

Spinnaker C

3.1.0.79

Generated by Doxygen 1.9.1

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
11.2 Camera Enumerations	22
11.3 Chunk Data Structures	22
11.4 Spinnaker C QuickSpin API	22
11.4.1 Detailed Description	22
11.5 QuickSpin Access	22
11.6 Spinnaker C API	23
11.6.1 Detailed Description	23
11.7 Error Handling	23
11.8 System Access	23

11.9 InterfaceList Access	24
11.10 CameraList Access	24
11.11 ImageList Access	24
11.12 Interface Access	24
11.13 Camera Access	24
11.14 Image Access	25
11.15 Image Processor Access	25
11.16 Event Access	28
11.17 ImageStatistics Access	28
11.18 Logging Event Data Access	28
11.19 Device Event Data Access	28
11.20 Chunk data access	28
11.21 Spinnaker C Handles	29
11.22 Spinnaker C Function Signatures	29
11.23 Spinnaker C Enumerations	29
11.24 Spinnaker C Structures	29
11.25 Spinnaker C GenICam API	29
11.26 Node Map Access	29
11.27 Node Access	29
11.28 IValue Access	30
11.29 String Access	30
11.30 IInteger Access	30
11.31 IFloat Access	30
11.32 IEnumeration Access	30
11.33 IEnumEntry Access	31
11.34 IBoolean Access	31
11.35 ICommand Access	31
11.36 ICategory Access	31
11.37 IRegister Access	31
11.38 Spinnaker C GenICam Handles	31
11.39 Spinnaker C GenICam Enumerations	32
11.40 SpinVideo Recording Access	32
11.41 Transport Layer Enumerations	32
11.42 TLDevice Structures	32
11.43 TLInterface Structures	32
11.44 TLStream Structures	33
11.45 TLSystem Structures	33
12 Data Structure Documentation	35
12.1 actionCommandResult Struct Reference	35
12.1.1 Detailed Description	35
12.1.2 Field Documentation	35

12.1.2.1 DeviceAddress	35
12.1.2.2 Status	35
12.2 quickSpin Struct Reference	36
12.2.1 Field Documentation	48
12.2.1.1 AasRoiEnable	48
12.2.1.2 AasRoiHeight	48
12.2.1.3 AasRoiOffsetX	48
12.2.1.4 AasRoiOffsetY	48
12.2.1.5 AasRoiWidth	49
12.2.1.6 AcquisitionAbort	49
12.2.1.7 AcquisitionArm	49
12.2.1.8 AcquisitionBurstFrameCount	49
12.2.1.9 AcquisitionFrameCount	49
12.2.1.10 AcquisitionFrameRate	49
12.2.1.11 AcquisitionFrameRateEnable	49
12.2.1.12 AcquisitionLineRate	49
12.2.1.13 AcquisitionMode	50
12.2.1.14 AcquisitionResultingFrameRate	50
12.2.1.15 AcquisitionStart	50
12.2.1.16 AcquisitionStatus	50
12.2.1.17 AcquisitionStatusSelector	50
12.2.1.18 AcquisitionStop	50
12.2.1.19 ActionDeviceKey	50
12.2.1.20 ActionGroupKey	50
12.2.1.21 ActionGroupMask	51
12.2.1.22 ActionQueueSize	51
12.2.1.23 ActionSelector	51
12.2.1.24 ActionUnconditionalMode	51
12.2.1.25 AdaptiveCompressionEnable	51
12.2.1.26 AdcBitDepth	51
12.2.1.27 aPAUSEMACCtrlFramesReceived	51
12.2.1.28 aPAUSEMACCtrlFramesTransmitted	51
12.2.1.29 AutoAlgorithmSelector	52
12.2.1.30 AutoExposureControlLoopDamping	52
12.2.1.31 AutoExposureControlPriority	52
12.2.1.32 AutoExposureEVCompensation	52
12.2.1.33 AutoExposureExposureTimeLowerLimit	52
12.2.1.34 AutoExposureExposureTimeUpperLimit	52
12.2.1.35 AutoExposureGainLowerLimit	52
12.2.1.36 AutoExposureGainUpperLimit	52
12.2.1.37 AutoExposureGreyValueLowerLimit	53
12.2.1.38 AutoExposureGreyValueUpperLimit	53

12.2.1.39 AutoExposureLightingMode	53
12.2.1.40 AutoExposureMeteringMode	53
12.2.1.41 AutoExposureTargetGreyValue	53
12.2.1.42 AutoExposureTargetGreyValueAuto	53
12.2.1.43 BalanceRatio	53
12.2.1.44 BalanceRatioSelector	53
12.2.1.45 BalanceWhiteAuto	54
12.2.1.46 BalanceWhiteAutoDamping	54
12.2.1.47 BalanceWhiteAutoLowerLimit	54
12.2.1.48 BalanceWhiteAutoProfile	54
12.2.1.49 BalanceWhiteAutoUpperLimit	54
12.2.1.50 BinningHorizontal	54
12.2.1.51 BinningHorizontalMode	54
12.2.1.52 BinningSelector	54
12.2.1.53 BinningVertical	55
12.2.1.54 BinningVerticalMode	55
12.2.1.55 BlackLevel	55
12.2.1.56 BlackLevelAuto	55
12.2.1.57 BlackLevelAutoBalance	55
12.2.1.58 BlackLevelClampingEnable	55
12.2.1.59 BlackLevelRaw	55
12.2.1.60 BlackLevelSelector	55
12.2.1.61 ChunkBlackLevel	56
12.2.1.62 ChunkBlackLevelSelector	56
12.2.1.63 ChunkCompressionMode	56
12.2.1.64 ChunkCompressionRatio	56
12.2.1.65 ChunkCounterSelector	56
12.2.1.66 ChunkCounterValue	56
12.2.1.67 ChunkCRC	56
12.2.1.68 ChunkEnable	56
12.2.1.69 ChunkEncoderSelector	57
12.2.1.70 ChunkEncoderStatus	57
12.2.1.71 ChunkEncoderValue	57
12.2.1.72 ChunkExposureEndLineStatusAll	57
12.2.1.73 ChunkExposureTime	57
12.2.1.74 ChunkExposureTimeSelector	57
12.2.1.75 ChunkFrameID	57
12.2.1.76 ChunkGain	57
12.2.1.77 ChunkGainSelector	58
12.2.1.78 ChunkHeight	58
12.2.1.79 ChunkImage	58
12.2.1.80 ChunkImageComponent	58

12.2.1.81 ChunkInferenceBoundingBoxResult	58
12.2.1.82 ChunkInferenceConfidence	58
12.2.1.83 ChunkInferenceFrameId	58
12.2.1.84 ChunkInferenceResult	58
12.2.1.85 ChunkLinePitch	59
12.2.1.86 ChunkLineStatusAll	59
12.2.1.87 ChunkModeActive	59
12.2.1.88 ChunkOffsetX	59
12.2.1.89 ChunkOffsetY	59
12.2.1.90 ChunkPartSelector	59
12.2.1.91 ChunkPixelDynamicRangeMax	59
12.2.1.92 ChunkPixelDynamicRangeMin	59
12.2.1.93 ChunkPixelFormat	60
12.2.1.94 ChunkRegionID	60
12.2.1.95 ChunkScan3dAxisMax	60
12.2.1.96 ChunkScan3dAxisMin	60
12.2.1.97 ChunkScan3dCoordinateOffset	60
12.2.1.98 ChunkScan3dCoordinateReferenceSelector	60
12.2.1.99 ChunkScan3dCoordinateReferenceValue	60
12.2.1.100 ChunkScan3dCoordinateScale	60
12.2.1.101 ChunkScan3dCoordinateSelector	61
12.2.1.102 ChunkScan3dCoordinateSystem	61
12.2.1.103 ChunkScan3dCoordinateSystemReference	61
12.2.1.104 ChunkScan3dCoordinateTransformSelector	61
12.2.1.105 ChunkScan3dDistanceUnit	61
12.2.1.106 ChunkScan3dInvalidDataFlag	61
12.2.1.107 ChunkScan3dInvalidDataValue	61
12.2.1.108 ChunkScan3dOutputMode	61
12.2.1.109 ChunkScan3dTransformValue	62
12.2.1.110 ChunkScanLineSelector	62
12.2.1.111 ChunkSelector	62
12.2.1.112 ChunkSequencerSetActive	62
12.2.1.113 ChunkSerialData	62
12.2.1.114 ChunkSerialDataLength	62
12.2.1.115 ChunkSerialReceiveOverflow	62
12.2.1.116 ChunkSourceID	62
12.2.1.117 ChunkStreamChannelID	63
12.2.1.118 ChunkTimerSelector	63
12.2.1.119 ChunkTimerValue	63
12.2.1.120 ChunkTimestamp	63
12.2.1.121 ChunkTimestampLatchValue	63
12.2.1.122 ChunkTransferBlockID	63

12.2.1.123 ChunkTransferQueueCurrentBlockCount	63
12.2.1.124 ChunkTransferStreamID	63
12.2.1.125 ChunkWidth	64
12.2.1.126 CIConfiguration	64
12.2.1.127 CTimeSlotsCount	64
12.2.1.128 ColorTransformationEnable	64
12.2.1.129 ColorTransformationSelector	64
12.2.1.130 ColorTransformationValue	64
12.2.1.131 ColorTransformationValueSelector	64
12.2.1.132 CompressionRatio	64
12.2.1.133 CompressionSaturationPriority	65
12.2.1.134 CounterDelay	65
12.2.1.135 CounterDuration	65
12.2.1.136 CounterEventActivation	65
12.2.1.137 CounterEventSource	65
12.2.1.138 CounterReset	65
12.2.1.139 CounterResetActivation	65
12.2.1.140 CounterResetSource	65
12.2.1.141 CounterSelector	66
12.2.1.142 CounterStatus	66
12.2.1.143 CounterTriggerActivation	66
12.2.1.144 CounterTriggerSource	66
12.2.1.145 CounterValue	66
12.2.1.146 CounterValueAtReset	66
12.2.1.147 CxpConnectionSelector	66
12.2.1.148 CxpConnectionTestErrorCount	66
12.2.1.149 CxpConnectionTestMode	67
12.2.1.150 CxpConnectionTestPacketCount	67
12.2.1.151 CxpLinkConfiguration	67
12.2.1.152 CxpLinkConfigurationPreferred	67
12.2.1.153 CxpLinkConfigurationStatus	67
12.2.1.154 CxpPoCxpAuto	67
12.2.1.155 CxpPoCxpStatus	67
12.2.1.156 CxpPoCxpTripReset	67
12.2.1.157 CxpPoCxpTurnOff	68
12.2.1.158 DecimationHorizontal	68
12.2.1.159 DecimationHorizontalMode	68
12.2.1.160 DecimationSelector	68
12.2.1.161 DecimationVertical	68
12.2.1.162 DecimationVerticalMode	68
12.2.1.163 DefectCorrectionMode	68
12.2.1.164 DefectCorrectStaticEnable	68

12.2.1.165 DefectTableApply	69
12.2.1.166 DefectTableCoordinateX	69
12.2.1.167 DefectTableCoordinateY	69
12.2.1.168 DefectTableFactoryRestore	69
12.2.1.169 DefectTableIndex	69
12.2.1.170 DefectTablePixelCount	69
12.2.1.171 DefectTableSave	69
12.2.1.172 Deinterlacing	69
12.2.1.173 DeviceCharacterSet	70
12.2.1.174 DeviceClockFrequency	70
12.2.1.175 DeviceClockSelector	70
12.2.1.176 DeviceConnectionSelector	70
12.2.1.177 DeviceConnectionSpeed	70
12.2.1.178 DeviceConnectionStatus	70
12.2.1.179 DeviceEventChannelCount	70
12.2.1.180 DeviceFamilyName	70
12.2.1.181 DeviceFeaturePersistenceEnd	71
12.2.1.182 DeviceFeaturePersistenceStart	71
12.2.1.183 DeviceFirmwareVersion	71
12.2.1.184 DeviceGenCPVersionMajor	71
12.2.1.185 DeviceGenCPVersionMinor	71
12.2.1.186 DeviceID	71
12.2.1.187 DeviceIndicatorMode	71
12.2.1.188 DeviceLinkBandwidthReserve	71
12.2.1.189 DeviceLinkCommandTimeout	72
12.2.1.190 DeviceLinkConnectionCount	72
12.2.1.191 DeviceLinkCurrentThroughput	72
12.2.1.192 DeviceLinkHeartbeatMode	72
12.2.1.193 DeviceLinkHeartbeatTimeout	72
12.2.1.194 DeviceLinkSelector	72
12.2.1.195 DeviceLinkSpeed	72
12.2.1.196 DeviceLinkThroughputLimit	72
12.2.1.197 DeviceLinkThroughputLimitMode	73
12.2.1.198 DeviceManifestEntrySelector	73
12.2.1.199 DeviceManifestPrimaryURL	73
12.2.1.200 DeviceManifestSchemaMajorVersion	73
12.2.1.201 DeviceManifestSchemaMinorVersion	73
12.2.1.202 DeviceManifestSecondaryURL	73
12.2.1.203 DeviceManifestXMLMajorVersion	73
12.2.1.204 DeviceManifestXMLMinorVersion	73
12.2.1.205 DeviceManifestXMLSubMinorVersion	74
12.2.1.206 DeviceManufacturerInfo	74

