferStartFrameID  
    [quickSpin, 403](#)

EventFrameTransferStartTimestamp  
    [quickSpin, 403](#)

EventFrameTrigger  
    [quickSpin, 403](#)

EventFrameTriggerFrameID  
    [quickSpin, 403](#)

EventFrameTriggerTimestamp  
    [quickSpin, 403](#)

EventLine0AnyEdge  
    [quickSpin, 403](#)

EventLine0AnyEdgeFrameID  
    [quickSpin, 403](#)

EventLine0AnyEdgeTimestamp  
    [quickSpin, 403](#)

EventLine0FallingEdge  
    [quickSpin, 404](#)

EventLine0FallingEdgeFrameID  
    [quickSpin, 404](#)

EventLine0FallingEdgeTimestamp  
    [quickSpin, 404](#)

EventLine0RisingEdge  
    [quickSpin, 404](#)

EventLine0RisingEdgeFrameID  
    [quickSpin, 404](#)

EventLine0RisingEdgeTimestamp  
    [quickSpin, 404](#)

EventLine1AnyEdge  
    [quickSpin, 404](#)

EventLine1AnyEdgeFrameID  
    [quickSpin, 404](#)

EventLine1AnyEdgeTimestamp  
    [quickSpin, 405](#)

EventLine1FallingEdge  
    [quickSpin, 405](#)

EventLine1FallingEdgeFrameID  
    [quickSpin, 405](#)

EventLine1FallingEdgeTimestamp  
    [quickSpin, 405](#)

EventLine1RisingEdge  
    [quickSpin, 405](#)

EventLine1RisingEdgeFrameID  
    [quickSpin, 405](#)

- EventLine1RisingEdgeTimestamp
  - [quickSpin](#), 405
- EventLinkSpeedChange
  - [quickSpin](#), 405
- EventLinkSpeedChangeFrameID
  - [quickSpin](#), 406
- EventLinkSpeedChangeTimestamp
  - [quickSpin](#), 406
- EventLinkTrigger0
  - [quickSpin](#), 406
- EventLinkTrigger0FrameID
  - [quickSpin](#), 406
- EventLinkTrigger0Timestamp
  - [quickSpin](#), 406
- EventLinkTrigger1
  - [quickSpin](#), 406
- EventLinkTrigger1FrameID
  - [quickSpin](#), 406
- EventLinkTrigger1Timestamp
  - [quickSpin](#), 406
- EventNotification
  - [quickSpin](#), 407
- EventSelector
  - [quickSpin](#), 407
- EventSequencerSetChange
  - [quickSpin](#), 407
- EventSequencerSetChangeFrameID
  - [quickSpin](#), 407
- EventSequencerSetChangeTimestamp
  - [quickSpin](#), 407
- EventSerialData
  - [quickSpin](#), 407
- EventSerialDataLength
  - [quickSpin](#), 407
- EventSerialPortReceive
  - [quickSpin](#), 407
- EventSerialPortReceiveTimestamp
  - [quickSpin](#), 408
- EventSerialReceiveOverflow
  - [quickSpin](#), 408
- EventStream0TransferBlockEnd
  - [quickSpin](#), 408
- EventStream0TransferBlockEndFrameID
  - [quickSpin](#), 408
- EventStream0TransferBlockEndTimestamp
  - [quickSpin](#), 408
- EventStream0TransferBlockStart
  - [quickSpin](#), 408
- EventStream0TransferBlockStartFrameID
  - [quickSpin](#), 408
- EventStream0TransferBlockStartTimestamp
  - [quickSpin](#), 408
- EventStream0TransferBlockTrigger
  - [quickSpin](#), 409
- EventStream0TransferBlockTriggerFrameID
  - [quickSpin](#), 409
- EventStream0TransferBlockTriggerTimestamp
  - [quickSpin](#), 409
- EventStream0TransferBurstEnd
  - [quickSpin](#), 409
- EventStream0TransferBurstEndFrameID
  - [quickSpin](#), 409
- EventStream0TransferBurstEndTimestamp
  - [quickSpin](#), 409
- EventStream0TransferBurstStart
  - [quickSpin](#), 409
- EventStream0TransferBurstStartFrameID
  - [quickSpin](#), 409
- EventStream0TransferBurstStartTimestamp
  - [quickSpin](#), 410
- EventStream0TransferEnd
  - [quickSpin](#), 410
- EventStream0TransferEndFrameID
  - [quickSpin](#), 410
- EventStream0TransferEndTimestamp
  - [quickSpin](#), 410
- EventStream0TransferOverflow
  - [quickSpin](#), 410
- EventStream0TransferOverflowFrameID
  - [quickSpin](#), 410
- EventStream0TransferOverflowTimestamp
  - [quickSpin](#), 410
- EventStream0TransferPause
  - [quickSpin](#), 410
- EventStream0TransferPauseFrameID
  - [quickSpin](#), 411
- EventStream0TransferPauseTimestamp
  - [quickSpin](#), 411
- EventStream0TransferResume
  - [quickSpin](#), 411
- EventStream0TransferResumeFrameID
  - [quickSpin](#), 411
- EventStream0TransferResumeTimestamp
  - [quickSpin](#), 411
- EventStream0TransferStart
  - [quickSpin](#), 411
- EventStream0TransferStartFrameID
  - [quickSpin](#), 411
- EventStream0TransferStartTimestamp
  - [quickSpin](#), 411
- EventTest
  - [quickSpin](#), 412
- EventTestTimestamp
  - [quickSpin](#), 412
- EventTimer0End
  - [quickSpin](#), 412
- EventTimer0EndFrameID
  - [quickSpin](#), 412
- EventTimer0EndTimestamp
  - [quickSpin](#), 412
- EventTimer0Start
  - [quickSpin](#), 412
- EventTimer0StartFrameID
  - [quickSpin](#), 412
- EventTimer0StartTimestamp
  - [quickSpin](#), 412

- EventTimer1End
  - quickSpin, [413](#)
- EventTimer1EndFrameID
  - quickSpin, [413](#)
- EventTimer1EndTimestamp
  - quickSpin, [413](#)
- EventTimer1Start
  - quickSpin, [413](#)
- EventTimer1StartFrameID
  - quickSpin, [413](#)
- EventTimer1StartTimestamp
  - quickSpin, [413](#)
- ExposureActiveMode
  - quickSpin, [413](#)
- ExposureAuto
  - quickSpin, [413](#)
- ExposureMode
  - quickSpin, [414](#)
- ExposureTime
  - quickSpin, [414](#)
- ExposureTimeMode
  - quickSpin, [414](#)
- ExposureTimeSelector
  - quickSpin, [414](#)
- FactoryReset
  - quickSpin, [414](#)
- False
  - Spinnaker C Definitions, [22](#)
- FileAccessBuffer
  - quickSpin, [414](#)
- FileAccessLength
  - quickSpin, [414](#)
- FileAccessOffset
  - quickSpin, [414](#)
- FileOpenMode
  - quickSpin, [415](#)
- FileOperationExecute
  - quickSpin, [415](#)
- FileOperationResult
  - quickSpin, [415](#)
- FileOperationSelector
  - quickSpin, [415](#)
- FileOperationStatus
  - quickSpin, [415](#)
- FileSelector
  - quickSpin, [415](#)
- FileSize
  - quickSpin, [415](#)
- FilterDriverStatus
  - quickSpinTLInterface, [455](#)
- frameRate
  - spinAVIOption, [469](#)
  - spinH264Option, [477](#)
  - spinMJPEGOption, [481](#)
- GUIXMLLocation
  - quickSpinTLDevice, [452](#)
- GUIXMLPath
  - quickSpinTLDevice, [453](#)
- Gain
  - quickSpin, [415](#)
- GainAuto
  - quickSpin, [416](#)
- GainAutoBalance
  - quickSpin, [416](#)
- GainSelector
  - quickSpin, [416](#)
- Gamma
  - quickSpin, [416](#)
- GammaEnable
  - quickSpin, [416](#)
- GenICamXMLLocation
  - quickSpinTLDevice, [450](#)
- GenICamXMLPath
  - quickSpinTLDevice, [450](#)
- GenTLFNCVersionMajor
  - quickSpinTLSystem, [465](#)
- GenTLFNCVersionMinor
  - quickSpinTLSystem, [465](#)
- GenTLFNCVersionSubMinor
  - quickSpinTLSystem, [465](#)
- GenTLVersionMajor
  - quickSpinTLSystem, [465](#)
- GenTLVersionMinor
  - quickSpinTLSystem, [466](#)
- GevActionDeviceKey
  - quickSpinTLInterface, [455](#)
- GevActionGroupKey
  - quickSpinTLInterface, [455](#)
- GevActionGroupMask
  - quickSpinTLInterface, [456](#)
- GevActionTime
  - quickSpinTLInterface, [456](#)
- GevActiveLinkCount
  - quickSpin, [416](#)
- GevCCP
  - quickSpin, [416](#)
  - quickSpinTLDevice, [450](#)
- GevCurrentDefaultGateway
  - quickSpin, [416](#)
- GevCurrentIPAddress
  - quickSpin, [417](#)
- GevCurrentIPConfigurationDHCP
  - quickSpin, [417](#)
- GevCurrentIPConfigurationLLA
  - quickSpin, [417](#)
- GevCurrentIPConfigurationPersistentIP
  - quickSpin, [417](#)
- GevCurrentPhysicalLinkConfiguration
  - quickSpin, [417](#)
- GevCurrentSubnetMask
  - quickSpin, [417](#)
- GevDeviceAutoForceIP
  - quickSpinTLDevice, [450](#)
  - quickSpinTLInterface, [456](#)
- GevDeviceDiscoverMaximumPacketSize

- quickSpinTLDevice, [450](#)
- GevDeviceForceGateway
  - quickSpinTLDevice, [450](#)
  - quickSpinTLInterface, [456](#)
- GevDeviceForceIPAddress
  - quickSpinTLDevice, [451](#)
  - quickSpinTLInterface, [456](#)
- GevDeviceForceIP
  - quickSpinTLDevice, [451](#)
  - quickSpinTLInterface, [456](#)
- GevDeviceForceSubnetMask
  - quickSpinTLDevice, [451](#)
  - quickSpinTLInterface, [456](#)
- GevDeviceGateway
  - quickSpinTLDevice, [451](#)
  - quickSpinTLInterface, [456](#)
- GevDeviceIPAddress
  - quickSpinTLDevice, [451](#)
  - quickSpinTLInterface, [457](#)
- GevDeviceIsWrongSubnet
  - quickSpinTLDevice, [451](#)
- GevDeviceMACAddress
  - quickSpinTLDevice, [451](#)
  - quickSpinTLInterface, [457](#)
- GevDeviceMaximumPacketSize
  - quickSpinTLDevice, [451](#)
- GevDeviceMaximumRetryCount
  - quickSpinTLDevice, [452](#)
- GevDeviceModelsBigEndian
  - quickSpinTLDevice, [452](#)
- GevDevicePort
  - quickSpinTLDevice, [452](#)
- GevDeviceReadAndWriteTimeout
  - quickSpinTLDevice, [452](#)
- GevDeviceSubnetMask
  - quickSpinTLDevice, [452](#)
  - quickSpinTLInterface, [457](#)
- GevDiscoveryAckDelay
  - quickSpin, [417](#)
- GevFailedPacketCount
  - quickSpinTLStream, [461](#)
- GevFirstURL
  - quickSpin, [417](#)
- GevGVCPExtendedStatusCodes
  - quickSpin, [418](#)
- GevGVCPExtendedStatusCodesSelector
  - quickSpin, [418](#)
- GevGVCPHeartbeatDisable
  - quickSpin, [418](#)
- GevGVCPPendingAck
  - quickSpin, [418](#)
- GevGVCPPendingTimeout
  - quickSpin, [418](#)
- GevGVSPExtendedIDMode
  - quickSpin, [418](#)
- GevHeartbeatTimeout
  - quickSpin, [418](#)
- GevIEEE1588
  - quickSpin, [418](#)
- GevIEEE1588ClockAccuracy
  - quickSpin, [419](#)
- GevIEEE1588Mode
  - quickSpin, [419](#)
- GevIEEE1588Status
  - quickSpin, [419](#)
- GevIPConfigurationStatus
  - quickSpin, [419](#)
- GevInterfaceDefaultGateway
  - quickSpinTLSystem, [466](#)
- GevInterfaceDefaultIPAddress
  - quickSpinTLSystem, [466](#)
- GevInterfaceDefaultSubnetMask
  - quickSpinTLSystem, [466](#)
- GevInterfaceGateway
  - quickSpinTLInterface, [457](#)
- GevInterfaceGatewaySelector
  - quickSpinTLInterface, [457](#)
- GevInterfaceMACAddress
  - quickSpinTLInterface, [457](#)
  - quickSpinTLSystem, [466](#)
- GevInterfaceMTU
  - quickSpinTLInterface, [457](#)
- GevInterfaceReceiveLinkSpeed
  - quickSpinTLInterface, [457](#)
- GevInterfaceSelector
  - quickSpin, [419](#)
- GevInterfaceSubnetIPAddress
  - quickSpinTLInterface, [458](#)
- GevInterfaceSubnetMask
  - quickSpinTLInterface, [458](#)
- GevInterfaceSubnetSelector
  - quickSpinTLInterface, [458](#)
- GevInterfaceTransmitLinkSpeed
  - quickSpinTLInterface, [458](#)
- GevMACAddress
  - quickSpin, [419](#)
- GevMCDA
  - quickSpin, [419](#)
- GevMCPHostPort
  - quickSpin, [419](#)
- GevMCRC
  - quickSpin, [420](#)
- GevMCSP
  - quickSpin, [420](#)
- GevMCTT
  - quickSpin, [420](#)
- GevMaximumNumberResendRequests
  - quickSpinTLStream, [461](#)
- GevNumberOfInterfaces
  - quickSpin, [420](#)
- GevPAUSEFrameReception
  - quickSpin, [420](#)
- GevPAUSEFrameTransmission
  - quickSpin, [420](#)
- GevPacketResendMode
  - quickSpinTLStream, [461](#)