12.2.1.207 DeviceMaxThroughput	74
12.2.1.208 DeviceModelName	74
12.2.1.209 DevicePowerSupplySelector	74
12.2.1.210 DeviceRegistersCheck	74
12.2.1.211 DeviceRegistersEndianness	74
12.2.1.212 DeviceRegistersStreamingEnd	74
12.2.1.213 DeviceRegistersStreamingStart	75
12.2.1.214 DeviceRegistersValid	75
12.2.1.215 DeviceReset	75
12.2.1.216 DeviceScanType	75
12.2.1.217 DeviceSerialNumber	75
12.2.1.218 DeviceSerialPortBaudRate	75
12.2.1.219 DeviceSerialPortSelector	75
12.2.1.220 DeviceSFNCVersionMajor	75
12.2.1.221 DeviceSFNCVersionMinor	76
12.2.1.222 DeviceSFNCVersionSubMinor	76
12.2.1.223 DeviceStreamChannelCount	76
12.2.1.224 DeviceStreamChannelEndianness	76
12.2.1.225 DeviceStreamChannelLink	76
12.2.1.226 DeviceStreamChannelPacketSize	76
12.2.1.227 DeviceStreamChannelSelector	76
12.2.1.228 DeviceStreamChannelType	76
12.2.1.229 DeviceTapGeometry	77
12.2.1.230 DeviceTemperature	77
12.2.1.231 DeviceTemperatureSelector	77
12.2.1.232 DeviceTLType	77
12.2.1.233 DeviceTLVersionMajor	77
12.2.1.234 DeviceTLVersionMinor	77
12.2.1.235 DeviceTLVersionSubMinor	77
12.2.1.236 DeviceType	77
12.2.1.237 DeviceUptime	78
12.2.1.238 DeviceUserID	78
12.2.1.239 DeviceVendorName	78
12.2.1.240 DeviceVersion	78
12.2.1.241 EncoderDivider	78
12.2.1.242 EncoderMode	78
12.2.1.243 EncoderOutputMode	78
12.2.1.244 EncoderReset	78
12.2.1.245 EncoderResetActivation	79
12.2.1.246 EncoderResetSource	79
12.2.1.247 EncoderSelector	79
12.2.1.248 EncoderSourceA	79

12.2.1.249 EncoderSourceB	79
12.2.1.250 EncoderStatus	79
12.2.1.251 EncoderTimeout	79
12.2.1.252 EncoderValue	79
12.2.1.253 EncoderValueAtReset	80
12.2.1.254 EnumerationCount	80
12.2.1.255 EventAcquisitionEnd	80
12.2.1.256 EventAcquisitionEndFrameID	80
12.2.1.257 EventAcquisitionEndTimestamp	80
12.2.1.258 EventAcquisitionError	80
12.2.1.259 EventAcquisitionErrorFrameID	80
12.2.1.260 EventAcquisitionErrorTimestamp	80
12.2.1.261 EventAcquisitionStart	81
12.2.1.262 EventAcquisitionStartFrameID	81
12.2.1.263 EventAcquisitionStartTimestamp	81
12.2.1.264 EventAcquisitionTransferEnd	81
12.2.1.265 EventAcquisitionTransferEndFrameID	81
12.2.1.266 EventAcquisitionTransferEndTimestamp	81
12.2.1.267 EventAcquisitionTransferStart	81
12.2.1.268 EventAcquisitionTransferStartFrameID	81
12.2.1.269 EventAcquisitionTransferStartTimestamp	82
12.2.1.270 EventAcquisitionTrigger	82
12.2.1.271 EventAcquisitionTriggerFrameID	82
12.2.1.272 EventAcquisitionTriggerTimestamp	82
12.2.1.273 EventActionLate	82
12.2.1.274 EventActionLateFrameID	82
12.2.1.275 EventActionLateTimestamp	82
12.2.1.276 EventCounter0End	82
12.2.1.277 EventCounter0EndFrameID	83
12.2.1.278 EventCounter0EndTimestamp	83
12.2.1.279 EventCounter0Start	83
12.2.1.280 EventCounter0StartFrameID	83
12.2.1.281 EventCounter0StartTimestamp	83
12.2.1.282 EventCounter1End	83
12.2.1.283 EventCounter1EndFrameID	83
12.2.1.284 EventCounter1EndTimestamp	83
12.2.1.285 EventCounter1Start	84
12.2.1.286 EventCounter1StartFrameID	84
12.2.1.287 EventCounter1StartTimestamp	84
12.2.1.288 EventEncoder0Restarted	84
12.2.1.289 EventEncoder0RestartedFrameID	84
12.2.1.290 EventEncoder0RestartedTimestamp	84

12.2.1.291 EventEncoder0Stopped	84
12.2.1.292 EventEncoder0StoppedFrameID	84
12.2.1.293 EventEncoder0StoppedTimestamp	85
12.2.1.294 EventEncoder1 Restarted	85
12.2.1.295 EventEncoder1 RestartedFrameID	85
12.2.1.296 EventEncoder1 RestartedTimestamp	85
12.2.1.297 EventEncoder1 Stopped	85
12.2.1.298 EventEncoder1 StoppedFrameID	85
12.2.1.299 EventEncoder1 StoppedTimestamp	85
12.2.1.300 EventError	85
12.2.1.301 EventErrorCode	86
12.2.1.302 EventErrorFrameID	86
12.2.1.303 EventErrorTimestamp	86
12.2.1.304 EventExposureEnd	86
12.2.1.305 EventExposureEndFrameID	86
12.2.1.306 EventExposureEndTimestamp	86
12.2.1.307 EventExposureStart	86
12.2.1.308 EventExposureStartFrameID	86
12.2.1.309 EventExposureStartTimestamp	87
12.2.1.310 EventFrameBurstEnd	87
12.2.1.311 EventFrameBurstEndFrameID	87
12.2.1.312 EventFrameBurstEndTimestamp	87
12.2.1.313 EventFrameBurstStart	87
12.2.1.314 EventFrameBurstStartFrameID	87
12.2.1.315 EventFrameBurstStartTimestamp	87
12.2.1.316 EventFrameEnd	87
12.2.1.317 EventFrameEndFrameID	88
12.2.1.318 EventFrameEndTimestamp	88
12.2.1.319 EventFrameStart	88
12.2.1.320 EventFrameStartFrameID	88
12.2.1.321 EventFrameStartTimestamp	88
12.2.1.322 EventFrameTransferEnd	88
12.2.1.323 EventFrameTransferEndFrameID	88
12.2.1.324 EventFrameTransferEndTimestamp	88
12.2.1.325 EventFrameTransferStart	89
12.2.1.326 EventFrameTransferStartFrameID	89
12.2.1.327 EventFrameTransferStartTimestamp	89
12.2.1.328 EventFrameTrigger	89
12.2.1.329 EventFrameTriggerFrameID	89
12.2.1.330 EventFrameTriggerTimestamp	89
12.2.1.331 EventLine0AnyEdge	89
12.2.1.332 EventLine0AnyEdgeFrameID	89

12.2.1.333 EventLine0AnyEdgeTimestamp	90
12.2.1.334 EventLine0FallingEdge	90
12.2.1.335 EventLine0FallingEdgeFrameID	90
12.2.1.336 EventLine0FallingEdgeTimestamp	90
12.2.1.337 EventLine0RisingEdge	90
12.2.1.338 EventLine0RisingEdgeFrameID	90
12.2.1.339 EventLine0RisingEdgeTimestamp	90
12.2.1.340 EventLine1AnyEdge	90
12.2.1.341 EventLine1AnyEdgeFrameID	91
12.2.1.342 EventLine1AnyEdgeTimestamp	91
12.2.1.343 EventLine1FallingEdge	91
12.2.1.344 EventLine1FallingEdgeFrameID	91
12.2.1.345 EventLine1FallingEdgeTimestamp	91
12.2.1.346 EventLine1RisingEdge	91
12.2.1.347 EventLine1RisingEdgeFrameID	91
12.2.1.348 EventLine1RisingEdgeTimestamp	91
12.2.1.349 EventLinkSpeedChange	92
12.2.1.350 EventLinkSpeedChangeFrameID	92
12.2.1.351 EventLinkSpeedChangeTimestamp	92
12.2.1.352 EventLinkTrigger0	92
12.2.1.353 EventLinkTrigger0FrameID	92
12.2.1.354 EventLinkTrigger0Timestamp	92
12.2.1.355 EventLinkTrigger1	92
12.2.1.356 EventLinkTrigger1FrameID	92
12.2.1.357 EventLinkTrigger1Timestamp	93
12.2.1.358 EventNotification	93
12.2.1.359 EventSelector	93
12.2.1.360 EventSequencerSetChange	93
12.2.1.361 EventSequencerSetChangeFrameID	93
12.2.1.362 EventSequencerSetChangeTimestamp	93
12.2.1.363 EventSerialData	93
12.2.1.364 EventSerialDataLength	93
12.2.1.365 EventSerialPortReceive	94
12.2.1.366 EventSerialPortReceiveTimestamp	94
12.2.1.367 EventSerialReceiveOverflow	94
12.2.1.368 EventStream0TransferBlockEnd	94
12.2.1.369 EventStream0TransferBlockEndFrameID	94
12.2.1.370 EventStream0TransferBlockEndTimestamp	94
12.2.1.371 EventStream0TransferBlockStart	94
12.2.1.372 EventStream0TransferBlockStartFrameID	94
12.2.1.373 EventStream0TransferBlockStartTimestamp	95
12.2.1.374 EventStream0TransferBlockTrigger	95

12.2.1.375 EventStream0TransferBlockTriggerFrameID	95
12.2.1.376 EventStream0TransferBlockTriggerTimestamp	95
12.2.1.377 EventStream0TransferBurstEnd	95
12.2.1.378 EventStream0TransferBurstEndFrameID	95
12.2.1.379 EventStream0TransferBurstEndTimestamp	95
12.2.1.380 EventStream0TransferBurstStart	95
12.2.1.381 EventStream0TransferBurstStartFrameID	96
12.2.1.382 EventStream0TransferBurstStartTimestamp	96
12.2.1.383 EventStream0TransferEnd	96
12.2.1.384 EventStream0TransferEndFrameID	96
12.2.1.385 EventStream0TransferEndTimestamp	96
12.2.1.386 EventStream0TransferOverflow	96
12.2.1.387 EventStream0TransferOverflowFrameID	96
12.2.1.388 EventStream0TransferOverflowTimestamp	96
12.2.1.389 EventStream0TransferPause	97
12.2.1.390 EventStream0TransferPauseFrameID	97
12.2.1.391 EventStream0TransferPauseTimestamp	97
12.2.1.392 EventStream0TransferResume	97
12.2.1.393 EventStream0TransferResumeFrameID	97
12.2.1.394 EventStream0TransferResumeTimestamp	97
12.2.1.395 EventStream0TransferStart	97
12.2.1.396 EventStream0TransferStartFrameID	97
12.2.1.397 EventStream0TransferStartTimestamp	98
12.2.1.398 EventTest	98
12.2.1.399 EventTestTimestamp	98
12.2.1.400 EventTimer0End	98
12.2.1.401 EventTimer0EndFrameID	98
12.2.1.402 EventTimer0EndTimestamp	98
12.2.1.403 EventTimer0Start	98
12.2.1.404 EventTimer0StartFrameID	98
12.2.1.405 EventTimer0StartTimestamp	99
12.2.1.406 EventTimer1End	99
12.2.1.407 EventTimer1EndFrameID	99
12.2.1.408 EventTimer1EndTimestamp	99
12.2.1.409 EventTimer1Start	99
12.2.1.410 EventTimer1StartFrameID	99
12.2.1.411 EventTimer1StartTimestamp	99
12.2.1.412 ExposureActiveMode	99
12.2.1.413 ExposureAuto	100
12.2.1.414 ExposureMode	100
12.2.1.415 ExposureTime	100
12.2.1.416 ExposureTimeMode	100