- GevPacketResendTimeout
  - quickSpinTLStream, [461](#)
- GevPersistentDefaultGateway
  - quickSpin, [420](#)
- GevPersistentIPAddress
  - quickSpin, [420](#)
- GevPersistentSubnetMask
  - quickSpin, [421](#)
- GevPhysicalLinkConfiguration
  - quickSpin, [421](#)
- GevPrimaryApplicationIPAddress
  - quickSpin, [421](#)
- GevPrimaryApplicationSocket
  - quickSpin, [421](#)
- GevPrimaryApplicationSwitchoverKey
  - quickSpin, [421](#)
- GevResendPacketCount
  - quickSpinTLStream, [461](#)
- GevResendRequestCount
  - quickSpinTLStream, [461](#)
- GevSCCFGAllInTransmission
  - quickSpin, [421](#)
- GevSCCFGExtendedChunkData
  - quickSpin, [421](#)
- GevSCCFGPacketResendDestination
  - quickSpin, [421](#)
- GevSCCFGUnconditionalStreaming
  - quickSpin, [422](#)
- GevSCDA
  - quickSpin, [422](#)
- GevSCPDDirection
  - quickSpin, [422](#)
- GevSCPHostPort
  - quickSpin, [422](#)
- GevSCPIInterfaceIndex
  - quickSpin, [422](#)
- GevSCPSBigEndian
  - quickSpin, [422](#)
- GevSCPSDoNotFragment
  - quickSpin, [422](#)
- GevSCPSFireTestPacket
  - quickSpin, [423](#)
- GevSCPSPacketSize
  - quickSpin, [423](#)
- GevSCPD
  - quickSpin, [422](#)
- GevSCSP
  - quickSpin, [423](#)
- GevSCZoneConfigurationLock
  - quickSpin, [423](#)
- GevSCZoneCount
  - quickSpin, [423](#)
- GevSCZoneDirectionAll
  - quickSpin, [423](#)
- GevSecondURL
  - quickSpin, [423](#)
- GevStreamChannelSelector
  - quickSpin, [423](#)
- GevSupportedOption
  - quickSpin, [424](#)
- GevSupportedOptionSelector
  - quickSpin, [424](#)
- GevTimestampTickFrequency
  - quickSpin, [424](#)
- GevTotalPacketCount
  - quickSpinTLStream, [461](#)
- GevVersionMajor
  - quickSpinTLDevice, [452](#)
  - quickSpinTLSystem, [466](#)
- GevVersionMinor
  - quickSpinTLDevice, [452](#)
  - quickSpinTLSystem, [466](#)
- GuiXmlManifestAddress
  - quickSpin, [424](#)
- Height
  - quickSpin, [424](#)
- height
  - spinH264Option, [477](#)
- HeightMax
  - quickSpin, [424](#)
- HostAdapterDriverVersion
  - quickSpinTLInterface, [458](#)
- HostAdapterName
  - quickSpinTLInterface, [458](#)
- HostAdapterVendor
  - quickSpinTLInterface, [458](#)
- IBoolean Access, [309](#)
  - spinBooleanGetValue, [309](#)
  - spinBooleanSetValue, [310](#)
- ICategory Access, [313](#)
  - spinCategoryGetFeatureByIndex, [313](#)
  - spinCategoryGetNumFeatures, [314](#)
- ICommand Access, [311](#)
  - spinCommandExecute, [311](#)
  - spinCommandIsDone, [312](#)
- IEnumEntry Access, [306](#)
  - spinEnumerationEntryGetEnumValue, [306](#)
  - spinEnumerationEntryGetIntValue, [307](#)
  - spinEnumerationEntryGetSymbolic, [307](#)
- IEnumeration Access, [301](#)
  - spinEnumerationGetCurrentEntry, [301](#)
  - spinEnumerationGetEntryByIndex, [302](#)
  - spinEnumerationGetEntryByName, [302](#)
  - spinEnumerationGetNumEntries, [303](#)
  - spinEnumerationReleaseNode, [303](#)
  - spinEnumerationSetEnumValue, [304](#)
  - spinEnumerationSetIntValue, [304](#)
- IFloat Access, [296](#)
  - spinFloatGetMax, [296](#)
  - spinFloatGetMin, [297](#)
  - spinFloatGetRepresentation, [297](#)
  - spinFloatGetUnit, [298](#)
  - spinFloatGetValue, [298](#)
  - spinFloatGetValueEx, [299](#)
  - spinFloatSetValue, [299](#)

- spinFloatSetValueEx, 300
- Integer Access, 291
  - spinIntegerGetInc, 291
  - spinIntegerGetMax, 292
  - spinIntegerGetMin, 292
  - spinIntegerGetRepresentation, 293
  - spinIntegerGetValue, 293
  - spinIntegerGetValueEx, 294
  - spinIntegerSetValue, 294
  - spinIntegerSetValueEx, 295
- IRegister Access, 315
  - spinRegisterGet, 315
  - spinRegisterGetAddress, 316
  - spinRegisterGetEx, 316
  - spinRegisterGetLength, 317
  - spinRegisterSet, 318
  - spinRegisterSetEx, 318
  - spinRegisterSetReference, 319
- IValue Access, 284
  - spinNodeFromString, 284
  - spinNodeFromStringEx, 285
  - spinNodeToString, 285
  - spinNodeToStringEx, 286
- Image Access, 194
  - spinImageCalculateStatistics, 196
  - spinImageCheckCRC, 197
  - spinImageConvert, 197
  - spinImageConvertEx, 198
  - spinImageCreate, 198
  - spinImageCreateEmpty, 199
  - spinImageCreateEx, 199
  - spinImageCreateEx2, 200
  - spinImageDeepCopy, 201
  - spinImageDestroy, 201
  - spinImageGetBitsPerPixel, 201
  - spinImageGetBufferSize, 202
  - spinImageGetChunkLayoutID, 202
  - spinImageGetColorProcessing, 203
  - spinImageGetData, 203
  - spinImageGetDefaultColorProcessing, 204
  - spinImageGetFrameID, 204
  - spinImageGetHeight, 205
  - spinImageGetID, 205
  - spinImageGetNumDecompressionThreads, 206
  - spinImageGetOffsetX, 206
  - spinImageGetOffsetY, 207
  - spinImageGetPaddingX, 207
  - spinImageGetPaddingY, 208
  - spinImageGetPayloadType, 208
  - spinImageGetPixelFormat, 209
  - spinImageGetPixelFormatName, 209
  - spinImageGetPrivateData, 210
  - spinImageGetSize, 210
  - spinImageGetStatus, 211
  - spinImageGetStatusDescription, 211
  - spinImageGetStride, 212
  - spinImageGetTLPayloadType, 213
  - spinImageGetTLPixelFormat, 213
  - spinImageGetTLPixelFormatNamespace, 214
  - spinImageGetTimeStamp, 212
  - spinImageGetValidPayloadSize, 214
  - spinImageGetWidth, 215
  - spinImageHasCRC, 215
  - spinImageIsIncomplete, 216
  - spinImageRelease, 216
  - spinImageReset, 216
  - spinImageResetEx, 217
  - spinImageSave, 218
  - spinImageSaveBmp, 218
  - spinImageSaveFromExt, 219
  - spinImageSaveJpeg, 219
  - spinImageSaveJpg2, 220
  - spinImageSavePgm, 220
  - spinImageSavePng, 221
  - spinImageSavePpm, 221
  - spinImageSaveTiff, 222
  - spinImageSetDefaultColorProcessing, 222
  - spinImageSetNumDecompressionThreads, 223
- ImageComponentEnable
  - quickSpin, 424
- ImageComponentSelector
  - quickSpin, 424
- ImageCompressionBitrate
  - quickSpin, 425
- ImageCompressionJPEGFormatOption
  - quickSpin, 425
- ImageCompressionMode
  - quickSpin, 425
- ImageCompressionQuality
  - quickSpin, 425
- ImageCompressionRateOption
  - quickSpin, 425
- ImageStatistics Access, 231
  - spinImageStatisticsCreate, 232
  - spinImageStatisticsDestroy, 232
  - spinImageStatisticsDisableAll, 232
  - spinImageStatisticsEnableAll, 233
  - spinImageStatisticsEnableGreyOnly, 233
  - spinImageStatisticsEnableHslOnly, 234
  - spinImageStatisticsEnableRgbOnly, 234
  - spinImageStatisticsGetAll, 235
  - spinImageStatisticsGetChannelStatus, 235
  - spinImageStatisticsGetHistogram, 236
  - spinImageStatisticsGetMean, 236
  - spinImageStatisticsGetNumPixelValues, 237
  - spinImageStatisticsGetPixelValueRange, 237
  - spinImageStatisticsGetRange, 238
  - spinImageStatisticsSetChannelStatus, 238
- include/spinc/CameraDefsC.h, 487
- include/spinc/ChunkDataDefC.h, 520
- include/spinc/QuickSpinC.h, 521
- include/spinc/QuickSpinDefsC.h, 521
- include/spinc/SpinVideoC.h, 546
- include/spinc/SpinnakerC.h, 523
- include/spinc/SpinnakerDefsC.h, 533
- include/spinc/SpinnakerGenApiC.h, 538

- include/spinc/SpinnakerGenApiDefsC.h, 542
- include/spinc/SpinnakerPlatformC.h, 545
- include/spinc/TransportLayerDefsC.h, 546
- include/spinc/TransportLayerDeviceC.h, 548
- include/spinc/TransportLayerInterfaceC.h, 549
- include/spinc/TransportLayerStreamC.h, 550
- include/spinc/TransportLayerSystemC.h, 550
- IncompatibleDeviceCount
  - quickSpinTLInterface, 458
- IncompatibleDeviceID
  - quickSpinTLInterface, 459
- IncompatibleDeviceModelName
  - quickSpinTLInterface, 459
- IncompatibleDeviceSelector
  - quickSpinTLInterface, 459
- IncompatibleDeviceVendorName
  - quickSpinTLInterface, 459
- IncompatibleGevDeviceIPAddress
  - quickSpinTLInterface, 459
- IncompatibleGevDeviceMACAddress
  - quickSpinTLInterface, 459
- IncompatibleGevDeviceSubnetMask
  - quickSpinTLInterface, 459
- indexedColor\_8bit
  - spinBMPOption, 469
- Interface Access, 174
  - spinInterfaceGetCameras, 175
  - spinInterfaceGetCamerasEx, 175
  - spinInterfaceGetTLNodeMap, 176
  - spinInterfaceIsInUse, 176
  - spinInterfaceRegisterDeviceArrivalEventHandler, 177
  - spinInterfaceRegisterDeviceRemovalEvent↔Handler, 177
  - spinInterfaceRegisterInterfaceEventHandler, 178
  - spinInterfaceRelease, 178
  - spinInterfaceSendActionCommand, 179
  - spinInterfaceUnregisterDeviceArrivalEventHandler, 179
  - spinInterfaceUnregisterDeviceRemovalEvent↔Handler, 180
  - spinInterfaceUnregisterInterfaceEventHandler, 180
  - spinInterfaceUpdateCameras, 181
- InterfaceDisplayName
  - quickSpinTLInterface, 459
  - quickSpinTLSystem, 466
- InterfaceID
  - quickSpinTLInterface, 460
  - quickSpinTLSystem, 467
- InterfaceList Access, 164
  - spinInterfaceListClear, 164
  - spinInterfaceListCreateEmpty, 165
  - spinInterfaceListDestroy, 165
  - spinInterfaceListGet, 166
  - spinInterfaceListGetSize, 166
- InterfaceSelector
  - quickSpinTLSystem, 467
- InterfaceType
  - quickSpinTLInterface, 460
- InterfaceUpdateList
  - quickSpinTLSystem, 467
- interlaced
  - spinPNGOption, 483
- IspEnable
  - quickSpin, 425
- LUTEnable
  - quickSpin, 428
- LUTIndex
  - quickSpin, 428
- LUTSelector
  - quickSpin, 428
- LUTValue
  - quickSpin, 428
- LUTValueAll
  - quickSpin, 428
- LineFilterWidth
  - quickSpin, 425
- LineFormat
  - quickSpin, 425
- LineInputFilterSelector
  - quickSpin, 426
- LineInverter
  - quickSpin, 426
- LineMode
  - quickSpin, 426
- LinePitch
  - quickSpin, 426
- LineSelector
  - quickSpin, 426
- LineSource
  - quickSpin, 426
- LineStatus
  - quickSpin, 426
- LineStatusAll
  - quickSpin, 426
- LinkErrorCount
  - quickSpin, 427
- LinkUptime
  - quickSpin, 427
- Logging Event Data Access, 240
  - spinLogDataGetCategoryName, 240
  - spinLogDataGetLogMessage, 241
  - spinLogDataGetNDC, 241
  - spinLogDataGetPriority, 242
  - spinLogDataGetPriorityName, 242
  - spinLogDataGetThreadName, 243
  - spinLogDataGetTimestamp, 243
- LogicBlockLUTInputActivation
  - quickSpin, 427
- LogicBlockLUTInputSelector
  - quickSpin, 427
- LogicBlockLUTInputSource
  - quickSpin, 427
- LogicBlockLUTOutputValue
  - quickSpin, 427
- LogicBlockLUTOutputValueAll

- quickSpin, [427](#)
- LogicBlockLUTRowIndex
  - quickSpin, [427](#)
- LogicBlockLUTSelector
  - quickSpin, [428](#)
- LogicBlockSelector
  - quickSpin, [428](#)
- m\_blackLevel
  - spinChunkData, [471](#)
- m\_cRC
  - spinChunkData, [471](#)
- m\_compressionMode
  - spinChunkData, [471](#)
- m\_compressionRatio
  - spinChunkData, [471](#)
- m\_counterValue
  - spinChunkData, [471](#)
- m\_encoderValue
  - spinChunkData, [472](#)
- m\_exposureEndLineStatusAll
  - spinChunkData, [472](#)
- m\_exposureTime
  - spinChunkData, [472](#)
- m\_frameID
  - spinChunkData, [472](#)
- m\_gain
  - spinChunkData, [472](#)
- m\_height
  - spinChunkData, [472](#)
- m\_image
  - spinChunkData, [472](#)
- m\_inferenceConfidence
  - spinChunkData, [472](#)
- m\_inferenceFrameID
  - spinChunkData, [473](#)
- m\_inferenceResult
  - spinChunkData, [473](#)
- m\_linePitch
  - spinChunkData, [473](#)
- m\_lineStatusAll
  - spinChunkData, [473](#)
- m\_offsetX
  - spinChunkData, [473](#)
- m\_offsetY
  - spinChunkData, [473](#)
- m\_partSelector
  - spinChunkData, [473](#)
- m\_pixelDynamicRangeMax
  - spinChunkData, [473](#)
- m\_pixelDynamicRangeMin
  - spinChunkData, [474](#)
- m\_scan3dAxisMax
  - spinChunkData, [474](#)
- m\_scan3dAxisMin
  - spinChunkData, [474](#)
- m\_scan3dCoordinateOffset
  - spinChunkData, [474](#)
- m\_scan3dCoordinateReferenceValue
  - spinChunkData, [474](#)
- m\_scan3dCoordinateScale
  - spinChunkData, [474](#)
- m\_scan3dInvalidDataValue
  - spinChunkData, [474](#)
- m\_scan3dTransformValue
  - spinChunkData, [474](#)
- m\_scanLineSelector
  - spinChunkData, [475](#)
- m\_sequencerSetActive
  - spinChunkData, [475](#)
- m\_serialDataLength
  - spinChunkData, [475](#)
- m\_streamChannelID
  - spinChunkData, [475](#)
- m\_timerValue
  - spinChunkData, [475](#)
- m\_timestamp
  - spinChunkData, [475](#)
- m\_timestampLatchValue
  - spinChunkData, [475](#)
- m\_transferBlockID
  - spinChunkData, [475](#)
- m\_transferQueueCurrentBlockCount
  - spinChunkData, [476](#)
- m\_width
  - spinChunkData, [476](#)
- major
  - spinLibraryVersion, [480](#)
- MaxDeviceResetTime
  - quickSpin, [428](#)
- minor
  - spinLibraryVersion, [480](#)
- Node Access, [272](#)
  - spinNodeDeregisterCallback, [273](#)
  - spinNodeGetAccessMode, [273](#)
  - spinNodeGetCachingMode, [274](#)
  - spinNodeGetDescription, [274](#)
  - spinNodeGetDisplayName, [275](#)
  - spinNodeGetImposedAccessMode, [276](#)
  - spinNodeGetImposedVisibility, [276](#)
  - spinNodeGetName, [276](#)
  - spinNodeGetNameSpace, [277](#)
  - spinNodeGetPollingTime, [277](#)
  - spinNodeGetToolTip, [278](#)
  - spinNodeGetType, [278](#)
  - spinNodeGetVisibility, [279](#)
  - spinNodeInvalidateNode, [279](#)
  - spinNodesAvailable, [280](#)
  - spinNodesEqual, [280](#)
  - spinNodesImplemented, [281](#)
  - spinNodesReadable, [281](#)
  - spinNodesWritable, [282](#)
  - spinNodeRegisterCallback, [282](#)
- Node Map Access, [268](#)
  - spinNodeMapGetNode, [268](#)
  - spinNodeMapGetNodeByIndex, [269](#)
  - spinNodeMapGetNumNodes, [269](#)

- spinNodeMapPoll, [270](#)
- spinNodeMapReleaseNode, [270](#)
- OffsetX
  - quickSpin, [429](#)
- OffsetY
  - quickSpin, [429](#)
- POEStatus
  - quickSpinTLInterface, [460](#)
- PacketResendRequestCount
  - quickSpin, [429](#)
- PayloadSize
  - quickSpin, [429](#)
- PixelColorFilter
  - quickSpin, [429](#)
- PixelDynamicRangeMax
  - quickSpin, [429](#)
- PixelDynamicRangeMin
  - quickSpin, [429](#)
- PixelFormat
  - quickSpin, [429](#)
- PixelFormatInfold
  - quickSpin, [430](#)
- PixelFormatInfoSelector
  - quickSpin, [430](#)
- PixelSize
  - quickSpin, [430](#)
- PowerSupplyCurrent
  - quickSpin, [430](#)
- PowerSupplyVoltage
  - quickSpin, [430](#)
- progressive
  - spinJPEGOption, [478](#)
- quality
  - spinJPEGOption, [478](#)
  - spinJPG2Option, [479](#)
  - spinMJPEGOption, [481](#)
- quickSpin, [350](#)
  - aPAUSEMACCtrlFramesReceived, [365](#)
  - aPAUSEMACCtrlFramesTransmitted, [365](#)
  - AasRoiEnable, [362](#)
  - AasRoiHeight, [362](#)
  - AasRoiOffsetX, [362](#)
  - AasRoiOffsetY, [362](#)
  - AasRoiWidth, [362](#)
  - AcquisitionAbort, [363](#)
  - AcquisitionArm, [363](#)
  - AcquisitionBurstFrameCount, [363](#)
  - AcquisitionFrameCount, [363](#)
  - AcquisitionFrameRate, [363](#)
  - AcquisitionFrameRateEnable, [363](#)
  - AcquisitionLineRate, [363](#)
  - AcquisitionMode, [363](#)
  - AcquisitionResultingFrameRate, [364](#)
  - AcquisitionStart, [364](#)
  - AcquisitionStatus, [364](#)
  - AcquisitionStatusSelector, [364](#)
  - AcquisitionStop, [364](#)
  - ActionDeviceKey, [364](#)
  - ActionGroupKey, [364](#)
  - ActionGroupMask, [364](#)
  - ActionQueueSize, [365](#)
  - ActionSelector, [365](#)
  - ActionUnconditionalMode, [365](#)
  - AdaptiveCompressionEnable, [365](#)
  - AdcBitDepth, [365](#)
  - AutoAlgorithmSelector, [365](#)
  - AutoExposureControlLoopDamping, [366](#)
  - AutoExposureControlPriority, [366](#)
  - AutoExposureEVCompensation, [366](#)
  - AutoExposureExposureTimeLowerLimit, [366](#)
  - AutoExposureExposureTimeUpperLimit, [366](#)
  - AutoExposureGainLowerLimit, [366](#)
  - AutoExposureGainUpperLimit, [366](#)
  - AutoExposureGreyValueLowerLimit, [366](#)
  - AutoExposureGreyValueUpperLimit, [367](#)
  - AutoExposureLightingMode, [367](#)
  - AutoExposureMeteringMode, [367](#)
  - AutoExposureTargetGreyValue, [367](#)
  - AutoExposureTargetGreyValueAuto, [367](#)
  - BalanceRatio, [367](#)
  - BalanceRatioSelector, [367](#)
  - BalanceWhiteAuto, [367](#)
  - BalanceWhiteAutoDamping, [368](#)
  - BalanceWhiteAutoLowerLimit, [368](#)
  - BalanceWhiteAutoProfile, [368](#)
  - BalanceWhiteAutoUpperLimit, [368](#)
  - BinningHorizontal, [368](#)
  - BinningHorizontalMode, [368](#)
  - BinningSelector, [368](#)
  - BinningVertical, [368](#)
  - BinningVerticalMode, [369](#)
  - BlackLevel, [369](#)
  - BlackLevelAuto, [369](#)
  - BlackLevelAutoBalance, [369](#)
  - BlackLevelClampingEnable, [369](#)
  - BlackLevelRaw, [369](#)
  - BlackLevelSelector, [369](#)
  - ChunkBlackLevel, [369](#)
  - ChunkBlackLevelSelector, [370](#)
  - ChunkCRC, [370](#)
  - ChunkCompressionMode, [370](#)
  - ChunkCompressionRatio, [370](#)
  - ChunkCounterSelector, [370](#)
  - ChunkCounterValue, [370](#)
  - ChunkEnable, [370](#)
  - ChunkEncoderSelector, [370](#)
  - ChunkEncoderStatus, [371](#)
  - ChunkEncoderValue, [371](#)
  - ChunkExposureEndLineStatusAll, [371](#)
  - ChunkExposureTime, [371](#)
  - ChunkExposureTimeSelector, [371](#)
  - ChunkFrameID, [371](#)
  - ChunkGain, [371](#)
  - ChunkGainSelector, [371](#)

- ChunkHeight, [372](#)
- ChunkImage, [372](#)
- ChunkImageComponent, [372](#)
- ChunkInferenceBoundingBoxResult, [372](#)
- ChunkInferenceConfidence, [372](#)
- ChunkInferenceFrameId, [372](#)
- ChunkInferenceResult, [372](#)
- ChunkLinePitch, [372](#)
- ChunkLineStatusAll, [373](#)
- ChunkModeActive, [373](#)
- ChunkOffsetX, [373](#)
- ChunkOffsetY, [373](#)
- ChunkPartSelector, [373](#)
- ChunkPixelDynamicRangeMax, [373](#)
- ChunkPixelDynamicRangeMin, [373](#)
- ChunkPixelFormat, [373](#)
- ChunkRegionID, [374](#)
- ChunkScan3dAxisMax, [374](#)
- ChunkScan3dAxisMin, [374](#)
- ChunkScan3dCoordinateOffset, [374](#)
- ChunkScan3dCoordinateReferenceSelector, [374](#)
- ChunkScan3dCoordinateReferenceValue, [374](#)
- ChunkScan3dCoordinateScale, [374](#)
- ChunkScan3dCoordinateSelector, [374](#)
- ChunkScan3dCoordinateSystem, [375](#)
- ChunkScan3dCoordinateSystemReference, [375](#)
- ChunkScan3dCoordinateTransformSelector, [375](#)
- ChunkScan3dDistanceUnit, [375](#)
- ChunkScan3dInvalidDataFlag, [375](#)
- ChunkScan3dInvalidDataValue, [375](#)
- ChunkScan3dOutputMode, [375](#)
- ChunkScan3dTransformValue, [375](#)
- ChunkScanLineSelector, [376](#)
- ChunkSelector, [376](#)
- ChunkSequencerSetActive, [376](#)
- ChunkSerialData, [376](#)
- ChunkSerialDataLength, [376](#)
- ChunkSerialReceiveOverflow, [376](#)
- ChunkSourceID, [376](#)
- ChunkStreamChannelID, [376](#)
- ChunkTimerSelector, [377](#)
- ChunkTimerValue, [377](#)
- ChunkTimestamp, [377](#)
- ChunkTimestampLatchValue, [377](#)
- ChunkTransferBlockID, [377](#)
- ChunkTransferQueueCurrentBlockCount, [377](#)
- ChunkTransferStreamID, [377](#)
- ChunkWidth, [377](#)
- CIConfiguration, [378](#)
- CITimeSlotsCount, [378](#)
- ColorTransformationEnable, [378](#)
- ColorTransformationSelector, [378](#)
- ColorTransformationValue, [378](#)
- ColorTransformationValueSelector, [378](#)
- CompressionRatio, [378](#)
- CompressionSaturationPriority, [378](#)
- CounterDelay, [379](#)
- CounterDuration, [379](#)
- CounterEventActivation, [379](#)
- CounterEventSource, [379](#)
- CounterReset, [379](#)
- CounterResetActivation, [379](#)
- CounterResetSource, [379](#)
- CounterSelector, [379](#)
- CounterStatus, [380](#)
- CounterTriggerActivation, [380](#)
- CounterTriggerSource, [380](#)
- CounterValue, [380](#)
- CounterValueAtReset, [380](#)
- CxpConnectionSelector, [380](#)
- CxpConnectionTestErrorCount, [380](#)
- CxpConnectionTestMode, [380](#)
- CxpConnectionTestPacketCount, [381](#)
- CxpLinkConfiguration, [381](#)
- CxpLinkConfigurationPreferred, [381](#)
- CxpLinkConfigurationStatus, [381](#)
- CxpPoCxpAuto, [381](#)
- CxpPoCxpStatus, [381](#)
- CxpPoCxpTripReset, [381](#)
- CxpPoCxpTurnOff, [381](#)
- DecimationHorizontal, [382](#)
- DecimationHorizontalMode, [382](#)
- DecimationSelector, [382](#)
- DecimationVertical, [382](#)
- DecimationVerticalMode, [382](#)
- DefectCorrectStaticEnable, [382](#)
- DefectCorrectionMode, [382](#)
- DefectTableApply, [382](#)
- DefectTableCoordinateX, [383](#)
- DefectTableCoordinateY, [383](#)
- DefectTableFactoryRestore, [383](#)
- DefectTableIndex, [383](#)
- DefectTablePixelCount, [383](#)
- DefectTableSave, [383](#)
- Deinterlacing, [383](#)
- DeviceCharacterSet, [383](#)
- DeviceClockFrequency, [384](#)
- DeviceClockSelector, [384](#)
- DeviceConnectionSelector, [384](#)
- DeviceConnectionSpeed, [384](#)
- DeviceConnectionStatus, [384](#)
- DeviceEventChannelCount, [384](#)
- DeviceFamilyName, [384](#)
- DeviceFeaturePersistenceEnd, [384](#)
- DeviceFeaturePersistenceStart, [385](#)
- DeviceFirmwareVersion, [385](#)
- DeviceGenCPVersionMajor, [385](#)
- DeviceGenCPVersionMinor, [385](#)
- DeviceID, [385](#)
- DeviceIndicatorMode, [385](#)
- DeviceLinkBandwidthReserve, [385](#)
- DeviceLinkCommandTimeout, [385](#)
- DeviceLinkConnectionCount, [386](#)
- DeviceLinkCurrentThroughput, [386](#)
- DeviceLinkHeartbeatMode, [386](#)
- DeviceLinkHeartbeatTimeout, [386](#)

- DeviceLinkSelector, [386](#)
- DeviceLinkSpeed, [386](#)
- DeviceLinkThroughputLimit, [386](#)
- DeviceLinkThroughputLimitMode, [386](#)
- DeviceManifestEntrySelector, [387](#)
- DeviceManifestPrimaryURL, [387](#)
- DeviceManifestSchemaMajorVersion, [387](#)
- DeviceManifestSchemaMinorVersion, [387](#)
- DeviceManifestSecondaryURL, [387](#)
- DeviceManifestXMLMajorVersion, [387](#)
- DeviceManifestXMLMinorVersion, [387](#)
- DeviceManifestXMLSubMinorVersion, [387](#)
- DeviceManufacturerInfo, [388](#)
- DeviceMaxThroughput, [388](#)
- DeviceModelName, [388](#)
- DevicePowerSupplySelector, [388](#)
- DeviceRegistersCheck, [388](#)
- DeviceRegistersEndianness, [388](#)
- DeviceRegistersStreamingEnd, [388](#)
- DeviceRegistersStreamingStart, [388](#)
- DeviceRegistersValid, [389](#)
- DeviceReset, [389](#)
- DeviceSFNCVersionMajor, [389](#)
- DeviceSFNCVersionMinor, [389](#)
- DeviceSFNCVersionSubMinor, [390](#)
- DeviceScanType, [389](#)
- DeviceSerialNumber, [389](#)
- DeviceSerialPortBaudRate, [389](#)
- DeviceSerialPortSelector, [389](#)
- DeviceStreamChannelCount, [390](#)
- DeviceStreamChannelEndianness, [390](#)
- DeviceStreamChannelLink, [390](#)
- DeviceStreamChannelPacketSize, [390](#)
- DeviceStreamChannelSelector, [390](#)
- DeviceStreamChannelType, [390](#)
- DeviceTLType, [391](#)
- DeviceTLVersionMajor, [391](#)
- DeviceTLVersionMinor, [391](#)
- DeviceTLVersionSubMinor, [391](#)
- DeviceTapGeometry, [390](#)
- DeviceTemperature, [391](#)
- DeviceTemperatureSelector, [391](#)
- DeviceType, [391](#)
- DeviceUptime, [391](#)
- DeviceUserID, [392](#)
- DeviceVendorName, [392](#)
- DeviceVersion, [392](#)
- EncoderDivider, [392](#)
- EncoderMode, [392](#)
- EncoderOutputMode, [392](#)
- EncoderReset, [392](#)
- EncoderResetActivation, [392](#)
- EncoderResetSource, [393](#)
- EncoderSelector, [393](#)
- EncoderSourceA, [393](#)
- EncoderSourceB, [393](#)
- EncoderStatus, [393](#)
- EncoderTimeout, [393](#)
- EncoderValue, [393](#)
- EncoderValueAtReset, [393](#)
- EnumerationCount, [394](#)
- EventAcquisitionEnd, [394](#)
- EventAcquisitionEndFrameID, [394](#)
- EventAcquisitionEndTimestamp, [394](#)
- EventAcquisitionError, [394](#)
- EventAcquisitionErrorFrameID, [394](#)
- EventAcquisitionErrorTimestamp, [394](#)
- EventAcquisitionStart, [394](#)
- EventAcquisitionStartFrameID, [395](#)
- EventAcquisitionStartTimestamp, [395](#)
- EventAcquisitionTransferEnd, [395](#)
- EventAcquisitionTransferEndFrameID, [395](#)
- EventAcquisitionTransferEndTimestamp, [395](#)
- EventAcquisitionTransferStart, [395](#)
- EventAcquisitionTransferStartFrameID, [395](#)
- EventAcquisitionTransferStartTimestamp, [395](#)
- EventAcquisitionTrigger, [396](#)
- EventAcquisitionTriggerFrameID, [396](#)
- EventAcquisitionTriggerTimestamp, [396](#)
- EventActionLate, [396](#)
- EventActionLateFrameID, [396](#)
- EventActionLateTimestamp, [396](#)
- EventCounter0End, [396](#)
- EventCounter0EndFrameID, [396](#)
- EventCounter0EndTimestamp, [397](#)
- EventCounter0Start, [397](#)
- EventCounter0StartFrameID, [397](#)
- EventCounter0StartTimestamp, [397](#)
- EventCounter1End, [397](#)
- EventCounter1EndFrameID, [397](#)
- EventCounter1EndTimestamp, [397](#)
- EventCounter1Start, [397](#)
- EventCounter1StartFrameID, [398](#)
- EventCounter1StartTimestamp, [398](#)
- EventEncoder0Restarted, [398](#)
- EventEncoder0RestartedFrameID, [398](#)
- EventEncoder0RestartedTimestamp, [398](#)
- EventEncoder0Stopped, [398](#)
- EventEncoder0StoppedFrameID, [398](#)
- EventEncoder0StoppedTimestamp, [398](#)
- EventEncoder1Restarted, [399](#)
- EventEncoder1RestartedFrameID, [399](#)
- EventEncoder1RestartedTimestamp, [399](#)
- EventEncoder1Stopped, [399](#)
- EventEncoder1StoppedFrameID, [399](#)
- EventEncoder1StoppedTimestamp, [399](#)
- EventError, [399](#)
- EventErrorCode, [399](#)
- EventErrorFrameID, [400](#)
- EventErrorTimestamp, [400](#)
- EventExposureEnd, [400](#)
- EventExposureEndFrameID, [400](#)
- EventExposureEndTimestamp, [400](#)
- EventExposureStart, [400](#)
- EventExposureStartFrameID, [400](#)
- EventExposureStartTimestamp, [400](#)