12.2.1.417 ExposureTimeSelector	100
12.2.1.418 FactoryReset	100
12.2.1.419 FileAccessBuffer	100
12.2.1.420 FileAccessLength	100
12.2.1.421 FileAccessOffset	101
12.2.1.422 FileOpenMode	101
12.2.1.423 FileOperationExecute	101
12.2.1.424 FileOperationResult	101
12.2.1.425 FileOperationSelector	101
12.2.1.426 FileOperationStatus	101
12.2.1.427 FileSelector	101
12.2.1.428 FileSize	101
12.2.1.429 Gain	102
12.2.1.430 GainAuto	102
12.2.1.431 GainAutoBalance	102
12.2.1.432 GainSelector	102
12.2.1.433 Gamma	102
12.2.1.434 GammaEnable	102
12.2.1.435 GevActiveLinkCount	102
12.2.1.436 GevCCP	102
12.2.1.437 GevCurrentDefaultGateway	103
12.2.1.438 GevCurrentIPAddress	103
12.2.1.439 GevCurrentIPConfigurationDHCP	103
12.2.1.440 GevCurrentIPConfigurationLLA	103
12.2.1.441 GevCurrentIPConfigurationPersistentIP	103
12.2.1.442 GevCurrentPhysicalLinkConfiguration	103
12.2.1.443 GevCurrentSubnetMask	103
12.2.1.444 GevDiscoveryAckDelay	103
12.2.1.445 GevFirstURL	104
12.2.1.446 GevGVCPExtendedStatusCodes	104
12.2.1.447 GevGVCPExtendedStatusCodesSelector	104
12.2.1.448 GevGVCPHeartbeatDisable	104
12.2.1.449 GevGVCPPendingAck	104
12.2.1.450 GevGVCPPendingTimeout	104
12.2.1.451 GevGVSPExtendedIDMode	104
12.2.1.452 GevHeartbeatTimeout	104
12.2.1.453 GevIEEE1588	105
12.2.1.454 GevIEEE1588ClockAccuracy	105
12.2.1.455 GevIEEE1588Mode	105
12.2.1.456 GevIEEE1588Status	105
12.2.1.457 GevInterfaceSelector	105
12.2.1.458 GevIPConfigurationStatus	105

12.2.1.459	GevMACAddress	105
12.2.1.460	GevMCDA	105
12.2.1.461	GevMCPHostPort	106
12.2.1.462	GevMCRC	106
12.2.1.463	GevMCSP	106
12.2.1.464	GevMCTT	106
12.2.1.465	GevNumberOfInterfaces	106
12.2.1.466	GevPAUSEFrameReception	106
12.2.1.467	GevPAUSEFrameTransmission	106
12.2.1.468	GevPersistentDefaultGateway	106
12.2.1.469	GevPersistentIPAddress	107
12.2.1.470	GevPersistentSubnetMask	107
12.2.1.471	GevPhysicalLinkConfiguration	107
12.2.1.472	GevPrimaryApplicationIPAddress	107
12.2.1.473	GevPrimaryApplicationSocket	107
12.2.1.474	GevPrimaryApplicationSwitchoverKey	107
12.2.1.475	GevSCCFGAllInTransmission	107
12.2.1.476	GevSCCFGExtendedChunkData	107
12.2.1.477	GevSCCFGPacketResendDestination	108
12.2.1.478	GevSCCFGUnconditionalStreaming	108
12.2.1.479	GevSCDA	108
12.2.1.480	GevSCPD	108
12.2.1.481	GevSCPDDirection	108
12.2.1.482	GevSCPHostPort	108
12.2.1.483	GevSCPInterfaceIndex	108
12.2.1.484	GevSCPSBigEndian	108
12.2.1.485	GevSCPSDoNotFragment	109
12.2.1.486	GevSCPSFireTestPacket	109
12.2.1.487	GevSCPSPacketSize	109
12.2.1.488	GevSCSP	109
12.2.1.489	GevSCZoneConfigurationLock	109
12.2.1.490	GevSCZoneCount	109
12.2.1.491	GevSCZoneDirectionAll	109
12.2.1.492	GevSecondURL	109
12.2.1.493	GevStreamChannelSelector	110
12.2.1.494	GevSupportedOption	110
12.2.1.495	GevSupportedOptionSelector	110
12.2.1.496	GevTimestampTickFrequency	110
12.2.1.497	GuiXmlManifestAddress	110
12.2.1.498	Height	110
12.2.1.499	HeightMax	110
12.2.1.500	ImageComponentEnable	110

12.2.1.501 ImageComponentSelector	111
12.2.1.502 ImageCompressionBitrate	111
12.2.1.503 ImageCompressionJPEGFormatOption	111
12.2.1.504 ImageCompressionMode	111
12.2.1.505 ImageCompressionQuality	111
12.2.1.506 ImageCompressionRateOption	111
12.2.1.507 IspEnable	111
12.2.1.508 LineFilterWidth	111
12.2.1.509 LineFormat	112
12.2.1.510 LineInputFilterSelector	112
12.2.1.511 LineInverter	112
12.2.1.512 LineMode	112
12.2.1.513 LinePitch	112
12.2.1.514 LineSelector	112
12.2.1.515 LineSource	112
12.2.1.516 LineStatus	112
12.2.1.517 LineStatusAll	113
12.2.1.518 LinkErrorCount	113
12.2.1.519 LinkUptime	113
12.2.1.520 LogicBlockLUTInputActivation	113
12.2.1.521 LogicBlockLUTInputSelector	113
12.2.1.522 LogicBlockLUTInputSource	113
12.2.1.523 LogicBlockLUTOutputValue	113
12.2.1.524 LogicBlockLUTOutputValueAll	113
12.2.1.525 LogicBlockLUTRowIndex	114
12.2.1.526 LogicBlockLUTSelector	114
12.2.1.527 LogicBlockSelector	114
12.2.1.528 LUTEnable	114
12.2.1.529 LUTIndex	114
12.2.1.530 LUTSelector	114
12.2.1.531 LUTValue	114
12.2.1.532 LUTValueAll	114
12.2.1.533 MaxDeviceResetTime	115
12.2.1.534 OffsetX	115
12.2.1.535 OffsetY	115
12.2.1.536 PacketResendRequestCount	115
12.2.1.537 PayloadSize	115
12.2.1.538 PixelColorFilter	115
12.2.1.539 PixelDynamicRangeMax	115
12.2.1.540 PixelDynamicRangeMin	115
12.2.1.541 PixelFormat	116
12.2.1.542 PixelFormatInFold	116

12.2.1.543 PixelFormatInfoSelector	116
12.2.1.544 PixelSize	116
12.2.1.545 PowerSupplyCurrent	116
12.2.1.546 PowerSupplyVoltage	116
12.2.1.547 RegionDestination	116
12.2.1.548 RegionMode	116
12.2.1.549 RegionSelector	117
12.2.1.550 ReverseX	117
12.2.1.551 ReverseY	117
12.2.1.552 RgbTransformLightSource	117
12.2.1.553 Saturation	117
12.2.1.554 SaturationEnable	117
12.2.1.555 Scan3dAxisMax	117
12.2.1.556 Scan3dAxisMin	117
12.2.1.557 Scan3dCoordinateOffset	118
12.2.1.558 Scan3dCoordinateReferenceSelector	118
12.2.1.559 Scan3dCoordinateReferenceValue	118
12.2.1.560 Scan3dCoordinateScale	118
12.2.1.561 Scan3dCoordinateSelector	118
12.2.1.562 Scan3dCoordinateSystem	118
12.2.1.563 Scan3dCoordinateSystemReference	118
12.2.1.564 Scan3dCoordinateTransformSelector	118
12.2.1.565 Scan3dDistanceUnit	119
12.2.1.566 Scan3dInvalidDataFlag	119
12.2.1.567 Scan3dInvalidDataValue	119
12.2.1.568 Scan3dOutputMode	119
12.2.1.569 Scan3dTransformValue	119
12.2.1.570 SensorDescription	119
12.2.1.571 SensorDigitizationTaps	119
12.2.1.572 SensorHeight	119
12.2.1.573 SensorShutterMode	120
12.2.1.574 SensorTaps	120
12.2.1.575 SensorWidth	120
12.2.1.576 SequencerConfigurationMode	120
12.2.1.577 SequencerConfigurationValid	120
12.2.1.578 SequencerFeatureEnable	120
12.2.1.579 SequencerMode	120
12.2.1.580 SequencerPathSelector	120
12.2.1.581 SequencerSetActive	121
12.2.1.582 SequencerSetLoad	121
12.2.1.583 SequencerSetNext	121
12.2.1.584 SequencerSetSave	121

12.2.1.585 SequencerSetSelector	121
12.2.1.586 SequencerSetStart	121
12.2.1.587 SequencerSetValid	121
12.2.1.588 SequencerTriggerActivation	121
12.2.1.589 SequencerTriggerSource	122
12.2.1.590 SerialPortBaudRate	122
12.2.1.591 SerialPortDataBits	122
12.2.1.592 SerialPortParity	122
12.2.1.593 SerialPortSelector	122
12.2.1.594 SerialPortSource	122
12.2.1.595 SerialPortStopBits	122
12.2.1.596 SerialReceiveFramingErrorCount	122
12.2.1.597 SerialReceiveParityErrorCount	123
12.2.1.598 SerialReceiveQueueClear	123
12.2.1.599 SerialReceiveQueueCurrentCharacterCount	123
12.2.1.600 SerialReceiveQueueMaxCharacterCount	123
12.2.1.601 SerialTransmitQueueCurrentCharacterCount	123
12.2.1.602 SerialTransmitQueueMaxCharacterCount	123
12.2.1.603 Sharpening	123
12.2.1.604 SharpeningAuto	123
12.2.1.605 SharpeningEnable	124
12.2.1.606 SharpeningThreshold	124
12.2.1.607 SoftwareSignalPulse	124
12.2.1.608 SoftwareSignalSelector	124
12.2.1.609 SourceCount	124
12.2.1.610 SourceSelector	124
12.2.1.611 Test0001	124
12.2.1.612 TestEventGenerate	124
12.2.1.613 TestPattern	125
12.2.1.614 TestPatternGeneratorSelector	125
12.2.1.615 TestPendingAck	125
12.2.1.616 TimerDelay	125
12.2.1.617 TimerDuration	125
12.2.1.618 TimerReset	125
12.2.1.619 TimerSelector	125
12.2.1.620 TimerStatus	125
12.2.1.621 TimerTriggerActivation	126
12.2.1.622 TimerTriggerSource	126
12.2.1.623 TimerValue	126
12.2.1.624 Timestamp	126
12.2.1.625 TimestampLatch	126
12.2.1.626 TimestampLatchValue	126

12.2.1.627 TimestampReset	126
12.2.1.628 TLParamsLocked	126
12.2.1.629 TransferAbort	127
12.2.1.630 TransferBlockCount	127
12.2.1.631 TransferBurstCount	127
12.2.1.632 TransferComponentSelector	127
12.2.1.633 TransferControlMode	127
12.2.1.634 TransferOperationMode	127
12.2.1.635 TransferPause	127
12.2.1.636 TransferQueueCurrentBlockCount	127
12.2.1.637 TransferQueueMaxBlockCount	128
12.2.1.638 TransferQueueMode	128
12.2.1.639 TransferQueueOverflowCount	128
12.2.1.640 TransferResume	128
12.2.1.641 TransferSelector	128
12.2.1.642 TransferStart	128
12.2.1.643 TransferStatus	128
12.2.1.644 TransferStatusSelector	128
12.2.1.645 TransferStop	129
12.2.1.646 TransferStreamChannel	129
12.2.1.647 TransferTriggerActivation	129
12.2.1.648 TransferTriggerMode	129
12.2.1.649 TransferTriggerSelector	129
12.2.1.650 TransferTriggerSource	129
12.2.1.651 TriggerActivation	129
12.2.1.652 TriggerDelay	129
12.2.1.653 TriggerDivider	130
12.2.1.654 TriggerEventTest	130
12.2.1.655 TriggerMode	130
12.2.1.656 TriggerMultiplier	130
12.2.1.657 TriggerOverlap	130
12.2.1.658 TriggerSelector	130
12.2.1.659 TriggerSoftware	130
12.2.1.660 TriggerSource	130
12.2.1.661 UserOutputSelector	131
12.2.1.662 UserOutputValue	131
12.2.1.663 UserOutputValueAll	131
12.2.1.664 UserOutputValueAllMask	131
12.2.1.665 UserSetDefault	131
12.2.1.666 UserSetFeatureEnable	131
12.2.1.667 UserSetLoad	131
12.2.1.668 UserSetSave	131

12.2.1.669 UserSetSelector	132
12.2.1.670 V3_3Enable	132
12.2.1.671 WhiteClip	132
12.2.1.672 WhiteClipSelector	132
12.2.1.673 Width	132
12.2.1.674 WidthMax	132
12.3 quickSpinTLDevice Struct Reference	133
12.3.1 Field Documentation	133
12.3.1.1 DeviceAccessStatus	134
12.3.1.2 DeviceBootloaderVersion	134
12.3.1.3 DeviceCurrentSpeed	134
12.3.1.4 DeviceDisplayName	134
12.3.1.5 DeviceDriverVersion	134
12.3.1.6 DeviceEndianessMechanism	134
12.3.1.7 DeviceID	134
12.3.1.8 DeviceInstanceld	134
12.3.1.9 DeviceIsUpdater	135
12.3.1.10 DeviceLinkSpeed	135
12.3.1.11 DeviceLocation	135
12.3.1.12 DeviceModelName	135
12.3.1.13 DeviceMulticastMonitorMode	135
12.3.1.14 DevicePortId	135
12.3.1.15 DeviceReset	135
12.3.1.16 DeviceSerialNumber	135
12.3.1.17 DeviceType	136
12.3.1.18 DeviceU3VProtocol	136
12.3.1.19 DeviceUserID	136
12.3.1.20 DeviceVendorName	136
12.3.1.21 DeviceVersion	136
12.3.1.22 GenICamXMLLocation	136
12.3.1.23 GenICamXMLPath	136
12.3.1.24 GevCCP	136
12.3.1.25 GevDeviceAutoForceIP	137
12.3.1.26 GevDeviceDiscoverMaximumPacketSize	137
12.3.1.27 GevDeviceForceGateway	137
12.3.1.28 GevDeviceForceIP	137
12.3.1.29 GevDeviceForceIPAddress	137
12.3.1.30 GevDeviceForceSubnetMask	137
12.3.1.31 GevDeviceGateway	137
12.3.1.32 GevDeviceIPAddress	137
12.3.1.33 GevDeviceIsWrongSubnet	138
12.3.1.34 GevDeviceMACAddress	138