- EventFrameBurstEnd, [401](#)
- EventFrameBurstEndFrameID, [401](#)
- EventFrameBurstEndTimestamp, [401](#)
- EventFrameBurstStart, [401](#)
- EventFrameBurstStartFrameID, [401](#)
- EventFrameBurstStartTimestamp, [401](#)
- EventFrameEnd, [401](#)
- EventFrameEndFrameID, [401](#)
- EventFrameEndTimestamp, [402](#)
- EventFrameStart, [402](#)
- EventFrameStartFrameID, [402](#)
- EventFrameStartTimestamp, [402](#)
- EventFrameTransferEnd, [402](#)
- EventFrameTransferEndFrameID, [402](#)
- EventFrameTransferEndTimestamp, [402](#)
- EventFrameTransferStart, [402](#)
- EventFrameTransferStartFrameID, [403](#)
- EventFrameTransferStartTimestamp, [403](#)
- EventFrameTrigger, [403](#)
- EventFrameTriggerFrameID, [403](#)
- EventFrameTriggerTimestamp, [403](#)
- EventLine0AnyEdge, [403](#)
- EventLine0AnyEdgeFrameID, [403](#)
- EventLine0AnyEdgeTimestamp, [403](#)
- EventLine0FallingEdge, [404](#)
- EventLine0FallingEdgeFrameID, [404](#)
- EventLine0FallingEdgeTimestamp, [404](#)
- EventLine0RisingEdge, [404](#)
- EventLine0RisingEdgeFrameID, [404](#)
- EventLine0RisingEdgeTimestamp, [404](#)
- EventLine1AnyEdge, [404](#)
- EventLine1AnyEdgeFrameID, [404](#)
- EventLine1AnyEdgeTimestamp, [405](#)
- EventLine1FallingEdge, [405](#)
- EventLine1FallingEdgeFrameID, [405](#)
- EventLine1FallingEdgeTimestamp, [405](#)
- EventLine1RisingEdge, [405](#)
- EventLine1RisingEdgeFrameID, [405](#)
- EventLine1RisingEdgeTimestamp, [405](#)
- EventLinkSpeedChange, [405](#)
- EventLinkSpeedChangeFrameID, [406](#)
- EventLinkSpeedChangeTimestamp, [406](#)
- EventLinkTrigger0, [406](#)
- EventLinkTrigger0FrameID, [406](#)
- EventLinkTrigger0Timestamp, [406](#)
- EventLinkTrigger1, [406](#)
- EventLinkTrigger1FrameID, [406](#)
- EventLinkTrigger1Timestamp, [406](#)
- EventNotification, [407](#)
- EventSelector, [407](#)
- EventSequencerSetChange, [407](#)
- EventSequencerSetChangeFrameID, [407](#)
- EventSequencerSetChangeTimestamp, [407](#)
- EventSerialData, [407](#)
- EventSerialDataLength, [407](#)
- EventSerialPortReceive, [407](#)
- EventSerialPortReceiveTimestamp, [408](#)
- EventSerialReceiveOverflow, [408](#)
- EventStream0TransferBlockEnd, [408](#)
- EventStream0TransferBlockEndFrameID, [408](#)
- EventStream0TransferBlockEndTimestamp, [408](#)
- EventStream0TransferBlockStart, [408](#)
- EventStream0TransferBlockStartFrameID, [408](#)
- EventStream0TransferBlockStartTimestamp, [408](#)
- EventStream0TransferBlockTrigger, [409](#)
- EventStream0TransferBlockTriggerFrameID, [409](#)
- EventStream0TransferBlockTriggerTimestamp, [409](#)
- EventStream0TransferBurstEnd, [409](#)
- EventStream0TransferBurstEndFrameID, [409](#)
- EventStream0TransferBurstEndTimestamp, [409](#)
- EventStream0TransferBurstStart, [409](#)
- EventStream0TransferBurstStartFrameID, [409](#)
- EventStream0TransferBurstStartTimestamp, [410](#)
- EventStream0TransferEnd, [410](#)
- EventStream0TransferEndFrameID, [410](#)
- EventStream0TransferEndTimestamp, [410](#)
- EventStream0TransferOverflow, [410](#)
- EventStream0TransferOverflowFrameID, [410](#)
- EventStream0TransferOverflowTimestamp, [410](#)
- EventStream0TransferPause, [410](#)
- EventStream0TransferPauseFrameID, [411](#)
- EventStream0TransferPauseTimestamp, [411](#)
- EventStream0TransferResume, [411](#)
- EventStream0TransferResumeFrameID, [411](#)
- EventStream0TransferResumeTimestamp, [411](#)
- EventStream0TransferStart, [411](#)
- EventStream0TransferStartFrameID, [411](#)
- EventStream0TransferStartTimestamp, [411](#)
- EventTest, [412](#)
- EventTestTimestamp, [412](#)
- EventTimer0End, [412](#)
- EventTimer0EndFrameID, [412](#)
- EventTimer0EndTimestamp, [412](#)
- EventTimer0Start, [412](#)
- EventTimer0StartFrameID, [412](#)
- EventTimer0StartTimestamp, [412](#)
- EventTimer1End, [413](#)
- EventTimer1EndFrameID, [413](#)
- EventTimer1EndTimestamp, [413](#)
- EventTimer1Start, [413](#)
- EventTimer1StartFrameID, [413](#)
- EventTimer1StartTimestamp, [413](#)
- ExposureActiveMode, [413](#)
- ExposureAuto, [413](#)
- ExposureMode, [414](#)
- ExposureTime, [414](#)
- ExposureTimeMode, [414](#)
- ExposureTimeSelector, [414](#)
- FactoryReset, [414](#)
- FileAccessBuffer, [414](#)
- FileAccessLength, [414](#)
- FileAccessOffset, [414](#)
- FileOpenMode, [415](#)
- FileOperationExecute, [415](#)
- FileOperationResult, [415](#)
- FileOperationSelector, [415](#)

FileOperationStatus, 415  
FileSelector, 415  
FileSize, 415  
Gain, 415  
GainAuto, 416  
GainAutoBalance, 416  
GainSelector, 416  
Gamma, 416  
GammaEnable, 416  
GevActiveLinkCount, 416  
GevCCP, 416  
GevCurrentDefaultGateway, 416  
GevCurrentIPAddress, 417  
GevCurrentIPConfigurationDHCP, 417  
GevCurrentIPConfigurationLLA, 417  
GevCurrentIPConfigurationPersistentIP, 417  
GevCurrentPhysicalLinkConfiguration, 417  
GevCurrentSubnetMask, 417  
GevDiscoveryAckDelay, 417  
GevFirstURL, 417  
GevGVCPExtendedStatusCodes, 418  
GevGVCPExtendedStatusCodesSelector, 418  
GevGVCPHeartbeatDisable, 418  
GevGVCPPendingAck, 418  
GevGVCPPendingTimeout, 418  
GevGVSPExtendedIDMode, 418  
GevHeartbeatTimeout, 418  
GevIEEE1588, 418  
GevIEEE1588ClockAccuracy, 419  
GevIEEE1588Mode, 419  
GevIEEE1588Status, 419  
GevIPConfigurationStatus, 419  
GevInterfaceSelector, 419  
GevMACAddress, 419  
GevMCDA, 419  
GevMCPHostPort, 419  
GevMCRC, 420  
GevMCSP, 420  
GevMCTT, 420  
GevNumberOfInterfaces, 420  
GevPAUSEFrameReception, 420  
GevPAUSEFrameTransmission, 420  
GevPersistentDefaultGateway, 420  
GevPersistentIPAddress, 420  
GevPersistentSubnetMask, 421  
GevPhysicalLinkConfiguration, 421  
GevPrimaryApplicationIPAddress, 421  
GevPrimaryApplicationSocket, 421  
GevPrimaryApplicationSwitchoverKey, 421  
GevSCCFGAllInTransmission, 421  
GevSCCFGExtendedChunkData, 421  
GevSCCFGPacketResendDestination, 421  
GevSCCFGUnconditionalStreaming, 422  
GevSCDA, 422  
GevSCPDDirection, 422  
GevSCPHostPort, 422  
GevSCPIInterfaceIndex, 422  
GevSCPSBigEndian, 422  
GevSCPSDoNotFragment, 422  
GevSCPSFireTestPacket, 423  
GevSCPSPacketSize, 423  
GevSCPD, 422  
GevSCSP, 423  
GevSCZoneConfigurationLock, 423  
GevSCZoneCount, 423  
GevSCZoneDirectionAll, 423  
GevSecondURL, 423  
GevStreamChannelSelector, 423  
GevSupportedOption, 424  
GevSupportedOptionSelector, 424  
GevTimestampTickFrequency, 424  
GuiXmlManifestAddress, 424  
Height, 424  
HeightMax, 424  
ImageComponentEnable, 424  
ImageComponentSelector, 424  
ImageCompressionBitrate, 425  
ImageCompressionJPEGFormatOption, 425  
ImageCompressionMode, 425  
ImageCompressionQuality, 425  
ImageCompressionRateOption, 425  
IspEnable, 425  
LUTEnable, 428  
LUTIndex, 428  
LUTSelector, 428  
LUTValue, 428  
LUTValueAll, 428  
LineFilterWidth, 425  
LineFormat, 425  
LineInputFilterSelector, 426  
LineInverter, 426  
LineMode, 426  
LinePitch, 426  
LineSelector, 426  
LineSource, 426  
LineStatus, 426  
LineStatusAll, 426  
LinkErrorCount, 427  
LinkUptime, 427  
LogicBlockLUTInputActivation, 427  
LogicBlockLUTInputSelector, 427  
LogicBlockLUTInputSource, 427  
LogicBlockLUTOutputValue, 427  
LogicBlockLUTOutputValueAll, 427  
LogicBlockLUTRowIndex, 427  
LogicBlockLUTSelector, 428  
LogicBlockSelector, 428  
MaxDeviceResetTime, 428  
OffsetX, 429  
OffsetY, 429  
PacketResendRequestCount, 429  
PayloadSize, 429  
PixelColorFilter, 429  
PixelDynamicRangeMax, 429  
PixelDynamicRangeMin, 429  
PixelFormat, 429

- PixelFormatInfoID, [430](#)
- PixelFormatInfoSelector, [430](#)
- PixelSize, [430](#)
- PowerSupplyCurrent, [430](#)
- PowerSupplyVoltage, [430](#)
- RegionDestination, [430](#)
- RegionMode, [430](#)
- RegionSelector, [430](#)
- ReverseX, [431](#)
- ReverseY, [431](#)
- RgbTransformLightSource, [431](#)
- Saturation, [431](#)
- SaturationEnable, [431](#)
- Scan3dAxisMax, [431](#)
- Scan3dAxisMin, [431](#)
- Scan3dCoordinateOffset, [431](#)
- Scan3dCoordinateReferenceSelector, [432](#)
- Scan3dCoordinateReferenceValue, [432](#)
- Scan3dCoordinateScale, [432](#)
- Scan3dCoordinateSelector, [432](#)
- Scan3dCoordinateSystem, [432](#)
- Scan3dCoordinateSystemReference, [432](#)
- Scan3dCoordinateTransformSelector, [432](#)
- Scan3dDistanceUnit, [432](#)
- Scan3dInvalidDataFlag, [433](#)
- Scan3dInvalidDataValue, [433](#)
- Scan3dOutputMode, [433](#)
- Scan3dTransformValue, [433](#)
- SensorDescription, [433](#)
- SensorDigitizationTaps, [433](#)
- SensorHeight, [433](#)
- SensorShutterMode, [433](#)
- SensorTaps, [434](#)
- SensorWidth, [434](#)
- SequencerConfigurationMode, [434](#)
- SequencerConfigurationValid, [434](#)
- SequencerFeatureEnable, [434](#)
- SequencerMode, [434](#)
- SequencerPathSelector, [434](#)
- SequencerSetActive, [434](#)
- SequencerSetLoad, [435](#)
- SequencerSetNext, [435](#)
- SequencerSetSave, [435](#)
- SequencerSetSelector, [435](#)
- SequencerSetStart, [435](#)
- SequencerSetValid, [435](#)
- SequencerTriggerActivation, [435](#)
- SequencerTriggerSource, [435](#)
- SerialPortBaudRate, [436](#)
- SerialPortDataBits, [436](#)
- SerialPortParity, [436](#)
- SerialPortSelector, [436](#)
- SerialPortSource, [436](#)
- SerialPortStopBits, [436](#)
- SerialReceiveFramingErrorCount, [436](#)
- SerialReceiveParityErrorCount, [436](#)
- SerialReceiveQueueClear, [437](#)
- SerialReceiveQueueCurrentCharacterCount, [437](#)
- SerialReceiveQueueMaxCharacterCount, [437](#)
- SerialTransmitQueueCurrentCharacterCount, [437](#)
- SerialTransmitQueueMaxCharacterCount, [437](#)
- Sharpening, [437](#)
- SharpeningAuto, [437](#)
- SharpeningEnable, [437](#)
- SharpeningThreshold, [438](#)
- SoftwareSignalPulse, [438](#)
- SoftwareSignalSelector, [438](#)
- SourceCount, [438](#)
- SourceSelector, [438](#)
- TLParamsLocked, [440](#)
- Test0001, [438](#)
- TestEventGenerate, [438](#)
- TestPattern, [438](#)
- TestPatternGeneratorSelector, [439](#)
- TestPendingAck, [439](#)
- TimerDelay, [439](#)
- TimerDuration, [439](#)
- TimerReset, [439](#)
- TimerSelector, [439](#)
- TimerStatus, [439](#)
- TimerTriggerActivation, [439](#)
- TimerTriggerSource, [440](#)
- TimerValue, [440](#)
- Timestamp, [440](#)
- TimestampLatch, [440](#)
- TimestampLatchValue, [440](#)
- TimestampReset, [440](#)
- TransferAbort, [440](#)
- TransferBlockCount, [441](#)
- TransferBurstCount, [441](#)
- TransferComponentSelector, [441](#)
- TransferControlMode, [441](#)
- TransferOperationMode, [441](#)
- TransferPause, [441](#)
- TransferQueueCurrentBlockCount, [441](#)
- TransferQueueMaxBlockCount, [441](#)
- TransferQueueMode, [442](#)
- TransferQueueOverflowCount, [442](#)
- TransferResume, [442](#)
- TransferSelector, [442](#)
- TransferStart, [442](#)
- TransferStatus, [442](#)
- TransferStatusSelector, [442](#)
- TransferStop, [442](#)
- TransferStreamChannel, [443](#)
- TransferTriggerActivation, [443](#)
- TransferTriggerMode, [443](#)
- TransferTriggerSelector, [443](#)
- TransferTriggerSource, [443](#)
- TriggerActivation, [443](#)
- TriggerDelay, [443](#)
- TriggerDivider, [443](#)
- TriggerEventTest, [444](#)
- TriggerMode, [444](#)
- TriggerMultiplier, [444](#)
- TriggerOverlap, [444](#)

- TriggerSelector, [444](#)
- TriggerSoftware, [444](#)
- TriggerSource, [444](#)
- UserOutputSelector, [444](#)
- UserOutputValue, [445](#)
- UserOutputValueAll, [445](#)
- UserOutputValueAllMask, [445](#)
- UserSetDefault, [445](#)
- UserSetFeatureEnable, [445](#)
- UserSetLoad, [445](#)
- UserSetSave, [445](#)
- UserSetSelector, [445](#)
- V3\_3Enable, [446](#)
- WhiteClip, [446](#)
- WhiteClipSelector, [446](#)
- Width, [446](#)
- WidthMax, [446](#)
- QuickSpin Access, [141](#)
  - quickSpinInit, [141](#)
  - quickSpinInitEx, [141](#)
  - quickSpinTLDeviceInit, [142](#)
  - quickSpinTLInterfaceInit, [142](#)
  - quickSpinTLStreamInit, [142](#)
  - quickSpinTLSystemInit, [142](#)
- quickSpinBooleanNode
  - QuickSpinDefsC.h, [522](#)
- quickSpinCommandNode
  - QuickSpinDefsC.h, [522](#)
- QuickSpinDefsC.h
  - quickSpinBooleanNode, [522](#)
  - quickSpinCommandNode, [522](#)
  - quickSpinEnumerationNode, [522](#)
  - quickSpinFloatNode, [522](#)
  - quickSpinIntegerNode, [523](#)
  - quickSpinRegisterNode, [523](#)
  - quickSpinStringNode, [523](#)
- quickSpinEnumerationNode
  - QuickSpinDefsC.h, [522](#)
- quickSpinFloatNode
  - QuickSpinDefsC.h, [522](#)
- quickSpinInit
  - QuickSpin Access, [141](#)
- quickSpinInitEx
  - QuickSpin Access, [141](#)
- quickSpinIntegerNode
  - QuickSpinDefsC.h, [523](#)
- quickSpinRegisterNode
  - QuickSpinDefsC.h, [523](#)
- quickSpinStringNode
  - QuickSpinDefsC.h, [523](#)
- quickSpinTLDevice, [447](#)
  - DeviceAccessStatus, [447](#)
  - DeviceCurrentSpeed, [448](#)
  - DeviceDisplayName, [448](#)
  - DeviceDriverVersion, [448](#)
  - DeviceEndiannessMechanism, [448](#)
  - DeviceID, [448](#)
  - DeviceInstanceId, [448](#)
  - DevicesUpdater, [448](#)
  - DeviceLinkSpeed, [448](#)
  - DeviceLocation, [449](#)
  - DeviceModelName, [449](#)
  - DeviceMulticastMonitorMode, [449](#)
  - DevicePortId, [449](#)
  - DeviceSerialNumber, [449](#)
  - DeviceType, [449](#)
  - DeviceU3VProtocol, [449](#)
  - DeviceUserID, [449](#)
  - DeviceVendorName, [450](#)
  - DeviceVersion, [450](#)
  - GUIXMLLocation, [452](#)
  - GUIXMLPath, [453](#)
  - GenICamXMLLocation, [450](#)
  - GenICamXMLPath, [450](#)
  - GevCCP, [450](#)
  - GevDeviceAutoForceIP, [450](#)
  - GevDeviceDiscoverMaximumPacketSize, [450](#)
  - GevDeviceForceGateway, [450](#)
  - GevDeviceForceIPAddress, [451](#)
  - GevDeviceForceIP, [451](#)
  - GevDeviceForceSubnetMask, [451](#)
  - GevDeviceGateway, [451](#)
  - GevDeviceIPAddress, [451](#)
  - GevDevicesWrongSubnet, [451](#)
  - GevDeviceMACAddress, [451](#)
  - GevDeviceMaximumPacketSize, [451](#)
  - GevDeviceMaximumRetryCount, [452](#)
  - GevDeviceModelsBigEndian, [452](#)
  - GevDevicePort, [452](#)
  - GevDeviceReadAndWriteTimeout, [452](#)
  - GevDeviceSubnetMask, [452](#)
  - GevVersionMajor, [452](#)
  - GevVersionMinor, [452](#)
- quickSpinTLDeviceInit
  - QuickSpin Access, [142](#)
- quickSpinTLInterface, [453](#)
  - ActionCommand, [454](#)
  - DeviceAccessStatus, [454](#)
  - DeviceCount, [454](#)
  - DeviceID, [454](#)
  - DeviceModelName, [454](#)
  - DeviceSelector, [455](#)
  - DeviceSerialNumber, [455](#)
  - DeviceUnlock, [455](#)
  - DeviceUpdateList, [455](#)
  - DeviceVendorName, [455](#)
  - FilterDriverStatus, [455](#)
  - GevActionDeviceKey, [455](#)
  - GevActionGroupKey, [455](#)
  - GevActionGroupMask, [456](#)
  - GevActionTime, [456](#)
  - GevDeviceAutoForceIP, [456](#)
  - GevDeviceForceGateway, [456](#)
  - GevDeviceForceIPAddress, [456](#)
  - GevDeviceForceIP, [456](#)
  - GevDeviceForceSubnetMask, [456](#)

- GevDeviceGateway, [456](#)
- GevDeviceIPAddress, [457](#)
- GevDeviceMACAddress, [457](#)
- GevDeviceSubnetMask, [457](#)
- GevInterfaceGateway, [457](#)
- GevInterfaceGatewaySelector, [457](#)
- GevInterfaceMACAddress, [457](#)
- GevInterfaceMTU, [457](#)
- GevInterfaceReceiveLinkSpeed, [457](#)
- GevInterfaceSubnetIPAddress, [458](#)
- GevInterfaceSubnetMask, [458](#)
- GevInterfaceSubnetSelector, [458](#)
- GevInterfaceTransmitLinkSpeed, [458](#)
- HostAdapterDriverVersion, [458](#)
- HostAdapterName, [458](#)
- HostAdapterVendor, [458](#)
- IncompatibleDeviceCount, [458](#)
- IncompatibleDeviceID, [459](#)
- IncompatibleDeviceModelName, [459](#)
- IncompatibleDeviceSelector, [459](#)
- IncompatibleDeviceVendorName, [459](#)
- IncompatibleGevDeviceIPAddress, [459](#)
- IncompatibleGevDeviceMACAddress, [459](#)
- IncompatibleGevDeviceSubnetMask, [459](#)
- InterfaceDisplayName, [459](#)
- InterfaceID, [460](#)
- InterfaceType, [460](#)
- POEStatus, [460](#)
- quickSpinTLInterfaceInit
  - QuickSpin Access, [142](#)
- quickSpinTLStream, [460](#)
  - GevFailedPacketCount, [461](#)
  - GevMaximumNumberResendRequests, [461](#)
  - GevPacketResendMode, [461](#)
  - GevPacketResendTimeout, [461](#)
  - GevResendPacketCount, [461](#)
  - GevResendRequestCount, [461](#)
  - GevTotalPacketCount, [461](#)
  - StreamAnnounceBufferMinimum, [461](#)
  - StreamAnnouncedBufferCount, [462](#)
  - StreamBlockTransferSize, [462](#)
  - StreamBufferAlignment, [462](#)
  - StreamBufferCountManual, [462](#)
  - StreamBufferCountMax, [462](#)
  - StreamBufferCountMode, [462](#)
  - StreamBufferCountResult, [462](#)
  - StreamBufferHandlingMode, [462](#)
  - StreamCRCCheckEnable, [463](#)
  - StreamChunkCountMaximum, [463](#)
  - StreamDeliveredFrameCount, [463](#)
  - StreamDroppedFrameCount, [463](#)
  - StreamFailedBufferCount, [463](#)
  - StreamID, [463](#)
  - StreamInputBufferCount, [463](#)
  - StreamIsGrabbing, [463](#)
  - StreamLostFrameCount, [464](#)
  - StreamOutputBufferCount, [464](#)
  - StreamStartedFrameCount, [464](#)
  - StreamType, [464](#)
- quickSpinTLStreamInit
  - QuickSpin Access, [142](#)
- quickSpinTLSystem, [464](#)
  - EnumerateGEVInterfaces, [465](#)
  - EnumerateUSBInterfaces, [465](#)
  - GenTLFNCVersionMajor, [465](#)
  - GenTLFNCVersionMinor, [465](#)
  - GenTLFNCVersionSubMinor, [465](#)
  - GenTLVersionMajor, [465](#)
  - GenTLVersionMinor, [466](#)
  - GevInterfaceDefaultGateway, [466](#)
  - GevInterfaceDefaultIPAddress, [466](#)
  - GevInterfaceDefaultSubnetMask, [466](#)
  - GevInterfaceMACAddress, [466](#)
  - GevVersionMajor, [466](#)
  - GevVersionMinor, [466](#)
  - InterfaceDisplayName, [466](#)
  - InterfaceID, [467](#)
  - InterfaceSelector, [467](#)
  - InterfaceUpdateList, [467](#)
  - TLDisplayName, [467](#)
  - TLFileName, [467](#)
  - TLID, [467](#)
  - TLModelName, [467](#)
  - TLPath, [467](#)
  - TLType, [468](#)
  - TLVendorName, [468](#)
  - TLVersion, [468](#)
- quickSpinTLSystemInit
  - QuickSpin Access, [142](#)
- RegionDestination
  - quickSpin, [430](#)
- RegionMode
  - quickSpin, [430](#)
- RegionSelector
  - quickSpin, [430](#)
- reserved
  - spinAVIOption, [469](#)
  - spinBMPOption, [470](#)
  - spinH264Option, [477](#)
  - spinJPEGOption, [478](#)
  - spinJPG2Option, [479](#)
  - spinMJPEGOption, [481](#)
  - spinPGMOption, [482](#)
  - spinPNGOption, [483](#)
  - spinPPMOption, [484](#)
  - spinTIFFOption, [485](#)
- ReverseX
  - quickSpin, [431](#)
- ReverseY
  - quickSpin, [431](#)
- RgbTransformLightSource
  - quickSpin, [431](#)
- SPINNAKERC\_API
  - SpinnakerPlatformC.h, [545](#)
- Saturation

- quickSpin, [431](#)
- SaturationEnable
  - quickSpin, [431](#)
- Scan3dAxisMax
  - quickSpin, [431](#)
- Scan3dAxisMin
  - quickSpin, [431](#)
- Scan3dCoordinateOffset
  - quickSpin, [431](#)
- Scan3dCoordinateReferenceSelector
  - quickSpin, [432](#)
- Scan3dCoordinateReferenceValue
  - quickSpin, [432](#)
- Scan3dCoordinateScale
  - quickSpin, [432](#)
- Scan3dCoordinateSelector
  - quickSpin, [432](#)
- Scan3dCoordinateSystem
  - quickSpin, [432](#)
- Scan3dCoordinateSystemReference
  - quickSpin, [432](#)
- Scan3dCoordinateTransformSelector
  - quickSpin, [432](#)
- Scan3dDistanceUnit
  - quickSpin, [432](#)
- Scan3dInvalidDataFlag
  - quickSpin, [433](#)
- Scan3dInvalidDataValue
  - quickSpin, [433](#)
- Scan3dOutputMode
  - quickSpin, [433](#)
- Scan3dTransformValue
  - quickSpin, [433](#)
- SensorDescription
  - quickSpin, [433](#)
- SensorDigitizationTaps
  - quickSpin, [433](#)
- SensorHeight
  - quickSpin, [433](#)
- SensorShutterMode
  - quickSpin, [433](#)
- SensorTaps
  - quickSpin, [434](#)
- SensorWidth
  - quickSpin, [434](#)
- SequencerConfigurationMode
  - quickSpin, [434](#)
- SequencerConfigurationValid
  - quickSpin, [434](#)
- SequencerFeatureEnable
  - quickSpin, [434](#)
- SequencerMode
  - quickSpin, [434](#)
- SequencerPathSelector
  - quickSpin, [434](#)
- SequencerSetActive
  - quickSpin, [434](#)
- SequencerSetLoad
  - quickSpin, [435](#)
- SequencerSetNext
  - quickSpin, [435](#)
- SequencerSetSave
  - quickSpin, [435](#)
- SequencerSetSelector
  - quickSpin, [435](#)
- SequencerSetStart
  - quickSpin, [435](#)
- SequencerSetValid
  - quickSpin, [435](#)
- SequencerTriggerActivation
  - quickSpin, [435](#)
- SequencerTriggerSource
  - quickSpin, [435](#)
- SerialPortBaudRate
  - quickSpin, [436](#)
- SerialPortDataBits
  - quickSpin, [436](#)
- SerialPortParity
  - quickSpin, [436](#)
- SerialPortSelector
  - quickSpin, [436](#)
- SerialPortSource
  - quickSpin, [436](#)
- SerialPortStopBits
  - quickSpin, [436](#)
- SerialReceiveFramingErrorCount
  - quickSpin, [436](#)
- SerialReceiveParityErrorCount
  - quickSpin, [436](#)
- SerialReceiveQueueClear
  - quickSpin, [437](#)
- SerialReceiveQueueCurrentCharacterCount
  - quickSpin, [437](#)
- SerialReceiveQueueMaxCharacterCount
  - quickSpin, [437](#)
- SerialTransmitQueueCurrentCharacterCount
  - quickSpin, [437](#)
- SerialTransmitQueueMaxCharacterCount
  - quickSpin, [437](#)
- Sharpening
  - quickSpin, [437](#)
- SharpeningAuto
  - quickSpin, [437](#)
- SharpeningEnable
  - quickSpin, [437](#)
- SharpeningThreshold
  - quickSpin, [438](#)
- SoftwareSignalPulse
  - quickSpin, [438](#)
- SoftwareSignalSelector
  - quickSpin, [438](#)
- SourceCount
  - quickSpin, [438](#)
- SourceSelector
  - quickSpin, [438](#)
- spinAVIOption, [468](#)

- frameRate, [469](#)
  - reserved, [469](#)
- spinAccessMode
  - Spinnaker C GenICam Enumerations, [324](#)
- spinAcquisitionModeEnums
  - Camera Enumerations, [55](#)
- spinAcquisitionStatusSelectorEnums
  - Camera Enumerations, [55](#)
- spinActionUnconditionalModeEnums
  - Camera Enumerations, [56](#)
- spinAdcBitDepthEnums
  - Camera Enumerations, [56](#)
- spinArrivalEventFunction
  - Spinnaker C Function Signatures, [253](#)
- spinAutoAlgorithmSelectorEnums
  - Camera Enumerations, [56](#)
- spinAutoExposureControlPriorityEnums
  - Camera Enumerations, [57](#)
- spinAutoExposureLightingModeEnums
  - Camera Enumerations, [57](#)
- spinAutoExposureMeteringModeEnums
  - Camera Enumerations, [58](#)
- spinAutoExposureTargetGreyValueAutoEnums
  - Camera Enumerations, [58](#)
- spinBMPOption, [469](#)
  - indexedColor\_8bit, [469](#)
  - reserved, [470](#)
- spinBalanceRatioSelectorEnums
  - Camera Enumerations, [58](#)
- spinBalanceWhiteAutoEnums
  - Camera Enumerations, [59](#)
- spinBalanceWhiteAutoProfileEnums
  - Camera Enumerations, [59](#)
- spinBinningHorizontalModeEnums
  - Camera Enumerations, [59](#)
- spinBinningSelectorEnums
  - Camera Enumerations, [60](#)
- spinBinningVerticalModeEnums
  - Camera Enumerations, [60](#)
- spinBlackLevelAutoBalanceEnums
  - Camera Enumerations, [60](#)
- spinBlackLevelAutoEnums
  - Camera Enumerations, [61](#)
- spinBlackLevelSelectorEnums
  - Camera Enumerations, [61](#)
- spinBooleanGetValue
  - IBoolean Access, [309](#)
- spinBooleanSetValue
  - IBoolean Access, [310](#)
- spinCachingMode
  - Spinnaker C GenICam Enumerations, [325](#)
- spinCamera
  - Spinnaker C Handles, [250](#)
- spinCameraBeginAcquisition
  - Camera Access, [183](#)
- spinCameraDeInit
  - Camera Access, [184](#)
- spinCameraDiscoverMaxPacketSize
  - Spinnaker C API, [144](#)
- spinCameraEndAcquisition
  - Camera Access, [184](#)
- spinCameraForceIP
  - SpinnakerC.h, [532](#)
- spinCameraGetAccessMode
  - Camera Access, [184](#)
- spinCameraGetGuiXml
  - Camera Access, [185](#)
- spinCameraGetNextImage
  - Camera Access, [185](#)
- spinCameraGetNextImageEx
  - Camera Access, [186](#)
- spinCameraGetNodeMap
  - Camera Access, [186](#)
- spinCameraGetTLDeviceNodeMap
  - Camera Access, [187](#)
- spinCameraGetTLStreamNodeMap
  - Camera Access, [187](#)
- spinCameraGetUniqueID
  - Camera Access, [188](#)
- spinCameraInit
  - Camera Access, [188](#)
- spinCamerasInitialized
  - Camera Access, [189](#)
- spinCamerasStreaming
  - Camera Access, [189](#)
- spinCamerasValid
  - Camera Access, [190](#)
- spinCameraList
  - Spinnaker C Handles, [250](#)
- spinCameraListAppend
  - CameraList Access, [168](#)
- spinCameraListClear
  - CameraList Access, [169](#)
- spinCameraListCreateEmpty
  - CameraList Access, [169](#)
- spinCameraListDestroy
  - CameraList Access, [170](#)
- spinCameraListGet
  - CameraList Access, [170](#)
- spinCameraListGetBySerial
  - CameraList Access, [171](#)
- spinCameraListGetSize
  - CameraList Access, [171](#)
- spinCameraListRemove
  - CameraList Access, [172](#)
- spinCameraListRemoveBySerial
  - CameraList Access, [172](#)
- spinCameraReadPort
  - Camera Access, [190](#)
- spinCameraRegisterDeviceEventHandler
  - Camera Access, [190](#)
- spinCameraRegisterDeviceEventHandlerEx
  - Camera Access, [191](#)
- spinCameraRegisterImageEventHandler
  - Camera Access, [191](#)
- spinCameraRelease