12.3.1.35	GevDeviceMaximumPacketSize	138
12.3.1.36	GevDeviceMaximumRetryCount	138
12.3.1.37	GevDeviceModelsBigEndian	138
12.3.1.38	GevDevicePort	138
12.3.1.39	GevDeviceReadAndWriteTimeout	138
12.3.1.40	GevDeviceSubnetMask	138
12.3.1.41	GevVersionMajor	139
12.3.1.42	GevVersionMinor	139
12.3.1.43	GUIXMLLocation	139
12.3.1.44	GUIXMLPath	139
12.4	quickSpinTLInterface Struct Reference	139
12.4.1	Field Documentation	140
12.4.1.1	ActionCommand	140
12.4.1.2	DeviceAccessStatus	140
12.4.1.3	DeviceCount	141
12.4.1.4	DeviceID	141
12.4.1.5	DeviceModelName	141
12.4.1.6	DeviceSelector	141
12.4.1.7	DeviceSerialNumber	141
12.4.1.8	DeviceUnlock	141
12.4.1.9	DeviceUpdateList	141
12.4.1.10	DeviceVendorName	141
12.4.1.11	FilterDriverStatus	142
12.4.1.12	GevActionDeviceKey	142
12.4.1.13	GevActionGroupKey	142
12.4.1.14	GevActionGroupMask	142
12.4.1.15	GevActionTime	142
12.4.1.16	GevDeviceAutoForceIP	142
12.4.1.17	GevDeviceForceGateway	142
12.4.1.18	GevDeviceForceIP	142
12.4.1.19	GevDeviceForceIPAddress	143
12.4.1.20	GevDeviceForceSubnetMask	143
12.4.1.21	GevDeviceGateway	143
12.4.1.22	GevDeviceIPAddress	143
12.4.1.23	GevDeviceMACAddress	143
12.4.1.24	GevDeviceSubnetMask	143
12.4.1.25	GevInterfaceGateway	143
12.4.1.26	GevInterfaceGatewaySelector	143
12.4.1.27	GevInterfaceMACAddress	144
12.4.1.28	GevInterfaceMTU	144
12.4.1.29	GevInterfaceReceiveLinkSpeed	144
12.4.1.30	GevInterfaceSubnetIPAddress	144

12.4.1.31	GevInterfaceSubnetMask	144
12.4.1.32	GevInterfaceSubnetSelector	144
12.4.1.33	GevInterfaceTransmitLinkSpeed	144
12.4.1.34	HostAdapterDriverVersion	144
12.4.1.35	HostAdapterName	145
12.4.1.36	HostAdapterVendor	145
12.4.1.37	IncompatibleDeviceCount	145
12.4.1.38	IncompatibleDeviceID	145
12.4.1.39	IncompatibleDeviceModelName	145
12.4.1.40	IncompatibleDeviceSelector	145
12.4.1.41	IncompatibleDeviceVendorName	145
12.4.1.42	IncompatibleGevDeviceIPAddress	145
12.4.1.43	IncompatibleGevDeviceMACAddress	146
12.4.1.44	IncompatibleGevDeviceSubnetMask	146
12.4.1.45	InterfaceDisplayName	146
12.4.1.46	InterfaceID	146
12.4.1.47	InterfaceType	146
12.4.1.48	POEStatus	146
12.5	quickSpinTLStream Struct Reference	147
12.5.1	Field Documentation	147
12.5.1.1	StreamAnnounceBufferMinimum	147
12.5.1.2	StreamAnnouncedBufferCount	147
12.5.1.3	StreamBlockTransferSize	148
12.5.1.4	StreamBufferAlignment	148
12.5.1.5	StreamBufferCountManual	148
12.5.1.6	StreamBufferCountMax	148
12.5.1.7	StreamBufferCountMode	148
12.5.1.8	StreamBufferCountResult	148
12.5.1.9	StreamBufferHandlingMode	148
12.5.1.10	StreamChunkCountMaximum	148
12.5.1.11	StreamCRCCheckEnable	149
12.5.1.12	StreamDeliveredFrameCount	149
12.5.1.13	StreamDroppedFrameCount	149
12.5.1.14	StreamID	149
12.5.1.15	StreamIncompleteFrameCount	149
12.5.1.16	StreamInputBufferCount	149
12.5.1.17	StreamIsGrabbing	149
12.5.1.18	StreamLostFrameCount	149
12.5.1.19	StreamMissedPacketCount	150
12.5.1.20	StreamMode	150
12.5.1.21	StreamOutputBufferCount	150
12.5.1.22	StreamPacketResendEnable	150

12.5.1.23 StreamPacketResendMaxRequests	150
12.5.1.24 StreamPacketResendReceivedPacketCount	150
12.5.1.25 StreamPacketResendRequestCount	150
12.5.1.26 StreamPacketResendRequestedPacketCount	150
12.5.1.27 StreamPacketResendRequestSuccessCount	151
12.5.1.28 StreamPacketResendTimeout	151
12.5.1.29 StreamReceivedFrameCount	151
12.5.1.30 StreamReceivedPacketCount	151
12.5.1.31 StreamStartedFrameCount	151
12.5.1.32 StreamType	151
12.6 quickSpinTLSystem Struct Reference	152
12.6.1 Field Documentation	152
12.6.1.1 EnumerateGen2Cameras	152
12.6.1.2 EnumerateGEVInterfaces	152
12.6.1.3 EnumerateUSBInterfaces	153
12.6.1.4 GenTLFNCVersionMajor	153
12.6.1.5 GenTLFNCVersionMinor	153
12.6.1.6 GenTLFNCVersionSubMinor	153
12.6.1.7 GenTLVersionMajor	153
12.6.1.8 GenTLVersionMinor	153
12.6.1.9 GevInterfaceDefaultGateway	153
12.6.1.10 GevInterfaceDefaultIPAddress	153
12.6.1.11 GevInterfaceDefaultSubnetMask	154
12.6.1.12 GevInterfaceMACAddress	154
12.6.1.13 GevVersionMajor	154
12.6.1.14 GevVersionMinor	154
12.6.1.15 InterfaceDisplayName	154
12.6.1.16 InterfaceID	154
12.6.1.17 InterfaceSelector	154
12.6.1.18 InterfaceUpdateList	154
12.6.1.19 TLDisplayName	155
12.6.1.20 TLFileName	155
12.6.1.21 TLID	155
12.6.1.22 TLModelName	155
12.6.1.23 TLPPath	155
12.6.1.24 TLType	155
12.6.1.25 TLVendorName	155
12.6.1.26 TLVersion	156
12.7 spinAVIOption Struct Reference	156
12.7.1 Detailed Description	156
12.7.2 Field Documentation	156
12.7.2.1 frameRate	156

12.7.2.2 height	156
12.7.2.3 reserved	157
12.7.2.4 width	157
12.8 spinBMPOption Struct Reference	157
12.8.1 Detailed Description	157
12.8.2 Field Documentation	157
12.8.2.1 indexedColor_8bit	157
12.8.2.2 reserved	158
12.9 spinChunkData Struct Reference	158
12.9.1 Detailed Description	159
12.9.2 Field Documentation	159
12.9.2.1 m_blackLevel	159
12.9.2.2 m_compressionMode	159
12.9.2.3 m_compressionRatio	159
12.9.2.4 m_counterValue	159
12.9.2.5 m_cRC	159
12.9.2.6 m_encoderValue	159
12.9.2.7 m_exposureEndLineStatusAll	160
12.9.2.8 m_exposureTime	160
12.9.2.9 m_frameID	160
12.9.2.10 m_gain	160
12.9.2.11 m_height	160
12.9.2.12 m_image	160
12.9.2.13 m_inferenceConfidence	160
12.9.2.14 m_inferenceFrameId	160
12.9.2.15 m_inferenceResult	161
12.9.2.16 m_linePitch	161
12.9.2.17 m_lineStatusAll	161
12.9.2.18 m_offsetX	161
12.9.2.19 m_offsetY	161
12.9.2.20 m_partSelector	161
12.9.2.21 m_pixelDynamicRangeMax	161
12.9.2.22 m_pixelDynamicRangeMin	161
12.9.2.23 m_scan3dAxisMax	162
12.9.2.24 m_scan3dAxisMin	162
12.9.2.25 m_scan3dCoordinateOffset	162
12.9.2.26 m_scan3dCoordinateReferenceValue	162
12.9.2.27 m_scan3dCoordinateScale	162
12.9.2.28 m_scan3dInvalidDataValue	162
12.9.2.29 m_scan3dTransformValue	162
12.9.2.30 m_scanLineSelector	162
12.9.2.31 m_sequencerSetActive	163

12.9.2.32 m_serialDataLength	163
12.9.2.33 m_streamChannelID	163
12.9.2.34 m_timerValue	163
12.9.2.35 m_timestamp	163
12.9.2.36 m_timestampLatchValue	163
12.9.2.37 m_transferBlockID	163
12.9.2.38 m_transferQueueCurrentBlockCount	163
12.9.2.39 m_width	164
12.10 spinH264Option Struct Reference	164
12.10.1 Detailed Description	164
12.10.2 Field Documentation	164
12.10.2.1 bitrate	164
12.10.2.2 frameRate	165
12.10.2.3 height	165
12.10.2.4 reserved	165
12.10.2.5 width	165
12.11 spinJPEGOption Struct Reference	165
12.11.1 Detailed Description	166
12.11.2 Field Documentation	166
12.11.2.1 progressive	166
12.11.2.2 quality	166
12.11.2.3 reserved	166
12.12 spinJPG2Option Struct Reference	166
12.12.1 Detailed Description	167
12.12.2 Field Documentation	167
12.12.2.1 quality	167
12.12.2.2 reserved	167
12.13 spinLibraryVersion Struct Reference	167
12.13.1 Detailed Description	168
12.13.2 Field Documentation	168
12.13.2.1 build	168
12.13.2.2 major	168
12.13.2.3 minor	168
12.13.2.4 type	168
12.14 spinMJPGOption Struct Reference	168
12.14.1 Detailed Description	169
12.14.2 Field Documentation	169
12.14.2.1 frameRate	169
12.14.2.2 height	169
12.14.2.3 quality	169
12.14.2.4 reserved	170
12.14.2.5 width	170

12.15 spinPGMOption Struct Reference	170
12.15.1 Detailed Description	170
12.15.2 Field Documentation	170
12.15.2.1 binaryFile	170
12.15.2.2 reserved	171
12.16 spinPNGOption Struct Reference	171
12.16.1 Detailed Description	171
12.16.2 Field Documentation	171
12.16.2.1 compressionLevel	171
12.16.2.2 interlaced	171
12.16.2.3 reserved	172
12.17 spinPPMOption Struct Reference	172
12.17.1 Detailed Description	172
12.17.2 Field Documentation	172
12.17.2.1 binaryFile	172
12.17.2.2 reserved	172
12.18 spinTIFFOption Struct Reference	173
12.18.1 Detailed Description	173
12.18.2 Field Documentation	173
12.18.2.1 compression	173
12.18.2.2 reserved	173
13 File Documentation	175
13.1 doc/spindocs/C/GettingStarted.dox File Reference	175
13.2 doc/spindocs/C/ProgrammerGuide.dox File Reference	175
13.3 doc/spindocs/shared/Benefits.dox File Reference	175
13.4 doc/spindocs/shared/FlyCapture2Comparison.dox File Reference	175
13.5 doc/spindocs/shared/GenICamGenTL.dox File Reference	175
13.6 doc/spindocs/shared/Licensing.dox File Reference	175
13.7 doc/spindocs/shared/Maintenance.dox File Reference	175
13.8 include/spinc/CameraDefsC.h File Reference	175
13.8.1 Enumeration Type Documentation	208
13.8.1.1 spinAcquisitionModeEnums	208
13.8.1.2 spinAcquisitionStatusSelectorEnums	208
13.8.1.3 spinActionUnconditionalModeEnums	209
13.8.1.4 spinAdcBitDepthEnums	209
13.8.1.5 spinAutoAlgorithmSelectorEnums	209
13.8.1.6 spinAutoExposureControlPriorityEnums	210
13.8.1.7 spinAutoExposureLightingModeEnums	210
13.8.1.8 spinAutoExposureMeteringModeEnums	210
13.8.1.9 spinAutoExposureTargetGreyValueAutoEnums	211
13.8.1.10 spinBalanceRatioSelectorEnums	211

13.8.1.11 spinBalanceWhiteAutoEnums	211
13.8.1.12 spinBalanceWhiteAutoProfileEnums	212
13.8.1.13 spinBinningHorizontalModeEnums	212
13.8.1.14 spinBinningSelectorEnums	213
13.8.1.15 spinBinningVerticalModeEnums	213
13.8.1.16 spinBlackLevelAutoBalanceEnums	213
13.8.1.17 spinBlackLevelAutoEnums	214
13.8.1.18 spinBlackLevelSelectorEnums	214
13.8.1.19 spinChunkBlackLevelSelectorEnums	214
13.8.1.20 spinChunkCounterSelectorEnums	214
13.8.1.21 spinChunkEncoderSelectorEnums	215
13.8.1.22 spinChunkEncoderStatusEnums	215
13.8.1.23 spinChunkExposureTimeSelectorEnums	215
13.8.1.24 spinChunkGainSelectorEnums	216
13.8.1.25 spinChunkImageComponentEnums	216
13.8.1.26 spinChunkPixelFormatEnums	217
13.8.1.27 spinChunkRegionIDEnums	217
13.8.1.28 spinChunkScan3dCoordinateReferenceSelectorEnums	217
13.8.1.29 spinChunkScan3dCoordinateSelectorEnums	218
13.8.1.30 spinChunkScan3dCoordinateSystemEnums	218
13.8.1.31 spinChunkScan3dCoordinateSystemReferenceEnums	218
13.8.1.32 spinChunkScan3dCoordinateTransformSelectorEnums	219
13.8.1.33 spinChunkScan3dDistanceUnitEnums	219
13.8.1.34 spinChunkScan3dOutputModeEnums	220
13.8.1.35 spinChunkSelectorEnums	221
13.8.1.36 spinChunkSourceIDEnums	221
13.8.1.37 spinChunkTimerSelectorEnums	221
13.8.1.38 spinChunkTransferStreamIDEnums	222
13.8.1.39 spinCIConfigurationEnums	222
13.8.1.40 spinCITimeSlotsCountEnums	223
13.8.1.41 spinColorTransformationSelectorEnums	223
13.8.1.42 spinColorTransformationValueSelectorEnums	223
13.8.1.43 spinCompressionSaturationPriorityEnums	224
13.8.1.44 spinCounterEventActivationEnums	224
13.8.1.45 spinCounterEventSourceEnums	224
13.8.1.46 spinCounterResetActivationEnums	225
13.8.1.47 spinCounterResetSourceEnums	225
13.8.1.48 spinCounterSelectorEnums	226
13.8.1.49 spinCounterStatusEnums	226
13.8.1.50 spinCounterTriggerActivationEnums	227
13.8.1.51 spinCounterTriggerSourceEnums	227
13.8.1.52 spinCxpConnectionTestModeEnums	228

13.8.1.53 spinCxpLinkConfigurationEnums	228
13.8.1.54 spinCxpLinkConfigurationPreferredEnums	229
13.8.1.55 spinCxpLinkConfigurationStatusEnums	230
13.8.1.56 spinCxpPoCxpStatusEnums	231
13.8.1.57 spinDecimationHorizontalModeEnums	231
13.8.1.58 spinDecimationSelectorEnums	231
13.8.1.59 spinDecimationVerticalModeEnums	232
13.8.1.60 spinDefectCorrectionModeEnums	232
13.8.1.61 spinDeinterlacingEnums	232
13.8.1.62 spinDeviceCharacterSetEnums	233
13.8.1.63 spinDeviceClockSelectorEnums	233
13.8.1.64 spinDeviceConnectionStatusEnums	233
13.8.1.65 spinDeviceIndicatorModeEnums	234
13.8.1.66 spinDeviceLinkHeartbeatModeEnums	234
13.8.1.67 spinDeviceLinkThroughputLimitModeEnums	234
13.8.1.68 spinDevicePowerSupplySelectorEnums	235
13.8.1.69 spinDeviceRegistersEndiannessEnums	235
13.8.1.70 spinDeviceScanTypeEnums	235
13.8.1.71 spinDeviceSerialPortBaudRateEnums	235
13.8.1.72 spinDeviceSerialPortSelectorEnums	236
13.8.1.73 spinDeviceStreamChannelEndiannessEnums	236
13.8.1.74 spinDeviceStreamChannelTypeEnums	236
13.8.1.75 spinDeviceTapGeometryEnums	237
13.8.1.76 spinDeviceTemperatureSelectorEnums	238
13.8.1.77 spinDeviceTLTypeEnums	238
13.8.1.78 spinDeviceTypeEnums	239
13.8.1.79 spinEncoderModeEnums	239
13.8.1.80 spinEncoderOutputModeEnums	239
13.8.1.81 spinEncoderResetActivationEnums	240
13.8.1.82 spinEncoderResetSourceEnums	240
13.8.1.83 spinEncoderSelectorEnums	241
13.8.1.84 spinEncoderSourceAEnums	242
13.8.1.85 spinEncoderSourceBEnums	242
13.8.1.86 spinEncoderStatusEnums	242
13.8.1.87 spinEventNotificationEnums	243
13.8.1.88 spinEventSelectorEnums	243
13.8.1.89 spinExposureActiveModeEnums	243
13.8.1.90 spinExposureAutoEnums	244
13.8.1.91 spinExposureModeEnums	244
13.8.1.92 spinExposureTimeModeEnums	244
13.8.1.93 spinExposureTimeSelectorEnums	245
13.8.1.94 spinFileOpenModeEnums	245

13.8.1.95 spinFileOperationSelectorEnums	245
13.8.1.96 spinFileOperationStatusEnums	246
13.8.1.97 spinFileSelectorEnums	246
13.8.1.98 spinGainAutoBalanceEnums	246
13.8.1.99 spinGainAutoEnums	248
13.8.1.100 spinGainSelectorEnums	248
13.8.1.101 spinGevCCPEnums	248
13.8.1.102 spinGevCurrentPhysicalLinkConfigurationEnums	249
13.8.1.103 spinGevGVCPExtendedStatusCodesSelectorEnums	249
13.8.1.104 spinGevGVSPExtendedIDModeEnums	249
13.8.1.105 spinGevIEEE1588ClockAccuracyEnums	250
13.8.1.106 spinGevIEEE1588ModeEnums	250
13.8.1.107 spinGevIEEE1588StatusEnums	250
13.8.1.108 spinGevIPConfigurationStatusEnums	251
13.8.1.109 spinGevPhysicalLinkConfigurationEnums	251
13.8.1.110 spinGevSupportedOptionSelectorEnums	252
13.8.1.111 spinImageComponentSelectorEnums	252
13.8.1.112 spinImageCompressionJPEGFormatOptionEnums	253
13.8.1.113 spinImageCompressionModeEnums	253
13.8.1.114 spinImageCompressionRateOptionEnums	254
13.8.1.115 spinLineFormatEnums	254
13.8.1.116 spinLineInputFilterSelectorEnums	254
13.8.1.117 spinLineModeEnums	255
13.8.1.118 spinLineSelectorEnums	255
13.8.1.119 spinLineSourceEnums	255
13.8.1.120 spinLogicBlockLUTInputActivationEnums	256
13.8.1.121 spinLogicBlockLUTInputSelectorEnums	256
13.8.1.122 spinLogicBlockLUTInputSourceEnums	257
13.8.1.123 spinLogicBlockLUTSelectorEnums	257
13.8.1.124 spinLogicBlockSelectorEnums	258
13.8.1.125 spinLUTSelectorEnums	258
13.8.1.126 spinPixelColorFilterEnums	258
13.8.1.127 spinPixelFormatEnums	259
13.8.1.128 spinPixelFormatInfoSelectorEnums	264
13.8.1.129 spinPixelSizeEnums	270
13.8.1.130 spinRegionDestinationEnums	271
13.8.1.131 spinRegionModeEnums	271
13.8.1.132 spinRegionSelectorEnums	271
13.8.1.133 spinRgbTransformLightSourceEnums	272
13.8.1.134 spinScan3dCoordinateReferenceSelectorEnums	272
13.8.1.135 spinScan3dCoordinateSelectorEnums	273
13.8.1.136 spinScan3dCoordinateSystemEnums	273

13.8.1.137 spinScan3dCoordinateSystemReferenceEnums	273
13.8.1.138 spinScan3dCoordinateTransformSelectorEnums	274
13.8.1.139 spinScan3dDistanceUnitEnums	274
13.8.1.140 spinScan3dOutputModeEnums	274
13.8.1.141 spinSensorDigitizationTapsEnums	276
13.8.1.142 spinSensorShutterModeEnums	277
13.8.1.143 spinSensorTapsEnums	277
13.8.1.144 spinSequencerConfigurationModeEnums	278
13.8.1.145 spinSequencerConfigurationValidEnums	278
13.8.1.146 spinSequencerModeEnums	278
13.8.1.147 spinSequencerSetValidEnums	278
13.8.1.148 spinSequencerTriggerActivationEnums	279
13.8.1.149 spinSequencerTriggerSourceEnums	279
13.8.1.150 spinSerialPortBaudRateEnums	279
13.8.1.151 spinSerialPortParityEnums	280
13.8.1.152 spinSerialPortSelectorEnums	280
13.8.1.153 spinSerialPortSourceEnums	281
13.8.1.154 spinSerialPortStopBitsEnums	281
13.8.1.155 spinSoftwareSignalSelectorEnums	281
13.8.1.156 spinSourceSelectorEnums	282
13.8.1.157 spinTestPatternEnums	282
13.8.1.158 spinTestPatternGeneratorSelectorEnums	282
13.8.1.159 spinTimerSelectorEnums	283
13.8.1.160 spinTimerStatusEnums	283
13.8.1.161 spinTimerTriggerActivationEnums	283
13.8.1.162 spinTimerTriggerSourceEnums	284
13.8.1.163 spinTransferComponentSelectorEnums	285
13.8.1.164 spinTransferControlModeEnums	285
13.8.1.165 spinTransferOperationModeEnums	286
13.8.1.166 spinTransferQueueModeEnums	286
13.8.1.167 spinTransferSelectorEnums	286
13.8.1.168 spinTransferStatusSelectorEnums	287
13.8.1.169 spinTransferTriggerActivationEnums	287
13.8.1.170 spinTransferTriggerModeEnums	287
13.8.1.171 spinTransferTriggerSelectorEnums	288
13.8.1.172 spinTransferTriggerSourceEnums	288
13.8.1.173 spinTriggerActivationEnums	289
13.8.1.174 spinTriggerModeEnums	290
13.8.1.175 spinTriggerOverlapEnums	290
13.8.1.176 spinTriggerSelectorEnums	290
13.8.1.177 spinTriggerSourceEnums	291
13.8.1.178 spinUserOutputSelectorEnums	291

13.8.1.179 spinUserSetDefaultEnums	291
13.8.1.180 spinUserSetSelectorEnums	292
13.8.1.181 spinWhiteClipSelectorEnums	292
13.9 include/spinc/ChunkDataDefC.h File Reference	293
13.10 include/spinc/QuickSpinC.h File Reference	294
13.10.1 Function Documentation	294
13.10.1.1 quickSpinInit()	294
13.10.1.2 quickSpinInitEx()	295
13.10.1.3 quickSpinTLDeviceInit()	295
13.10.1.4 quickSpinTLInterfaceInit()	295
13.10.1.5 quickSpinTLStreamInit()	295
13.10.1.6 quickSpinTLSystemInit()	295
13.11 include/spinc/QuickSpinDefsC.h File Reference	296
13.11.1 Typedef Documentation	296
13.11.1.1 quickSpinBooleanNode	297
13.11.1.2 quickSpinCommandNode	297
13.11.1.3 quickSpinEnumerationNode	297
13.11.1.4 quickSpinFloatNode	297
13.11.1.5 quickSpinIntegerNode	297
13.11.1.6 quickSpinRegisterNode	297
13.11.1.7 quickSpinStringNode	297
13.12 include/spinc/SpinnakerC.h File Reference	298
13.12.1 Function Documentation	307
13.12.1.1 spinCameraBeginAcquisition()	308
13.12.1.2 spinCameraDelInit()	308
13.12.1.3 spinCameraDiscoverMaxPacketSize()	308
13.12.1.4 spinCameraEndAcquisition()	309
13.12.1.5 spinCameraForceIP()	309
13.12.1.6 spinCameraGetAccessMode()	310
13.12.1.7 spinCameraGetGuiXml()	310
13.12.1.8 spinCameraGetNextImage()	311
13.12.1.9 spinCameraGetNextImageEx()	311
13.12.1.10 spinCameraGetNextImageSync()	312
13.12.1.11 spinCameraGetNodeMap()	312
13.12.1.12 spinCameraGetTLDeviceNodeMap()	313
13.12.1.13 spinCameraGetTLStreamNodeMap()	313
13.12.1.14 spinCameraGetUniqueID()	314
13.12.1.15 spinCameraInit()	314
13.12.1.16 spinCameraIsInitialized()	315
13.12.1.17 spinCameraIsStreaming()	315
13.12.1.18 spinCameraIsValid()	315
13.12.1.19 spinCameraListAppend()	316