- Camera Access, [192](#)
- spinCameraUnregisterDeviceEventHandler
  - Camera Access, [192](#)
- spinCameraUnregisterImageEventHandler
  - Camera Access, [193](#)
- spinCameraWritePort
  - Camera Access, [193](#)
- spinCategoryGetFeatureByIndex
  - ICategory Access, [313](#)
- spinCategoryGetNumFeatures
  - ICategory Access, [314](#)
- spinCategoryReleaseNode
  - Spinnaker C GenICam API, [266](#)
- spinChunkBlackLevelSelectorEnums
  - Camera Enumerations, [61](#)
- spinChunkCounterSelectorEnums
  - Camera Enumerations, [62](#)
- spinChunkData, [470](#)
  - m\_blackLevel, [471](#)
  - m\_cRC, [471](#)
  - m\_compressionMode, [471](#)
  - m\_compressionRatio, [471](#)
  - m\_counterValue, [471](#)
  - m\_encoderValue, [472](#)
  - m\_exposureEndLineStatusAll, [472](#)
  - m\_exposureTime, [472](#)
  - m\_frameID, [472](#)
  - m\_gain, [472](#)
  - m\_height, [472](#)
  - m\_image, [472](#)
  - m\_inferenceConfidence, [472](#)
  - m\_inferenceFrameID, [473](#)
  - m\_inferenceResult, [473](#)
  - m\_linePitch, [473](#)
  - m\_lineStatusAll, [473](#)
  - m\_offsetX, [473](#)
  - m\_offsetY, [473](#)
  - m\_partSelector, [473](#)
  - m\_pixelDynamicRangeMax, [473](#)
  - m\_pixelDynamicRangeMin, [474](#)
  - m\_scan3dAxisMax, [474](#)
  - m\_scan3dAxisMin, [474](#)
  - m\_scan3dCoordinateOffset, [474](#)
  - m\_scan3dCoordinateReferenceValue, [474](#)
  - m\_scan3dCoordinateScale, [474](#)
  - m\_scan3dInvalidDataValue, [474](#)
  - m\_scan3dTransformValue, [474](#)
  - m\_scanLineSelector, [475](#)
  - m\_sequencerSetActive, [475](#)
  - m\_serialDataLength, [475](#)
  - m\_streamChannelID, [475](#)
  - m\_timerValue, [475](#)
  - m\_timestamp, [475](#)
  - m\_timestampLatchValue, [475](#)
  - m\_transferBlockID, [475](#)
  - m\_transferQueueCurrentBlockCount, [476](#)
  - m\_width, [476](#)
- spinChunkEncoderSelectorEnums
  - Camera Enumerations, [62](#)
- spinChunkEncoderStatusEnums
  - Camera Enumerations, [62](#)
- spinChunkExposureTimeSelectorEnums
  - Camera Enumerations, [63](#)
- spinChunkGainSelectorEnums
  - Camera Enumerations, [63](#)
- spinChunkImageComponentEnums
  - Camera Enumerations, [63](#)
- spinChunkPixelFormatEnums
  - Camera Enumerations, [64](#)
- spinChunkRegionIDEnums
  - Camera Enumerations, [64](#)
- spinChunkScan3dCoordinateReferenceSelectorEnums
  - Camera Enumerations, [65](#)
- spinChunkScan3dCoordinateSelectorEnums
  - Camera Enumerations, [65](#)
- spinChunkScan3dCoordinateSystemEnums
  - Camera Enumerations, [65](#)
- spinChunkScan3dCoordinateSystemReferenceEnums
  - Camera Enumerations, [66](#)
- spinChunkScan3dCoordinateTransformSelectorEnums
  - Camera Enumerations, [66](#)
- spinChunkScan3dDistanceUnitEnums
  - Camera Enumerations, [66](#)
- spinChunkScan3dOutputModeEnums
  - Camera Enumerations, [67](#)
- spinChunkSelectorEnums
  - Camera Enumerations, [67](#)
- spinChunkSourceIDEnums
  - Camera Enumerations, [68](#)
- spinChunkTimerSelectorEnums
  - Camera Enumerations, [68](#)
- spinChunkTransferStreamIDEnums
  - Camera Enumerations, [69](#)
- spinCIConfigurationEnums
  - Camera Enumerations, [69](#)
- spinCITimeSlotsCountEnums
  - Camera Enumerations, [69](#)
- spinColorProcessingAlgorithm
  - Spinnaker C Enumerations, [257](#)
- spinColorTransformationSelectorEnums
  - Camera Enumerations, [70](#)
- spinColorTransformationValueSelectorEnums
  - Camera Enumerations, [70](#)
- spinCommandExecute
  - ICommand Access, [311](#)
- spinCommandIsDone
  - ICommand Access, [312](#)
- spinCompressionMethod
  - Spinnaker C Structures, [264](#)
- spinCompressionSaturationPriorityEnums
  - Camera Enumerations, [71](#)
- spinCounterEventActivationEnums
  - Camera Enumerations, [71](#)
- spinCounterEventSourceEnums
  - Camera Enumerations, [71](#)
- spinCounterResetActivationEnums

- Camera Enumerations, [72](#)
- spinCounterResetSourceEnums
  - Camera Enumerations, [72](#)
- spinCounterSelectorEnums
  - Camera Enumerations, [73](#)
- spinCounterStatusEnums
  - Camera Enumerations, [73](#)
- spinCounterTriggerActivationEnums
  - Camera Enumerations, [74](#)
- spinCounterTriggerSourceEnums
  - Camera Enumerations, [74](#)
- spinCxpConnectionTestModeEnums
  - Camera Enumerations, [75](#)
- spinCxpLinkConfigurationEnums
  - Camera Enumerations, [75](#)
- spinCxpLinkConfigurationPreferredEnums
  - Camera Enumerations, [76](#)
- spinCxpLinkConfigurationStatusEnums
  - Camera Enumerations, [77](#)
- spinCxpPoCxpStatusEnums
  - Camera Enumerations, [78](#)
- spinDecimationHorizontalModeEnums
  - Camera Enumerations, [78](#)
- spinDecimationSelectorEnums
  - Camera Enumerations, [78](#)
- spinDecimationVerticalModeEnums
  - Camera Enumerations, [79](#)
- spinDefectCorrectionModeEnums
  - Camera Enumerations, [79](#)
- spinDeinterlacingEnums
  - Camera Enumerations, [79](#)
- spinDeviceArrivalEventHandler
  - Spinnaker C Handles, [250](#)
- spinDeviceArrivalEventHandlerCreate
  - Event Access, [224](#)
- spinDeviceArrivalEventHandlerDestroy
  - Event Access, [225](#)
- spinDeviceCharacterSetEnums
  - Camera Enumerations, [80](#)
- spinDeviceClockSelectorEnums
  - Camera Enumerations, [80](#)
- spinDeviceConnectionStatusEnums
  - Camera Enumerations, [80](#)
- spinDeviceEventData
  - Spinnaker C Handles, [250](#)
- spinDeviceEventFunction
  - Spinnaker C Function Signatures, [253](#)
- spinDeviceEventGetId
  - Device Event Data Access, [245](#)
- spinDeviceEventGetName
  - Device Event Data Access, [246](#)
- spinDeviceEventGetPayloadData
  - Device Event Data Access, [246](#)
- spinDeviceEventGetPayloadDataSize
  - Device Event Data Access, [247](#)
- spinDeviceEventHandler
  - Spinnaker C Handles, [250](#)
- spinDeviceEventHandlerCreate
  - Event Access, [225](#)
- spinDeviceEventHandlerDestroy
  - Event Access, [226](#)
- spinDeviceIndicatorModeEnums
  - Camera Enumerations, [81](#)
- spinDeviceLinkHeartbeatModeEnums
  - Camera Enumerations, [81](#)
- spinDeviceLinkThroughputLimitModeEnums
  - Camera Enumerations, [81](#)
- spinDevicePowerSupplySelectorEnums
  - Camera Enumerations, [81](#)
- spinDeviceRegistersEndiannessEnums
  - Camera Enumerations, [82](#)
- spinDeviceRemovalEventHandler
  - Spinnaker C Handles, [251](#)
- spinDeviceRemovalEventHandlerCreate
  - Event Access, [226](#)
- spinDeviceRemovalEventHandlerDestroy
  - Event Access, [227](#)
- spinDeviceScanTypeEnums
  - Camera Enumerations, [82](#)
- spinDeviceSerialPortBaudRateEnums
  - Camera Enumerations, [82](#)
- spinDeviceSerialPortSelectorEnums
  - Camera Enumerations, [83](#)
- spinDeviceStreamChannelEndiannessEnums
  - Camera Enumerations, [83](#)
- spinDeviceStreamChannelTypeEnums
  - Camera Enumerations, [83](#)
- spinDeviceTLTypeEnums
  - Camera Enumerations, [85](#)
- spinDeviceTapGeometryEnums
  - Camera Enumerations, [84](#)
- spinDeviceTemperatureSelectorEnums
  - Camera Enumerations, [85](#)
- spinDeviceTypeEnums
  - Camera Enumerations, [86](#)
- spinDisplayNotation
  - Spinnaker C GenICam Enumerations, [325](#)
- spinEncoderModeEnums
  - Camera Enumerations, [86](#)
- spinEncoderOutputModeEnums
  - Camera Enumerations, [86](#)
- spinEncoderResetActivationEnums
  - Camera Enumerations, [87](#)
- spinEncoderResetSourceEnums
  - Camera Enumerations, [87](#)
- spinEncoderSelectorEnums
  - Camera Enumerations, [88](#)
- spinEncoderSourceAEnums
  - Camera Enumerations, [89](#)
- spinEncoderSourceBEnums
  - Camera Enumerations, [89](#)
- spinEncoderStatusEnums
  - Camera Enumerations, [89](#)
- spinEndianness
  - Spinnaker C GenICam Enumerations, [325](#)
- spinEnumerationEntryGetEnumValue

- IEnumEntry Access, [306](#)
- spinEnumerationEntryGetIntValue
  - IEnumEntry Access, [307](#)
- spinEnumerationEntryGetSymbolic
  - IEnumEntry Access, [307](#)
- spinEnumerationGetCurrentEntry
  - IEnumeration Access, [301](#)
- spinEnumerationGetEntryByIndex
  - IEnumeration Access, [302](#)
- spinEnumerationGetEntryByName
  - IEnumeration Access, [302](#)
- spinEnumerationGetNumEntries
  - IEnumeration Access, [303](#)
- spinEnumerationReleaseNode
  - IEnumeration Access, [303](#)
- spinEnumerationSetEnumValue
  - IEnumeration Access, [304](#)
- spinEnumerationSetIntValue
  - IEnumeration Access, [304](#)
- spinError
  - Spinnaker C Enumerations, [258](#)
- spinErrorGetLast
  - Error Handling, [145](#)
- spinErrorGetLastBuildDate
  - Error Handling, [146](#)
- spinErrorGetLastBuildTime
  - Error Handling, [146](#)
- spinErrorGetLastFileName
  - Error Handling, [147](#)
- spinErrorGetLastFullMessage
  - Error Handling, [147](#)
- spinErrorGetLastFunctionName
  - Error Handling, [148](#)
- spinErrorGetLastLineNumber
  - Error Handling, [148](#)
- spinErrorGetLastMessage
  - Error Handling, [149](#)
- spinEventNotificationEnums
  - Camera Enumerations, [90](#)
- spinEventSelectorEnums
  - Camera Enumerations, [90](#)
- spinExposureActiveModeEnums
  - Camera Enumerations, [90](#)
- spinExposureAutoEnums
  - Camera Enumerations, [90](#)
- spinExposureModeEnums
  - Camera Enumerations, [91](#)
- spinExposureTimeModeEnums
  - Camera Enumerations, [91](#)
- spinExposureTimeSelectorEnums
  - Camera Enumerations, [92](#)
- spinFileOpenModeEnums
  - Camera Enumerations, [92](#)
- spinFileOperationSelectorEnums
  - Camera Enumerations, [92](#)
- spinFileOperationStatusEnums
  - Camera Enumerations, [93](#)
- spinFileSelectorEnums
  - Camera Enumerations, [93](#)
- spinFloatGetMax
  - IFloat Access, [296](#)
- spinFloatGetMin
  - IFloat Access, [297](#)
- spinFloatGetRepresentation
  - IFloat Access, [297](#)
- spinFloatGetUnit
  - IFloat Access, [298](#)
- spinFloatGetValue
  - IFloat Access, [298](#)
- spinFloatGetValueEx
  - IFloat Access, [299](#)
- spinFloatSetValue
  - IFloat Access, [299](#)
- spinFloatSetValueEx
  - IFloat Access, [300](#)
- spinGainAutoBalanceEnums
  - Camera Enumerations, [93](#)
- spinGainAutoEnums
  - Camera Enumerations, [95](#)
- spinGainSelectorEnums
  - Camera Enumerations, [95](#)
- spinGevCCPEnums
  - Camera Enumerations, [95](#)
- spinGevCurrentPhysicalLinkConfigurationEnums
  - Camera Enumerations, [96](#)
- spinGevGVCPExtendedStatusCodesSelectorEnums
  - Camera Enumerations, [96](#)
- spinGevGVSPExtendedIDModeEnums
  - Camera Enumerations, [96](#)
- spinGevIEEE1588ClockAccuracyEnums
  - Camera Enumerations, [97](#)
- spinGevIEEE1588ModeEnums
  - Camera Enumerations, [97](#)
- spinGevIEEE1588StatusEnums
  - Camera Enumerations, [97](#)
- spinGevIPConfigurationStatusEnums
  - Camera Enumerations, [98](#)
- spinGevPhysicalLinkConfigurationEnums
  - Camera Enumerations, [98](#)
- spinGevSupportedOptionSelectorEnums
  - Camera Enumerations, [98](#)
- spinH264Option, [476](#)
  - bitrate, [476](#)
  - frameRate, [477](#)
  - height, [477](#)
  - reserved, [477](#)
  - width, [477](#)
- spinImage
  - Spinnaker C Handles, [251](#)
- spinImageCalculateStatistics
  - Image Access, [196](#)
- spinImageCheckCRC
  - Image Access, [197](#)
- spinImageChunkDataGetFloatValue
  - Chunk data access, [248](#)
- spinImageChunkDataGetIntValue

- Chunk data access, [248](#)
- spinImageComponentSelectorEnums
  - Camera Enumerations, [99](#)
- spinImageCompressionJPEGFormatOptionEnums
  - Camera Enumerations, [100](#)
- spinImageCompressionModeEnums
  - Camera Enumerations, [100](#)
- spinImageCompressionRateOptionEnums
  - Camera Enumerations, [101](#)
- spinImageConvert
  - Image Access, [197](#)
- spinImageConvertEx
  - Image Access, [198](#)
- spinImageCreate
  - Image Access, [198](#)
- spinImageCreateEmpty
  - Image Access, [199](#)
- spinImageCreateEx
  - Image Access, [199](#)
- spinImageCreateEx2
  - Image Access, [200](#)
- spinImageDeepCopy
  - Image Access, [201](#)
- spinImageDestroy
  - Image Access, [201](#)
- spinImageEventFunction
  - Spinnaker C Function Signatures, [253](#)
- spinImageEventHandler
  - Spinnaker C Handles, [251](#)
- spinImageEventHandlerCreate
  - Event Access, [227](#)
- spinImageEventHandlerDestroy
  - Event Access, [228](#)
- spinImageFileFormat
  - Spinnaker C Enumerations, [259](#)
- spinImageGetBitsPerPixel
  - Image Access, [201](#)
- spinImageGetBufferSize
  - Image Access, [202](#)
- spinImageGetChunkLayoutID
  - Image Access, [202](#)
- spinImageGetColorProcessing
  - Image Access, [203](#)
- spinImageGetData
  - Image Access, [203](#)
- spinImageGetDefaultColorProcessing
  - Image Access, [204](#)
- spinImageGetFrameID
  - Image Access, [204](#)
- spinImageGetHeight
  - Image Access, [205](#)
- spinImageGetID
  - Image Access, [205](#)
- spinImageGetNumDecompressionThreads
  - Image Access, [206](#)
- spinImageGetOffsetX
  - Image Access, [206](#)
- spinImageGetOffsetY
  - Image Access, [207](#)
- spinImageGetPaddingX
  - Image Access, [207](#)
- spinImageGetPaddingY
  - Image Access, [208](#)
- spinImageGetPayloadType
  - Image Access, [208](#)
- spinImageGetPixelFormat
  - Image Access, [209](#)
- spinImageGetPixelFormatName
  - Image Access, [209](#)
- spinImageGetPrivateData
  - Image Access, [210](#)
- spinImageGetSize
  - Image Access, [210](#)
- spinImageGetStatus
  - Image Access, [211](#)
- spinImageGetStatusDescription
  - Image Access, [211](#)
- spinImageGetStride
  - Image Access, [212](#)
- spinImageGetTLPayloadType
  - Image Access, [213](#)
- spinImageGetTLPixelFormat
  - Image Access, [213](#)
- spinImageGetTLPixelFormatNamespace
  - Image Access, [214](#)
- spinImageGetTimeStamp
  - Image Access, [212](#)
- spinImageGetValidPayloadSize
  - Image Access, [214](#)
- spinImageGetWidth
  - Image Access, [215](#)
- spinImageHasCRC
  - Image Access, [215](#)
- spinImageIsIncomplete
  - Image Access, [216](#)
- spinImageRelease
  - Image Access, [216](#)
- spinImageReset
  - Image Access, [216](#)
- spinImageResetEx
  - Image Access, [217](#)
- spinImageSave
  - Image Access, [218](#)
- spinImageSaveBmp
  - Image Access, [218](#)
- spinImageSaveFromExt
  - Image Access, [219](#)
- spinImageSaveJpeg
  - Image Access, [219](#)
- spinImageSaveJpg2
  - Image Access, [220](#)
- spinImageSavePgm
  - Image Access, [220](#)
- spinImageSavePng
  - Image Access, [221](#)
- spinImageSavePpm
  - Image Access, [221](#)