13.12.1.20 spinCameraListClear()	316
13.12.1.21 spinCameraListCreateEmpty()	317
13.12.1.22 spinCameraListDestroy()	317
13.12.1.23 spinCameraListGet()	318
13.12.1.24 spinCameraListGetBySerial()	318
13.12.1.25 spinCameraListGetSize()	319
13.12.1.26 spinCameraListRemove()	319
13.12.1.27 spinCameraListRemoveBySerial()	320
13.12.1.28 spinCameraReadPort()	320
13.12.1.29 spinCameraRegisterDeviceEventHandler()	321
13.12.1.30 spinCameraRegisterDeviceEventHandlerEx()	321
13.12.1.31 spinCameraRegisterImageEventHandler()	322
13.12.1.32 spinCameraRegisterImageEventHandlerEx()	322
13.12.1.33 spinCameraRegisterImageListEventHandler()	323
13.12.1.34 spinCameraRelease()	323
13.12.1.35 spinCameraUnregisterDeviceEventHandler()	323
13.12.1.36 spinCameraUnregisterImageEventHandler()	324
13.12.1.37 spinCameraUnregisterImageListEventHandler()	324
13.12.1.38 spinCameraWritePort()	325
13.12.1.39 spinDeviceArrivalEventHandlerCreate()	325
13.12.1.40 spinDeviceArrivalEventHandlerDestroy()	326
13.12.1.41 spinDeviceEventGetId()	326
13.12.1.42 spinDeviceEventGetName()	326
13.12.1.43 spinDeviceEventGetPayloadData()	327
13.12.1.44 spinDeviceEventGetPayloadDataSize()	328
13.12.1.45 spinDeviceEventHandlerCreate()	328
13.12.1.46 spinDeviceEventHandlerDestroy()	329
13.12.1.47 spinDeviceRemovalEventHandlerCreate()	329
13.12.1.48 spinDeviceRemovalEventHandlerDestroy()	330
13.12.1.49 spinErrorGetLast()	330
13.12.1.50 spinErrorGetLastBuildDate()	330
13.12.1.51 spinErrorGetLastBuildTime()	331
13.12.1.52 spinErrorGetLastFileName()	331
13.12.1.53 spinErrorGetLastFullMessage()	332
13.12.1.54 spinErrorGetLastFunctionName()	332
13.12.1.55 spinErrorGetLastLineNumber()	333
13.12.1.56 spinErrorGetLastMessage()	333
13.12.1.57 spinImageCalculateStatistics()	334
13.12.1.58 spinImageCheckCRC()	334
13.12.1.59 spinImageChunkDataGetFloatValue()	335
13.12.1.60 spinImageChunkDataGetIntValue()	335
13.12.1.61 spinImageCreate()	335

13.12.1.62 spinImageCreateEmpty()	336
13.12.1.63 spinImageCreateEx()	336
13.12.1.64 spinImageCreateEx2()	337
13.12.1.65 spinImageDeepCopy()	337
13.12.1.66 spinImageDestroy()	338
13.12.1.67 spinImageEventHandlerCreate()	338
13.12.1.68 spinImageEventHandlerDestroy()	339
13.12.1.69 spinImageGetBitsPerPixel()	339
13.12.1.70 spinImageGetBufferSize()	340
13.12.1.71 spinImageGetChunkLayoutID()	340
13.12.1.72 spinImageGetColorProcessing()	341
13.12.1.73 spinImageGetData()	341
13.12.1.74 spinImageGetFrameID()	342
13.12.1.75 spinImageGetHeight()	342
13.12.1.76 spinImageGetID()	343
13.12.1.77 spinImageGetOffsetX()	343
13.12.1.78 spinImageGetOffsetY()	344
13.12.1.79 spinImageGetPaddingX()	344
13.12.1.80 spinImageGetPaddingY()	345
13.12.1.81 spinImageGetPayloadType()	345
13.12.1.82 spinImageGetPixelFormat()	346
13.12.1.83 spinImageGetPixelFormatName()	346
13.12.1.84 spinImageGetPrivateData()	347
13.12.1.85 spinImageGetSize()	347
13.12.1.86 spinImageGetStatus()	348
13.12.1.87 spinImageGetStatusDescription()	348
13.12.1.88 spinImageGetStride()	349
13.12.1.89 spinImageGetTimeStamp()	349
13.12.1.90 spinImageGetTLPayloadType()	350
13.12.1.91 spinImageGetTLPixelFormat()	350
13.12.1.92 spinImageGetTLPixelFormatNamespace()	351
13.12.1.93 spinImageGetValidPayloadSize()	351
13.12.1.94 spinImageGetWidth()	352
13.12.1.95 spinImageHasCRC()	352
13.12.1.96 spinImageIsIncomplete()	353
13.12.1.97 spinImageListAppend()	353
13.12.1.98 spinImageListClear()	354
13.12.1.99 spinImageListCreateEmpty()	354
13.12.1.100 spinImageListDestroy()	355
13.12.1.101 spinImageListEventHandlerCreate()	355
13.12.1.102 spinImageListEventHandlerDestroy()	356
13.12.1.103 spinImageListGet()	356

13.12.1.104 spinImageListGetByPixelFormat()	357
13.12.1.105 spinImageListGetSize()	357
13.12.1.106 spinImageListLoad()	358
13.12.1.107 spinImageListRelease()	358
13.12.1.108 spinImageListRemove()	358
13.12.1.109 spinImageListRemoveByPixelFormat()	359
13.12.1.110 spinImageListSave()	359
13.12.1.111 spinImageProcessorApplyGamma()	360
13.12.1.112 spinImageProcessorConvert()	360
13.12.1.113 spinImageProcessorConvertImageList()	361
13.12.1.114 spinImageProcessorCreate()	362
13.12.1.115 spinImageProcessorDestroy()	362
13.12.1.116 spinImageProcessorGetColorProcessing()	363
13.12.1.117 spinImageProcessorGetNumDecompressionThreads()	363
13.12.1.118 spinImageProcessorSetColorProcessing()	363
13.12.1.119 spinImageProcessorSetNumDecompressionThreads()	364
13.12.1.120 spinImageRelease()	364
13.12.1.121 spinImageReset()	365
13.12.1.122 spinImageResetEx()	365
13.12.1.123 spinImageSave()	366
13.12.1.124 spinImageSaveBmp()	367
13.12.1.125 spinImageSaveFromExt()	367
13.12.1.126 spinImageSaveJpeg()	368
13.12.1.127 spinImageSaveJpg2()	368
13.12.1.128 spinImageSavePgm()	369
13.12.1.129 spinImageSavePng()	369
13.12.1.130 spinImageSavePpm()	370
13.12.1.131 spinImageSaveTiff()	370
13.12.1.132 spinImageStatisticsCreate()	371
13.12.1.133 spinImageStatisticsDestroy()	371
13.12.1.134 spinImageStatisticsDisableAll()	371
13.12.1.135 spinImageStatisticsEnableAll()	372
13.12.1.136 spinImageStatisticsEnableGreyOnly()	372
13.12.1.137 spinImageStatisticsEnableHslOnly()	373
13.12.1.138 spinImageStatisticsEnableRgbOnly()	373
13.12.1.139 spinImageStatisticsGetAll()	374
13.12.1.140 spinImageStatisticsGetChannelStatus()	374
13.12.1.141 spinImageStatisticsGetHistogram()	375
13.12.1.142 spinImageStatisticsGetMean()	375
13.12.1.143 spinImageStatisticsGetNumPixelValues()	376
13.12.1.144 spinImageStatisticsGetPixelValueRange()	376
13.12.1.145 spinImageStatisticsGetRange()	377

13.12.1.146 spinImageStatisticsSetChannelStatus()	377
13.12.1.147 spinInterfaceEventHandlerCreate()	378
13.12.1.148 spinInterfaceEventHandlerDestroy()	378
13.12.1.149 spinInterfaceGetCameras()	379
13.12.1.150 spinInterfaceGetCamerasEx()	379
13.12.1.151 spinInterfaceGetTLNodeMap()	380
13.12.1.152 spinInterfaceIsInUse()	380
13.12.1.153 spinInterfaceListClear()	381
13.12.1.154 spinInterfaceListCreateEmpty()	381
13.12.1.155 spinInterfaceListDestroy()	382
13.12.1.156 spinInterfaceListGet()	382
13.12.1.157 spinInterfaceListGetSize()	383
13.12.1.158 spinInterfaceRegisterDeviceArrivalEventHandler()	383
13.12.1.159 spinInterfaceRegisterDeviceRemovalEventHandler()	384
13.12.1.160 spinInterfaceRegisterInterfaceEventHandler()	384
13.12.1.161 spinInterfaceRelease()	385
13.12.1.162 spinInterfaceSendActionCommand()	385
13.12.1.163 spinInterfaceUnregisterDeviceArrivalEventHandler()	386
13.12.1.164 spinInterfaceUnregisterDeviceRemovalEventHandler()	386
13.12.1.165 spinInterfaceUnregisterInterfaceEventHandler()	387
13.12.1.166 spinInterfaceUpdateCameras()	387
13.12.1.167 spinLogDataGetCategoryName()	388
13.12.1.168 spinLogDataGetLogMessage()	388
13.12.1.169 spinLogDataGetNDC()	389
13.12.1.170 spinLogDataGetPriority()	389
13.12.1.171 spinLogDataGetPriorityName()	390
13.12.1.172 spinLogDataGetThreadName()	390
13.12.1.173 spinLogDataGetTimestamp()	391
13.12.1.174 spinLogEventHandlerCreate()	391
13.12.1.175 spinLogEventHandlerDestroy()	392
13.12.1.176 spinSystemGetCameras()	392
13.12.1.177 spinSystemGetCamerasEx()	393
13.12.1.178 spinSystemGetInstance()	393
13.12.1.179 spinSystemGetInterfaces()	394
13.12.1.180 spinSystemGetLibraryVersion()	394
13.12.1.181 spinSystemGetLoggingLevel()	395
13.12.1.182 spinSystemGetTLNodeMap()	395
13.12.1.183 spinSystemIsInUse()	396
13.12.1.184 spinSystemRegisterDeviceArrivalEventHandler()	396
13.12.1.185 spinSystemRegisterDeviceRemovalEventHandler()	397
13.12.1.186 spinSystemRegisterInterfaceEventHandler()	397
13.12.1.187 spinSystemRegisterLogEventHandler()	398

13.12.1.188 spinSystemReleaseInstance()	398
13.12.1.189 spinSystemSendActionCommand()	398
13.12.1.190 spinSystemSetLoggingLevel()	399
13.12.1.191 spinSystemUnregisterAllLogEventHandlers()	400
13.12.1.192 spinSystemUnregisterDeviceArrivalEventHandler()	400
13.12.1.193 spinSystemUnregisterDeviceRemovalEventHandler()	401
13.12.1.194 spinSystemUnregisterInterfaceEventHandler()	401
13.12.1.195 spinSystemUnregisterLogEventHandler()	402
13.12.1.196 spinSystemUpdateCameras()	402
13.12.1.197 spinSystemUpdateCamerasEx()	403
13.13 include/spinc/SpinnakerDefsC.h File Reference	403
13.13.1 Typedef Documentation	408
13.13.1.1 bool8_t	408
13.13.1.2 spinArrivalEventFunction	408
13.13.1.3 spinCamera	408
13.13.1.4 spinCameraList	409
13.13.1.5 spinDeviceArrivalEventHandler	409
13.13.1.6 spinDeviceEventData	409
13.13.1.7 spinDeviceEventFunction	409
13.13.1.8 spinDeviceEventHandler	409
13.13.1.9 spinDeviceRemovalEventHandler	409
13.13.1.10 spinImage	410
13.13.1.11 spinImageEventFunction	410
13.13.1.12 spinImageEventHandler	410
13.13.1.13 spinImageList	410
13.13.1.14 spinImageListEventFunction	410
13.13.1.15 spinImageListEventHandler	410
13.13.1.16 spinImageProcessor	411
13.13.1.17 spinImageStatistics	411
13.13.1.18 spinInterface	411
13.13.1.19 spinInterfaceEventHandler	411
13.13.1.20 spinInterfaceList	411
13.13.1.21 spinLogEventData	411
13.13.1.22 spinLogEventFunction	412
13.13.1.23 spinLogEventHandler	412
13.13.1.24 spinRemovalEventFunction	412
13.13.1.25 spinSystem	412
13.13.1.26 spinVideo	412
13.13.2 Enumeration Type Documentation	412
13.13.2.1 spinActionCommandStatus	412
13.13.2.2 spinColorProcessingAlgorithm	413
13.13.2.3 spinError	413