- Image Access, [221](#)
- spinImageSaveTiff
  - Image Access, [222](#)
- spinImageSetDefaultColorProcessing
  - Image Access, [222](#)
- spinImageSetNumDecompressionThreads
  - Image Access, [223](#)
- spinImageStatistics
  - Spinnaker C Handles, [251](#)
- spinImageStatisticsCreate
  - ImageStatistics Access, [232](#)
- spinImageStatisticsDestroy
  - ImageStatistics Access, [232](#)
- spinImageStatisticsDisableAll
  - ImageStatistics Access, [232](#)
- spinImageStatisticsEnableAll
  - ImageStatistics Access, [233](#)
- spinImageStatisticsEnableGreyOnly
  - ImageStatistics Access, [233](#)
- spinImageStatisticsEnableHslOnly
  - ImageStatistics Access, [234](#)
- spinImageStatisticsEnableRgbOnly
  - ImageStatistics Access, [234](#)
- spinImageStatisticsGetAll
  - ImageStatistics Access, [235](#)
- spinImageStatisticsGetChannelStatus
  - ImageStatistics Access, [235](#)
- spinImageStatisticsGetHistogram
  - ImageStatistics Access, [236](#)
- spinImageStatisticsGetMean
  - ImageStatistics Access, [236](#)
- spinImageStatisticsGetNumPixelValues
  - ImageStatistics Access, [237](#)
- spinImageStatisticsGetPixelValueRange
  - ImageStatistics Access, [237](#)
- spinImageStatisticsGetRange
  - ImageStatistics Access, [238](#)
- spinImageStatisticsSetChannelStatus
  - ImageStatistics Access, [238](#)
- spinImageStatus
  - Spinnaker C Enumerations, [260](#)
- spinIncMode
  - Spinnaker C GenICam Enumerations, [326](#)
- spinInputDirection
  - Spinnaker C GenICam Enumerations, [326](#)
- spinIntegerGetInc
  - Integer Access, [291](#)
- spinIntegerGetMax
  - Integer Access, [292](#)
- spinIntegerGetMin
  - Integer Access, [292](#)
- spinIntegerGetRepresentation
  - Integer Access, [293](#)
- spinIntegerGetValue
  - Integer Access, [293](#)
- spinIntegerGetValueEx
  - Integer Access, [294](#)
- spinIntegerSetValue
  - Integer Access, [294](#)
- spinIntegerSetValueEx
  - Integer Access, [295](#)
- spinInterface
  - Spinnaker C Handles, [251](#)
- spinInterfaceEventHandler
  - Spinnaker C Handles, [251](#)
- spinInterfaceEventHandlerCreate
  - Event Access, [228](#)
- spinInterfaceEventHandlerDestroy
  - Event Access, [229](#)
- spinInterfaceGetCameras
  - Interface Access, [175](#)
- spinInterfaceGetCamerasEx
  - Interface Access, [175](#)
- spinInterfaceGetTLNodeMap
  - Interface Access, [176](#)
- spinInterfaceIsInUse
  - Interface Access, [176](#)
- spinInterfaceList
  - Spinnaker C Handles, [252](#)
- spinInterfaceListClear
  - InterfaceList Access, [164](#)
- spinInterfaceListCreateEmpty
  - InterfaceList Access, [165](#)
- spinInterfaceListDestroy
  - InterfaceList Access, [165](#)
- spinInterfaceListGet
  - InterfaceList Access, [166](#)
- spinInterfaceListGetSize
  - InterfaceList Access, [166](#)
- spinInterfaceRegisterDeviceArrivalEventHandler
  - Interface Access, [177](#)
- spinInterfaceRegisterDeviceRemovalEventHandler
  - Interface Access, [177](#)
- spinInterfaceRegisterInterfaceEventHandler
  - Interface Access, [178](#)
- spinInterfaceRelease
  - Interface Access, [178](#)
- spinInterfaceSendActionCommand
  - Interface Access, [179](#)
- spinInterfaceType
  - Spinnaker C GenICam Enumerations, [326](#)
- spinInterfaceUnregisterDeviceArrivalEventHandler
  - Interface Access, [179](#)
- spinInterfaceUnregisterDeviceRemovalEventHandler
  - Interface Access, [180](#)
- spinInterfaceUnregisterInterfaceEventHandler
  - Interface Access, [180](#)
- spinInterfaceUpdateCameras
  - Interface Access, [181](#)
- spinJPEGOption, [477](#)
  - progressive, [478](#)
  - quality, [478](#)
  - reserved, [478](#)
- spinJPG2Option, [479](#)
  - quality, [479](#)
  - reserved, [479](#)

- spinLUTSelectorEnums
  - Camera Enumerations, [105](#)
- spinLibraryVersion, [479](#)
  - build, [480](#)
  - major, [480](#)
  - minor, [480](#)
  - type, [480](#)
- spinLineFormatEnums
  - Camera Enumerations, [101](#)
- spinLineInputFilterSelectorEnums
  - Camera Enumerations, [101](#)
- spinLineModeEnums
  - Camera Enumerations, [102](#)
- spinLineSelectorEnums
  - Camera Enumerations, [102](#)
- spinLineSourceEnums
  - Camera Enumerations, [102](#)
- spinLinkType
  - Spinnaker C GenICam Enumerations, [327](#)
- spinLogDataGetCategoryName
  - Logging Event Data Access, [240](#)
- spinLogDataGetLogMessage
  - Logging Event Data Access, [241](#)
- spinLogDataGetNDC
  - Logging Event Data Access, [241](#)
- spinLogDataGetPriority
  - Logging Event Data Access, [242](#)
- spinLogDataGetPriorityName
  - Logging Event Data Access, [242](#)
- spinLogDataGetThreadName
  - Logging Event Data Access, [243](#)
- spinLogDataGetTimestamp
  - Logging Event Data Access, [243](#)
- spinLogEventData
  - Spinnaker C Handles, [252](#)
- spinLogEventFunction
  - Spinnaker C Function Signatures, [254](#)
- spinLogEventHandler
  - Spinnaker C Handles, [252](#)
- spinLogEventHandlerCreate
  - Event Access, [229](#)
- spinLogEventHandlerDestroy
  - Event Access, [230](#)
- spinLogicBlockLUTInputActivationEnums
  - Camera Enumerations, [103](#)
- spinLogicBlockLUTInputSelectorEnums
  - Camera Enumerations, [103](#)
- spinLogicBlockLUTInputSourceEnums
  - Camera Enumerations, [104](#)
- spinLogicBlockLUTSelectorEnums
  - Camera Enumerations, [104](#)
- spinLogicBlockSelectorEnums
  - Camera Enumerations, [105](#)
- spinMJPEGOption, [481](#)
  - frameRate, [481](#)
  - quality, [481](#)
  - reserved, [481](#)
- spinNameSpace
  - Spinnaker C GenICam Enumerations, [328](#)
- spinNodeCallbackFunction
  - Spinnaker C GenICam Handles, [320](#)
- spinNodeCallbackHandle
  - Spinnaker C GenICam Handles, [320](#)
- spinNodeDeregisterCallback
  - Node Access, [273](#)
- spinNodeFromString
  - IValue Access, [284](#)
- spinNodeFromStringEx
  - IValue Access, [285](#)
- spinNodeGetAccessMode
  - Node Access, [273](#)
- spinNodeGetCachingMode
  - Node Access, [274](#)
- spinNodeGetDescription
  - Node Access, [274](#)
- spinNodeGetDisplayName
  - Node Access, [275](#)
- spinNodeGetImposedAccessMode
  - Node Access, [276](#)
- spinNodeGetImposedVisibility
  - Node Access, [276](#)
- spinNodeGetName
  - Node Access, [276](#)
- spinNodeGetNameSpace
  - Node Access, [277](#)
- spinNodeGetPollingTime
  - Node Access, [277](#)
- spinNodeGetToolTip
  - Node Access, [278](#)
- spinNodeGetType
  - Node Access, [278](#)
- spinNodeGetVisibility
  - Node Access, [279](#)
- spinNodeHandle
  - Spinnaker C GenICam Handles, [320](#)
- spinNodeInvalidateNode
  - Node Access, [279](#)
- spinNodesAvailable
  - Node Access, [280](#)
- spinNodesEqual
  - Node Access, [280](#)
- spinNodesImplemented
  - Node Access, [281](#)
- spinNodesReadable
  - Node Access, [281](#)
- spinNodesWritable
  - Node Access, [282](#)
- spinNodeMapGetNode
  - Node Map Access, [268](#)
- spinNodeMapGetNodeByIndex
  - Node Map Access, [269](#)
- spinNodeMapGetNumNodes
  - Node Map Access, [269](#)
- spinNodeMapHandle
  - Spinnaker C GenICam Handles, [321](#)
- spinNodeMapPoll

- Node Map Access, [270](#)
- spinNodeMapReleaseNode
  - Node Map Access, [270](#)
- spinNodeRegisterCallback
  - Node Access, [282](#)
- spinNodeToString
  - IValue Access, [285](#)
- spinNodeToStringEx
  - IValue Access, [286](#)
- spinNodeType
  - Spinnaker C GenICam Enumerations, [328](#)
- spinPGMOption, [482](#)
  - binaryFile, [482](#)
  - reserved, [482](#)
- spinPNGOption, [483](#)
  - compressionLevel, [483](#)
  - interlaced, [483](#)
  - reserved, [483](#)
- spinPPMOption, [484](#)
  - binaryFile, [484](#)
  - reserved, [484](#)
- spinPayloadTypeInfoIds
  - Spinnaker C Enumerations, [261](#)
- spinPixelColorFilterEnums
  - Camera Enumerations, [105](#)
- spinPixelFormatEnums
  - Camera Enumerations, [106](#)
- spinPixelFormatInfoSelectorEnums
  - Camera Enumerations, [111](#)
- spinPixelFormatNamespacedID
  - Spinnaker C Enumerations, [261](#)
- spinPixelSizeEnums
  - Camera Enumerations, [117](#)
- spinRegionDestinationEnums
  - Camera Enumerations, [118](#)
- spinRegionModeEnums
  - Camera Enumerations, [118](#)
- spinRegionSelectorEnums
  - Camera Enumerations, [118](#)
- spinRegisterGet
  - IRegister Access, [315](#)
- spinRegisterGetAddress
  - IRegister Access, [316](#)
- spinRegisterGetEx
  - IRegister Access, [316](#)
- spinRegisterGetLength
  - IRegister Access, [317](#)
- spinRegisterSet
  - IRegister Access, [318](#)
- spinRegisterSetEx
  - IRegister Access, [318](#)
- spinRegisterSetReference
  - IRegister Access, [319](#)
- spinRemovalEventFunction
  - Spinnaker C Function Signatures, [254](#)
- spinRepresentation
  - Spinnaker C GenICam Enumerations, [329](#)
- spinRgbTransformLightSourceEnums
  - Camera Enumerations, [118](#)
- spinScan3dCoordinateReferenceSelectorEnums
  - Camera Enumerations, [119](#)
- spinScan3dCoordinateSelectorEnums
  - Camera Enumerations, [119](#)
- spinScan3dCoordinateSystemEnums
  - Camera Enumerations, [120](#)
- spinScan3dCoordinateSystemReferenceEnums
  - Camera Enumerations, [120](#)
- spinScan3dCoordinateTransformSelectorEnums
  - Camera Enumerations, [120](#)
- spinScan3dDistanceUnitEnums
  - Camera Enumerations, [121](#)
- spinScan3dOutputModeEnums
  - Camera Enumerations, [121](#)
- spinSensorDigitizationTapsEnums
  - Camera Enumerations, [122](#)
- spinSensorShutterModeEnums
  - Camera Enumerations, [122](#)
- spinSensorTapsEnums
  - Camera Enumerations, [123](#)
- spinSequencerConfigurationModeEnums
  - Camera Enumerations, [123](#)
- spinSequencerConfigurationValidEnums
  - Camera Enumerations, [124](#)
- spinSequencerModeEnums
  - Camera Enumerations, [124](#)
- spinSequencerSetValidEnums
  - Camera Enumerations, [124](#)
- spinSequencerTriggerActivationEnums
  - Camera Enumerations, [124](#)
- spinSequencerTriggerSourceEnums
  - Camera Enumerations, [125](#)
- spinSerialPortBaudRateEnums
  - Camera Enumerations, [125](#)
- spinSerialPortParityEnums
  - Camera Enumerations, [126](#)
- spinSerialPortSelectorEnums
  - Camera Enumerations, [126](#)
- spinSerialPortSourceEnums
  - Camera Enumerations, [126](#)
- spinSerialPortStopBitsEnums
  - Camera Enumerations, [127](#)
- spinSign
  - Spinnaker C GenICam Enumerations, [329](#)
- spinSlope
  - Spinnaker C GenICam Enumerations, [329](#)
- spinSoftwareSignalSelectorEnums
  - Camera Enumerations, [127](#)
- spinSourceSelectorEnums
  - Camera Enumerations, [127](#)
- spinStandardNameSpace
  - Spinnaker C GenICam Enumerations, [330](#)
- spinStatisticsChannel
  - Spinnaker C Enumerations, [262](#)
- spinStringGetMaxLength
  - String Access, [287](#)
- spinStringGetValue

- String Access, [288](#)
- spinStringGetValueEx
  - String Access, [288](#)
- spinStringSetValue
  - String Access, [289](#)
- spinStringSetValueEx
  - String Access, [289](#)
- spinSystem
  - Spinnaker C Handles, [252](#)
- spinSystemGetCameras
  - System Access, [151](#)
- spinSystemGetCamerasEx
  - System Access, [152](#)
- spinSystemGetInstance
  - System Access, [152](#)
- spinSystemGetInterfaces
  - System Access, [154](#)
- spinSystemGetLibraryVersion
  - System Access, [154](#)
- spinSystemGetLoggingLevel
  - System Access, [154](#)
- spinSystemGetTLNodeMap
  - System Access, [155](#)
- spinSystemIsInUse
  - System Access, [155](#)
- spinSystemRegisterDeviceArrivalEventHandler
  - System Access, [156](#)
- spinSystemRegisterDeviceRemovalEventHandler
  - System Access, [156](#)
- spinSystemRegisterInterfaceEventHandler
  - System Access, [157](#)
- spinSystemRegisterLogEventHandler
  - System Access, [157](#)
- spinSystemReleaseInstance
  - System Access, [158](#)
- spinSystemSendActionCommand
  - System Access, [158](#)
- spinSystemSetLoggingLevel
  - System Access, [159](#)
- spinSystemUnregisterAllLogEventHandlers
  - System Access, [160](#)
- spinSystemUnregisterDeviceArrivalEventHandler
  - System Access, [160](#)
- spinSystemUnregisterDeviceRemovalEventHandler
  - System Access, [160](#)
- spinSystemUnregisterInterfaceEventHandler
  - System Access, [161](#)
- spinSystemUnregisterLogEventHandler
  - System Access, [161](#)
- spinSystemUpdateCameras
  - System Access, [162](#)
- spinSystemUpdateCamerasEx
  - System Access, [162](#)
- spinTIFFOption, [484](#)
  - compression, [485](#)
  - reserved, [485](#)
- spinTLDeviceAccessStatusEnums
  - Transport Layer Enumerations, [337](#)
- spinTLDeviceCurrentSpeedEnums
  - Transport Layer Enumerations, [339](#)
- spinTLDeviceEndianessMechanismEnums
  - Transport Layer Enumerations, [339](#)
- spinTLDeviceTypeEnums
  - Transport Layer Enumerations, [339](#)
- spinTLFilterDriverStatusEnums
  - Transport Layer Enumerations, [340](#)
- spinTLGUIXMLLocationEnums
  - Transport Layer Enumerations, [341](#)
- spinTLGenICamXMLLocationEnums
  - Transport Layer Enumerations, [340](#)
- spinTLGevCCPEnums
  - Transport Layer Enumerations, [340](#)
- spinTLInterfaceTypeEnums
  - Transport Layer Enumerations, [341](#)
- spinTLPOEStatusEnums
  - Transport Layer Enumerations, [341](#)
- spinTLStreamBufferCountModeEnums
  - Transport Layer Enumerations, [342](#)
- spinTLStreamBufferHandlingModeEnums
  - Transport Layer Enumerations, [342](#)
- spinTLStreamTypeEnums
  - Transport Layer Enumerations, [343](#)
- spinTLTLTypeEnums
  - Transport Layer Enumerations, [343](#)
- spinTestPatternEnums
  - Camera Enumerations, [127](#)
- spinTestPatternGeneratorSelectorEnums
  - Camera Enumerations, [128](#)
- spinTimerSelectorEnums
  - Camera Enumerations, [128](#)
- spinTimerStatusEnums
  - Camera Enumerations, [128](#)
- spinTimerTriggerActivationEnums
  - Camera Enumerations, [129](#)
- spinTimerTriggerSourceEnums
  - Camera Enumerations, [129](#)
- spinTransferComponentSelectorEnums
  - Camera Enumerations, [131](#)
- spinTransferControlModeEnums
  - Camera Enumerations, [131](#)
- spinTransferOperationModeEnums
  - Camera Enumerations, [131](#)
- spinTransferQueueModeEnums
  - Camera Enumerations, [132](#)
- spinTransferSelectorEnums
  - Camera Enumerations, [132](#)
- spinTransferStatusSelectorEnums
  - Camera Enumerations, [132](#)
- spinTransferTriggerActivationEnums
  - Camera Enumerations, [133](#)
- spinTransferTriggerModeEnums
  - Camera Enumerations, [133](#)
- spinTransferTriggerSelectorEnums
  - Camera Enumerations, [133](#)
- spinTransferTriggerSourceEnums
  - Camera Enumerations, [134](#)

- spinTriggerActivationEnums
  - Camera Enumerations, [135](#)
- spinTriggerModeEnums
  - Camera Enumerations, [135](#)
- spinTriggerOverlapEnums
  - Camera Enumerations, [136](#)
- spinTriggerSelectorEnums
  - Camera Enumerations, [136](#)
- spinTriggerSourceEnums
  - Camera Enumerations, [136](#)
- spinUserOutputSelectorEnums
  - Camera Enumerations, [137](#)
- spinUserSetDefaultEnums
  - Camera Enumerations, [137](#)
- spinUserSetSelectorEnums
  - Camera Enumerations, [138](#)
- spinVideo
  - Spinnaker C Handles, [252](#)
- SpinVideo Recording Access, [333](#)
  - spinVideoAppend, [333](#)
  - spinVideoClose, [333](#)
  - spinVideoOpenH264, [334](#)
  - spinVideoOpenMJPEG, [334](#)
  - spinVideoOpenUncompressed, [334](#)
  - spinVideoSetMaximumFileSize, [334](#)
- spinVideoAppend
  - SpinVideo Recording Access, [333](#)
- spinVideoClose
  - SpinVideo Recording Access, [333](#)
- spinVideoOpenH264
  - SpinVideo Recording Access, [334](#)
- spinVideoOpenMJPEG
  - SpinVideo Recording Access, [334](#)
- spinVideoOpenUncompressed
  - SpinVideo Recording Access, [334](#)
- spinVideoSetMaximumFileSize
  - SpinVideo Recording Access, [334](#)
- spinVisibility
  - Spinnaker C GenICam Enumerations, [330](#)
- spinWhiteClipSelectorEnums
  - Camera Enumerations, [138](#)
- spinXMLValidation
  - Spinnaker C GenICam Enumerations, [330](#)
- spinYesNo
  - Spinnaker C GenICam Enumerations, [332](#)
- Spinnaker C API, [143](#)
  - spinCameraDiscoverMaxPacketSize, [144](#)
- Spinnaker C Definitions, [21](#)
  - bool8\_t, [22](#)
  - False, [22](#)
  - True, [22](#)
- Spinnaker C Enumerations, [255](#)
  - spinColorProcessingAlgorithm, [257](#)
  - spinError, [258](#)
  - spinImageFileFormat, [259](#)
  - spinImageStatus, [260](#)
  - spinPayloadTypeInfoIDs, [261](#)
  - spinPixelFormatNamespacedID, [261](#)
  - spinStatisticsChannel, [262](#)
  - spinnakerLogLevel, [260](#)
- Spinnaker C Function Signatures, [253](#)
  - spinArrivalEventFunction, [253](#)
  - spinDeviceEventFunction, [253](#)
  - spinImageEventFunction, [253](#)
  - spinLogEventFunction, [254](#)
  - spinRemovalEventFunction, [254](#)
- Spinnaker C GenICam API, [265](#)
  - spinCategoryReleaseNode, [266](#)
- Spinnaker C GenICam Enumerations, [322](#)
  - spinAccessMode, [324](#)
  - spinCachingMode, [325](#)
  - spinDisplayNotation, [325](#)
  - spinEndianess, [325](#)
  - spinIncMode, [326](#)
  - spinInputDirection, [326](#)
  - spinInterfaceType, [326](#)
  - spinLinkType, [327](#)
  - spinNameSpace, [328](#)
  - spinNodeType, [328](#)
  - spinRepresentation, [329](#)
  - spinSign, [329](#)
  - spinSlope, [329](#)
  - spinStandardNameSpace, [330](#)
  - spinVisibility, [330](#)
  - spinXMLValidation, [330](#)
  - spinYesNo, [332](#)
- Spinnaker C GenICam Handles, [320](#)
  - spinNodeCallbackFunction, [320](#)
  - spinNodeCallbackHandle, [320](#)
  - spinNodeHandle, [320](#)
  - spinNodeMapHandle, [321](#)
- Spinnaker C Handles, [249](#)
  - spinCamera, [250](#)
  - spinCameraList, [250](#)
  - spinDeviceArrivalEventHandler, [250](#)
  - spinDeviceEventData, [250](#)
  - spinDeviceEventHandler, [250](#)
  - spinDeviceRemovalEventHandler, [251](#)
  - spinImage, [251](#)
  - spinImageEventHandler, [251](#)
  - spinImageStatistics, [251](#)
  - spinInterface, [251](#)
  - spinInterfaceEventHandler, [251](#)
  - spinInterfaceList, [252](#)
  - spinLogEventData, [252](#)
  - spinLogEventHandler, [252](#)
  - spinSystem, [252](#)
  - spinVideo, [252](#)
- Spinnaker C QuickSpin API, [140](#)
- Spinnaker C Structures, [263](#)
  - actionCommandStatus, [264](#)
  - spinCompressionMethod, [264](#)
- SpinnakerC.h
  - spinCameraForceIP, [532](#)
- spinnakerLogLevel
  - Spinnaker C Enumerations, [260](#)

- SpinnakerPlatformC.h
  - SPINNAKERC\_API, 545
- Status
  - actionCommandResult, 349
- StreamAnnounceBufferMinimum
  - quickSpinTLStream, 461
- StreamAnnouncedBufferCount
  - quickSpinTLStream, 462
- StreamBlockTransferSize
  - quickSpinTLStream, 462
- StreamBufferAlignment
  - quickSpinTLStream, 462
- StreamBufferCountManual
  - quickSpinTLStream, 462
- StreamBufferCountMax
  - quickSpinTLStream, 462
- StreamBufferCountMode
  - quickSpinTLStream, 462
- StreamBufferCountResult
  - quickSpinTLStream, 462
- StreamBufferHandlingMode
  - quickSpinTLStream, 462
- StreamCRCCheckEnable
  - quickSpinTLStream, 463
- StreamChunkCountMaximum
  - quickSpinTLStream, 463
- StreamDeliveredFrameCount
  - quickSpinTLStream, 463
- StreamDroppedFrameCount
  - quickSpinTLStream, 463
- StreamFailedBufferCount
  - quickSpinTLStream, 463
- StreamID
  - quickSpinTLStream, 463
- StreamInputBufferCount
  - quickSpinTLStream, 463
- StreamIsGrabbing
  - quickSpinTLStream, 463
- StreamLostFrameCount
  - quickSpinTLStream, 464
- StreamOutputBufferCount
  - quickSpinTLStream, 464
- StreamStartedFrameCount
  - quickSpinTLStream, 464
- StreamType
  - quickSpinTLStream, 464
- String Access, 287
  - spinStringGetMaxLength, 287
  - spinStringGetValue, 288
  - spinStringGetValueEx, 288
  - spinStringSetValue, 289
  - spinStringSetValueEx, 289
- System Access, 150
  - spinSystemGetCameras, 151
  - spinSystemGetCamerasEx, 152
  - spinSystemGetInstance, 152
  - spinSystemGetInterfaces, 154
  - spinSystemGetLibraryVersion, 154
  - spinSystemGetLoggingLevel, 154
  - spinSystemGetTLNodeMap, 155
  - spinSystemIsInUse, 155
  - spinSystemRegisterDeviceArrivalEventHandler, 156
  - spinSystemRegisterDeviceRemovalEventHandler, 156
  - spinSystemRegisterInterfaceEventHandler, 157
  - spinSystemRegisterLogEventHandler, 157
  - spinSystemReleaseInstance, 158
  - spinSystemSendActionCommand, 158
  - spinSystemSetLoggingLevel, 159
  - spinSystemUnregisterAllLogEventHandlers, 160
  - spinSystemUnregisterDeviceArrivalEventHandler, 160
  - spinSystemUnregisterDeviceRemovalEventHandler, 160
  - spinSystemUnregisterInterfaceEventHandler, 161
  - spinSystemUnregisterLogEventHandler, 161
  - spinSystemUpdateCameras, 162
  - spinSystemUpdateCamerasEx, 162
- TLDevice Structures, 345
- TLDisplayName
  - quickSpinTLSystem, 467
- TLFileName
  - quickSpinTLSystem, 467
- TLID
  - quickSpinTLSystem, 467
- TLInterface Structures, 346
- TLModelName
  - quickSpinTLSystem, 467
- TLParamsLocked
  - quickSpin, 440
- TLPath
  - quickSpinTLSystem, 467
- TLStream Structures, 347
- TLSystem Structures, 348
- TLType
  - quickSpinTLSystem, 468
- TLVendorName
  - quickSpinTLSystem, 468
- TLVersion
  - quickSpinTLSystem, 468
- Test0001
  - quickSpin, 438
- TestEventGenerate
  - quickSpin, 438
- TestPattern
  - quickSpin, 438
- TestPatternGeneratorSelector
  - quickSpin, 439
- TestPendingAck
  - quickSpin, 439
- TimerDelay
  - quickSpin, 439
- TimerDuration
  - quickSpin, 439
- TimerReset

- quickSpin, [439](#)
- TimerSelector
  - quickSpin, [439](#)
- TimerStatus
  - quickSpin, [439](#)
- TimerTriggerActivation
  - quickSpin, [439](#)
- TimerTriggerSource
  - quickSpin, [440](#)
- TimerValue
  - quickSpin, [440](#)
- Timestamp
  - quickSpin, [440](#)
- TimestampLatch
  - quickSpin, [440](#)
- TimestampLatchValue
  - quickSpin, [440](#)
- TimestampReset
  - quickSpin, [440](#)
- TransferAbort
  - quickSpin, [440](#)
- TransferBlockCount
  - quickSpin, [441](#)
- TransferBurstCount
  - quickSpin, [441](#)
- TransferComponentSelector
  - quickSpin, [441](#)
- TransferControlMode
  - quickSpin, [441](#)
- TransferOperationMode
  - quickSpin, [441](#)
- TransferPause
  - quickSpin, [441](#)
- TransferQueueCurrentBlockCount
  - quickSpin, [441](#)
- TransferQueueMaxBlockCount
  - quickSpin, [441](#)
- TransferQueueMode
  - quickSpin, [442](#)
- TransferQueueOverflowCount
  - quickSpin, [442](#)
- TransferResume
  - quickSpin, [442](#)
- TransferSelector
  - quickSpin, [442](#)
- TransferStart
  - quickSpin, [442](#)
- TransferStatus
  - quickSpin, [442](#)
- TransferStatusSelector
  - quickSpin, [442](#)
- TransferStop
  - quickSpin, [442](#)
- TransferStreamChannel
  - quickSpin, [443](#)
- TransferTriggerActivation
  - quickSpin, [443](#)
- TransferTriggerMode
  - quickSpin, [443](#)
- quickSpin, [443](#)
- TransferTriggerSelector
  - quickSpin, [443](#)
- TransferTriggerSource
  - quickSpin, [443](#)
- Transport Layer Enumerations, [336](#)
  - spinTLDeviceAccessStatusEnums, [337](#)
  - spinTLDeviceCurrentSpeedEnums, [339](#)
  - spinTLDeviceEndiannessMechanismEnums, [339](#)
  - spinTLDeviceTypeEnums, [339](#)
  - spinTLFilterDriverStatusEnums, [340](#)
  - spinTLGUIXMLLocationEnums, [341](#)
  - spinTLGenICamXMLLocationEnums, [340](#)
  - spinTLGevCCPEnums, [340](#)
  - spinTLInterfaceTypeEnums, [341](#)
  - spinTLPOEStatusEnums, [341](#)
  - spinTLStreamBufferCountModeEnums, [342](#)
  - spinTLStreamBufferHandlingModeEnums, [342](#)
  - spinTLStreamTypeEnums, [343](#)
  - spinTLTLTypeEnums, [343](#)
- TriggerActivation
  - quickSpin, [443](#)
- TriggerDelay
  - quickSpin, [443](#)
- TriggerDivider
  - quickSpin, [443](#)
- TriggerEventTest
  - quickSpin, [444](#)
- TriggerMode
  - quickSpin, [444](#)
- TriggerMultiplier
  - quickSpin, [444](#)
- TriggerOverlap
  - quickSpin, [444](#)
- TriggerSelector
  - quickSpin, [444](#)
- TriggerSoftware
  - quickSpin, [444](#)
- TriggerSource
  - quickSpin, [444](#)
- True
  - Spinnaker C Definitions, [22](#)
- type
  - spinLibraryVersion, [480](#)
- UserOutputSelector
  - quickSpin, [444](#)
- UserOutputValue
  - quickSpin, [445](#)
- UserOutputValueAll
  - quickSpin, [445](#)
- UserOutputValueAllMask
  - quickSpin, [445](#)
- UserSetDefault
  - quickSpin, [445](#)
- UserSetFeatureEnable
  - quickSpin, [445](#)
- UserSetLoad
  - quickSpin, [445](#)

UserSetSave  
    [quickSpin, 445](#)  
UserSetSelector  
    [quickSpin, 445](#)  
  
V3\_3Enable  
    [quickSpin, 446](#)  
  
WhiteClip  
    [quickSpin, 446](#)  
WhiteClipSelector  
    [quickSpin, 446](#)  
Width  
    [quickSpin, 446](#)  
width  
    [spinH264Option, 477](#)  
WidthMax  
    [quickSpin, 446](#)