13.13.2.4 spinImageFileFormat	415
13.13.2.5 spinImageStatus	415
13.13.2.6 spinnakerLogLevel	416
13.13.2.7 spinStatisticsChannel	416
13.13.2.8 spinTIFFCompressionMethod	417
13.13.2.9 spinTLPayloadType	417
13.13.2.10 spinTLPixelFormatNamespace	418
13.13.3 Variable Documentation	418
13.13.3.1 False	418
13.13.3.2 True	418
13.14 include/spinc/SpinnakerGenApiC.h File Reference	419
13.14.1 Function Documentation	423
13.14.1.1 spinBooleanGetValue()	423
13.14.1.2 spinBooleanSetValue()	423
13.14.1.3 spinCategoryGetFeatureByIndex()	424
13.14.1.4 spinCategoryGetNumFeatures()	424
13.14.1.5 spinCategoryReleaseNode()	425
13.14.1.6 spinCommandExecute()	425
13.14.1.7 spinCommandIsDone()	426
13.14.1.8 spinEnumerationEntryGetEnumValue()	426
13.14.1.9 spinEnumerationEntryGetIntValue()	427
13.14.1.10 spinEnumerationEntryGetSymbolic()	427
13.14.1.11 spinEnumerationGetCurrentEntry()	428
13.14.1.12 spinEnumerationGetEntryByIndex()	428
13.14.1.13 spinEnumerationGetEntryByName()	429
13.14.1.14 spinEnumerationGetNumEntries()	429
13.14.1.15 spinEnumerationReleaseNode()	430
13.14.1.16 spinEnumerationSetEnumValue()	430
13.14.1.17 spinEnumerationSetIntValue()	431
13.14.1.18 spinFloatGetMax()	431
13.14.1.19 spinFloatGetMin()	432
13.14.1.20 spinFloatGetRepresentation()	432
13.14.1.21 spinFloatGetUnit()	433
13.14.1.22 spinFloatGetValue()	433
13.14.1.23 spinFloatGetValueEx()	434
13.14.1.24 spinFloatSetValue()	434
13.14.1.25 spinFloatSetValueEx()	435
13.14.1.26 spinIntegerGetInc()	435
13.14.1.27 spinIntegerGetMax()	436
13.14.1.28 spinIntegerGetMin()	436
13.14.1.29 spinIntegerGetRepresentation()	437
13.14.1.30 spinIntegerGetValue()	437

13.14.1.31 spinIntegerGetValueEx()	438
13.14.1.32 spinIntegerSetValue()	438
13.14.1.33 spinIntegerSetValueEx()	439
13.14.1.34 spinNodeDeregisterCallback()	439
13.14.1.35 spinNodeFromString()	440
13.14.1.36 spinNodeFromStringEx()	440
13.14.1.37 spinNodeGetAccessMode()	441
13.14.1.38 spinNodeGetCachingMode()	441
13.14.1.39 spinNodeGetDescription()	442
13.14.1.40 spinNodeGetDisplayName()	442
13.14.1.41 spinNodeGetImposedAccessMode()	443
13.14.1.42 spinNodeGetImposedVisibility()	443
13.14.1.43 spinNodeGetName()	444
13.14.1.44 spinNodeGetNameSpace()	444
13.14.1.45 spinNodeGetPollingTime()	445
13.14.1.46 spinNodeGetToolTip()	445
13.14.1.47 spinNodeGetType()	446
13.14.1.48 spinNodeGetVisibility()	446
13.14.1.49 spinNodeInvalidateNode()	447
13.14.1.50 spinNodeIsAvailable()	447
13.14.1.51 spinNodeIsEqual()	448
13.14.1.52 spinNodeIsImplemented()	448
13.14.1.53 spinNodeIsReadable()	449
13.14.1.54 spinNodeIsWritable()	449
13.14.1.55 spinNodeMapGetNode()	450
13.14.1.56 spinNodeMapGetNodeByIndex()	450
13.14.1.57 spinNodeMapGetNumNodes()	451
13.14.1.58 spinNodeMapPoll()	451
13.14.1.59 spinNodeMapReleaseNode()	452
13.14.1.60 spinNodeRegisterCallback()	452
13.14.1.61 spinNodeToString()	453
13.14.1.62 spinNodeToStringEx()	453
13.14.1.63 spinRegisterGet()	454
13.14.1.64 spinRegisterGetAddress()	454
13.14.1.65 spinRegisterGetEx()	455
13.14.1.66 spinRegisterGetLength()	455
13.14.1.67 spinRegisterSet()	456
13.14.1.68 spinRegisterSetEx()	456
13.14.1.69 spinRegisterSetReference()	457
13.14.1.70 spinStringGetMaxLength()	457
13.14.1.71 spinStringGetValue()	458
13.14.1.72 spinStringGetValueEx()	458

13.14.1.73 spinStringSetValue()	459
13.14.1.74 spinStringSetValueEx()	459
13.15 include/spinc/SpinnakerGenApiDefsC.h File Reference	460
13.15.1 Typedef Documentation	463
13.15.1.1 spinNodeCallbackFunction	463
13.15.1.2 spinNodeCallbackHandle	463
13.15.1.3 spinNodeHandle	463
13.15.1.4 spinNodeMapHandle	463
13.15.2 Enumeration Type Documentation	463
13.15.2.1 spinAccessMode	463
13.15.2.2 spinCachingMode	464
13.15.2.3 spinDisplayNotation	464
13.15.2.4 spinEndianess	465
13.15.2.5 spinIncMode	465
13.15.2.6 spinInputDirection	465
13.15.2.7 spinInterfaceType	466
13.15.2.8 spinLinkType	466
13.15.2.9 spinNameSpace	467
13.15.2.10 spinNodeType	467
13.15.2.11 spinRepresentation	468
13.15.2.12 spinSign	468
13.15.2.13 spinSlope	468
13.15.2.14 spinStandardNameSpace	469
13.15.2.15 spinVisibility	469
13.15.2.16 spinXMLValidation	470
13.15.2.17 spinYesNo	470
13.16 include/spinc/SpinnakerPlatformC.h File Reference	471
13.16.1 Macro Definition Documentation	471
13.16.1.1 SPINNAKERC_API	471
13.17 include/spinc/SpinVideoC.h File Reference	472
13.17.1 Function Documentation	472
13.17.1.1 spinVideoAppend()	472
13.17.1.2 spinVideoClose()	473
13.17.1.3 spinVideoOpenH264()	473
13.17.1.4 spinVideoOpenMJPEG()	473
13.17.1.5 spinVideoOpenUncompressed()	473
13.17.1.6 spinVideoSetMaximumFileSize()	473
13.18 include/spinc/TransportLayerDefsC.h File Reference	474
13.18.1 Enumeration Type Documentation	476
13.18.1.1 spinTLDeviceAccessStatusEnums	476
13.18.1.2 spinTLDeviceCurrentSpeedEnums	476
13.18.1.3 spinTLDeviceEndianessMechanismEnums	476

13.18.1.4 spinTLDeviceTypeEnums	478
13.18.1.5 spinTLFilterDriverStatusEnums	478
13.18.1.6 spinTLGenlCamXMLLocationEnums	478
13.18.1.7 spinTLGevCCPEnums	479
13.18.1.8 spinTLGUIXMLLocationEnums	479
13.18.1.9 spinTLInterfaceTypeEnums	479
13.18.1.10 spinTLPOEStatusEnums	480
13.18.1.11 spinTLStreamBufferCountModeEnums	480
13.18.1.12 spinTLStreamBufferHandlingModeEnums	480
13.18.1.13 spinTLStreamModeEnums	481
13.18.1.14 spinTLStreamTypeEnums	481
13.18.1.15 spinTLTLTypeEnums	482
13.19 include/spinc/TransportLayerDeviceC.h File Reference	482
13.20 include/spinc/TransportLayerInterfaceC.h File Reference	483
13.21 include/spinc/TransportLayerStreamC.h File Reference	484
13.22 include/spinc/TransportLayerSystemC.h File Reference	484
Index	487

Chapter 1

Getting Started

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

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

Chapter 2

Programmer's Guide

Chapter 3

Benefits of Spinnaker

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

Chapter 4

FlyCapture2 Feature Comparison with Spinnaker

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

Chapter 5

Working with GenICam GenTL Devices

5.1 GenTL Overview

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

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

5.2 Installation

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

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

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

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

5.3 Troubleshooting

5.3.1 Enable FLIR GenTL Logging

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

Sample log4cpp.gentl.property configuration file:

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

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

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

5.3.2 USB3 Device Image Tearing

Image tearing could occur with certain USB3 host controllers when streaming with a GenTL producer. To work around the issue, make sure the size of each buffer announced to the FLIR GenTL producer is 1024 bytes aligned. The size of each buffer should be $(\text{bufferSize} + 1024 - 1) / 1024 * 1024$ 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	<p>Copyright (c) 2001-2020 FLIR Systems, Inc. All Rights Reserved.</p> <p>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).</p> <p>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.</p>
GenICam	GenICam License
AdapterList	The Code Project Open License (CPOL)
Make ListView.ScrollIntoView Scroll the Item into the Center of the ListView	WP:CC_BY-SA License
Work with Bitmaps Faster in C#	The Code Project Open License (CPOL) 1.02
FreeImage	FreeImage public license
Boost	Boost Software License
Libusb	LGPLv2.1 License
Libraw1394	LGPLv2.0 License
FFMPEG	LGPLv2.1 License
log4Net	Apache license 2.0
log4Cpp	LGPL License

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

Chapter 7

Software Maintenance Policy

7.1 GenTL Overview

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

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

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

7.2 Platform Support Policy

7.2.1 Windows Support

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

7.2.2 Linux Desktop Support

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

7.2.3 Linux Embedded Support

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

7.2.4 MacOS Support

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

7.3 Versioning Policy

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

MAJOR.MINOR.0.PATCH

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

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

Chapter 8

Module Index

8.1 Modules

Here is a list of all modules:

Camera Enumerations	22
Chunk Data Structures	22
Spinnaker C QuickSpin API	22
TLDevice Structures	32
TLInterface Structures	32
TLStream Structures	33
TLSystem Structures	33
QuickSpin Access	22
Spinnaker C API	23
Spinnaker C Definitions	21
Error Handling	23
System Access	23
InterfaceList Access	24
CameraList Access	24
ImageList Access	24
Interface Access	24
Camera Access	24
Image Access	25
Image Processor Access	25
Event Access	28
ImageStatistics Access	28
Logging Event Data Access	28
Device Event Data Access	28
Chunk data access	28
Spinnaker C Handles	29
Spinnaker C Function Signatures	29
Spinnaker C Enumerations	29
Spinnaker C Structures	29
Spinnaker C GenICam API	29
Node Map Access	29
Node Access	29
IValue Access	30
String Access	30
Integer Access	30
IFloat Access	30

IEnumeration Access	30
IEnumEntry Access	31
IBoolean Access	31
ICommand Access	31
ICategory Access	31
IRegister Access	31
Spinnaker C GenICam Handles	31
Spinnaker C GenICam Enumerations	32
SpinVideo Recording Access	32
Transport Layer Enumerations	32

Chapter 9

Data Structure Index

9.1 Data Structures

Here are the data structures with brief descriptions:

actionCommandResult	
Action Command Result	35
quickSpin	36
quickSpinTLDevice	133
quickSpinTLInterface	139
quickSpinTLStream	147
quickSpinTLSystem	152
spinAVIOption	
Options for saving uncompressed videos	156
spinBMPOption	
Options for saving BMP images	157
spinChunkData	
The type of information that can be obtained from image chunk data	158
spinH264Option	
Options for saving H264 videos	164
spinJPEGOption	
Options for saving JPEG images	165
spinJPG2Option	
Options for saving JPEG 2000 images	166
spinLibraryVersion	
Provides easier access to the current version of Spinnaker	167
spinMJPGOption	
Options for saving MJPG videos	168
spinPGMOption	
Options for saving PGM images	170
spinPNGOption	
Options for saving PNG images	171
spinPPMOption	
Options for saving PPM images	172
spinTIFFOption	
Options for saving TIFF images	173

Chapter 10

File Index

10.1 File List

Here is a list of all files with brief descriptions:

include/spinc/ CameraDefsC.h	175
include/spinc/ ChunkDataDefC.h	293
include/spinc/ QuickSpinC.h	294
include/spinc/ QuickSpinDefsC.h	296
include/spinc/ SpinnakerC.h	298
include/spinc/ SpinnakerDefsC.h	403
include/spinc/ SpinnakerGenApiC.h	419
include/spinc/ SpinnakerGenApiDefsC.h	460
include/spinc/ SpinnakerPlatformC.h	471
include/spinc/ SpinVideoC.h	472
include/spinc/ TransportLayerDefsC.h	474
include/spinc/ TransportLayerDeviceC.h	482
include/spinc/ TransportLayerInterfaceC.h	483
include/spinc/ TransportLayerStreamC.h	484
include/spinc/ TransportLayerSystemC.h	484

Chapter 11

Module Documentation

11.1 Spinnaker C Definitions

Definitions for Spinnaker C.

Collaboration diagram for Spinnaker C Definitions:



Definitions for Spinnaker C.

Definitions for Spinnaker C API.

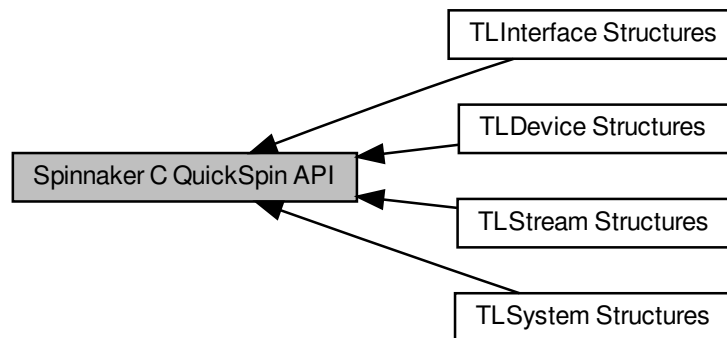
Holds enumerations, typedefs and structures that are used across the Spinnaker C API wrapper.

11.2 Camera Enumerations

11.3 Chunk Data Structures

11.4 Spinnaker C QuickSpin API

Collaboration diagram for Spinnaker C QuickSpin API:



Modules

- [TLDevice Structures](#)
- [TLInterface Structures](#)
- [TLStream Structures](#)
- [TLSystem Structures](#)

11.4.1 Detailed Description

11.5 QuickSpin Access

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

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

11.6 Spinnaker C API

SpinnakerPlatform C Include.

Collaboration diagram for Spinnaker C API:



Modules

- [Spinnaker C Definitions](#)
Definitions for Spinnaker C.

11.6.1 Detailed Description

SpinnakerPlatform C Include.

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

Spinnaker C Definition Includes

11.7 Error Handling

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

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

11.8 System Access

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

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

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

11.9 InterfaceList Access

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

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

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

11.10 CameraList Access

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

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

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

11.11 ImageList Access

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

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

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

11.12 Interface Access

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

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

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

11.13 Camera Access

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

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

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

11.14 Image Access

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

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

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

11.15 Image Processor Access

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

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

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

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

List of supported input image pixel formats:

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

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

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

List of supported output image pixel formats

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

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

11.16 Event Access

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

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

11.17 ImageStatistics Access

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

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

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

11.18 Logging Event Data Access

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

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

11.19 Device Event Data Access

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

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

11.20 Chunk data access

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

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

11.21 Spinnaker C Handles

Spinnaker C handle definitions.

Spinnaker C handle definitions.

11.22 Spinnaker C Function Signatures

Spinnaker C function signature definitions.

Spinnaker C function signature definitions.

11.23 Spinnaker C Enumerations

Spinnaker C enumeration definitions.

Spinnaker C enumeration definitions.

11.24 Spinnaker C Structures

Spinnaker C structure definitions.

Spinnaker C structure definitions.

11.25 Spinnaker C GenICam API

11.26 Node Map Access

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

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

This includes nodes, node counts, and polling.

11.27 Node Access

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

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

This includes node properties and callback registration.

11.28 IValue Access

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

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

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

11.29 String Access

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

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

This includes getters and setters of values and value lengths.

11.30 Integer Access

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

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

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

11.31 IFloat Access

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

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

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

11.32 IEnumeration Access

The functions in this section provide access to enum nodes.

The functions in this section provide access to enum nodes.

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

11.33 IEnumEntry Access

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

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

11.34 IBoolean Access

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

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

This includes value getters and setters.

11.35 ICommand Access

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

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

This includes node properties and callbacks.

11.36 ICategory Access

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

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

This includes node properties and callbacks.

11.37 IRegister Access

The functions in this section provide access to register nodes.

The functions in this section provide access to register nodes.

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

11.38 Spinnaker C GenICam Handles

Handle definitions for Spinnaker C GenICam API.

Handle definitions for Spinnaker C GenICam API.

11.39 Spinnaker C GenICam Enumerations

Enumeration definitions for Spinnaker C GenICam API.

Enumeration definitions for Spinnaker C GenICam API.

11.40 SpinVideo Recording Access

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

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

11.41 Transport Layer Enumerations

11.42 TLDevice Structures

Collaboration diagram for TLDevice Structures:



11.43 TLInterface Structures

Collaboration diagram for TLInterface Structures:



11.44 TLStream Structures

Collaboration diagram for TLStream Structures:



11.45 TLSystem Structures

Collaboration diagram for TLSystem Structures:



Chapter 12

Data Structure Documentation

12.1 actionCommandResult Struct Reference

Action Command Result.

Data Fields

- unsigned int [DeviceAddress](#)
- [spinActionCommandStatus](#) Status

12.1.1 Detailed Description

Action Command Result.

12.1.2 Field Documentation

12.1.2.1 DeviceAddress

```
unsigned int DeviceAddress
```

12.1.2.2 Status

```
spinActionCommandStatus Status
```

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

- include/spinc/[SpinnakerDefsC.h](#)

12.2 quickSpin Struct Reference

Data Fields

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

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

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

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

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

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

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

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

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

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

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

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

- [quickSpinIntegerNode aPAUSEMACCtrlFramesTransmitted](#)
- [quickSpinIntegerNode aPAUSEMACCtrlFramesReceived](#)
- [quickSpinEnumerationNode 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 DeviceUserID
- quickSpinStringNode DeviceVendorName
- quickSpinStringNode DeviceModelName
- quickSpinStringNode DeviceVersion
- quickSpinIntegerNode DeviceBootloaderVersion
- quickSpinEnumerationNode DeviceType
- quickSpinStringNode DeviceDisplayName
- quickSpinEnumerationNode DeviceAccessStatus
- quickSpinIntegerNode DeviceLinkSpeed
- quickSpinStringNode DeviceDriverVersion
- quickSpinBooleanNode DeviceIsUpdater
- quickSpinEnumerationNode GenICamXMLLocation
- quickSpinStringNode GenICamXMLPath
- quickSpinEnumerationNode GUIXMLLocation
- quickSpinStringNode GUIXMLPath
- quickSpinEnumerationNode GevCCP
- quickSpinIntegerNode GevDeviceMACAddress
- quickSpinIntegerNode GevDeviceIPAddress
- quickSpinIntegerNode GevDeviceSubnetMask
- quickSpinIntegerNode GevDeviceGateway
- quickSpinIntegerNode GevVersionMajor
- quickSpinIntegerNode GevVersionMinor
- quickSpinBooleanNode GevDeviceModelsBigEndian
- quickSpinIntegerNode GevDeviceReadAndWriteTimeout
- quickSpinIntegerNode GevDeviceMaximumRetryCount
- quickSpinIntegerNode GevDevicePort
- quickSpinCommandNode GevDeviceDiscoverMaximumPacketSize
- quickSpinIntegerNode GevDeviceMaximumPacketSize
- quickSpinBooleanNode GevDeviceIsWrongSubnet
- quickSpinCommandNode GevDeviceAutoForceIP
- quickSpinCommandNode GevDeviceForceIP
- quickSpinIntegerNode GevDeviceForceIPAddress
- quickSpinIntegerNode GevDeviceForceSubnetMask
- quickSpinIntegerNode GevDeviceForceGateway
- quickSpinBooleanNode DeviceMulticastMonitorMode
- quickSpinEnumerationNode DeviceEndiannessMechanism
- quickSpinCommandNode DeviceReset
- quickSpinStringNode DeviceInstanceId
- quickSpinStringNode DeviceLocation
- quickSpinEnumerationNode DeviceCurrentSpeed
- quickSpinBooleanNode DeviceU3VProtocol
- quickSpinStringNode DevicePortId

12.3.1 Field Documentation

12.3.1.1 DeviceAccessStatus

[quickSpinEnumerationNode](#) DeviceAccessStatus

12.3.1.2 DeviceBootloaderVersion

[quickSpinIntegerNode](#) DeviceBootloaderVersion

12.3.1.3 DeviceCurrentSpeed

[quickSpinEnumerationNode](#) DeviceCurrentSpeed

12.3.1.4 DeviceDisplayName

[quickSpinStringNode](#) DeviceDisplayName

12.3.1.5 DeviceDriverVersion

[quickSpinStringNode](#) DeviceDriverVersion

12.3.1.6 DeviceEndiannessMechanism

[quickSpinEnumerationNode](#) DeviceEndiannessMechanism

12.3.1.7 DeviceID

[quickSpinStringNode](#) DeviceID

12.3.1.8 DeviceInstanceId

[quickSpinStringNode](#) DeviceInstanceId

12.3.1.9 DeviceIsUpdater

`quickSpinBooleanNode` DeviceIsUpdater

12.3.1.10 DeviceLinkSpeed

`quickSpinIntegerNode` DeviceLinkSpeed

12.3.1.11 DeviceLocation

`quickSpinStringNode` DeviceLocation

12.3.1.12 DeviceModelName

`quickSpinStringNode` DeviceModelName

12.3.1.13 DeviceMulticastMonitorMode

`quickSpinBooleanNode` DeviceMulticastMonitorMode

12.3.1.14 DevicePortId

`quickSpinStringNode` DevicePortId

12.3.1.15 DeviceReset

`quickSpinCommandNode` DeviceReset

12.3.1.16 DeviceSerialNumber

`quickSpinStringNode` DeviceSerialNumber

12.3.1.17 DeviceType

[quickSpinEnumerationNode](#) DeviceType

12.3.1.18 DeviceU3VProtocol

[quickSpinBooleanNode](#) DeviceU3VProtocol

12.3.1.19 DeviceUserID

[quickSpinStringNode](#) DeviceUserID

12.3.1.20 DeviceVendorName

[quickSpinStringNode](#) DeviceVendorName

12.3.1.21 DeviceVersion

[quickSpinStringNode](#) DeviceVersion

12.3.1.22 GenICamXMLLocation

[quickSpinEnumerationNode](#) GenICamXMLLocation

12.3.1.23 GenICamXMLPath

[quickSpinStringNode](#) GenICamXMLPath

12.3.1.24 GevCCP

[quickSpinEnumerationNode](#) GevCCP

12.3.1.25 **GevDeviceAutoForceIP**

[quickSpinCommandNode](#) [GevDeviceAutoForceIP](#)

12.3.1.26 **GevDeviceDiscoverMaximumPacketSize**

[quickSpinCommandNode](#) [GevDeviceDiscoverMaximumPacketSize](#)

12.3.1.27 **GevDeviceForceGateway**

[quickSpinIntegerNode](#) [GevDeviceForceGateway](#)

12.3.1.28 **GevDeviceForceIP**

[quickSpinCommandNode](#) [GevDeviceForceIP](#)

12.3.1.29 **GevDeviceForceIPAddress**

[quickSpinIntegerNode](#) [GevDeviceForceIPAddress](#)

12.3.1.30 **GevDeviceForceSubnetMask**

[quickSpinIntegerNode](#) [GevDeviceForceSubnetMask](#)

12.3.1.31 **GevDeviceGateway**

[quickSpinIntegerNode](#) [GevDeviceGateway](#)

12.3.1.32 **GevDeviceIPAddress**

[quickSpinIntegerNode](#) [GevDeviceIPAddress](#)

12.3.1.33 **GevDeviceIsWrongSubnet**

`quickSpinBooleanNode` `GevDeviceIsWrongSubnet`

12.3.1.34 **GevDeviceMACAddress**

`quickSpinIntegerNode` `GevDeviceMACAddress`

12.3.1.35 **GevDeviceMaximumPacketSize**

`quickSpinIntegerNode` `GevDeviceMaximumPacketSize`

12.3.1.36 **GevDeviceMaximumRetryCount**

`quickSpinIntegerNode` `GevDeviceMaximumRetryCount`

12.3.1.37 **GevDeviceModelsBigEndian**

`quickSpinBooleanNode` `GevDeviceModeIsBigEndian`

12.3.1.38 **GevDevicePort**

`quickSpinIntegerNode` `GevDevicePort`

12.3.1.39 **GevDeviceReadAndWriteTimeout**

`quickSpinIntegerNode` `GevDeviceReadAndWriteTimeout`

12.3.1.40 **GevDeviceSubnetMask**

`quickSpinIntegerNode` `GevDeviceSubnetMask`

12.3.1.41 **GevVersionMajor**

[quickSpinIntegerNode](#) [GevVersionMajor](#)

12.3.1.42 **GevVersionMinor**

[quickSpinIntegerNode](#) [GevVersionMinor](#)

12.3.1.43 **GUIXMLLocation**

[quickSpinEnumerationNode](#) [GUIXMLLocation](#)

12.3.1.44 **GUIXMLPath**

[quickSpinStringNode](#) [GUIXMLPath](#)

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

- [include/spinc/TransportLayerDeviceC.h](#)

12.4 quickSpinTLInterface Struct Reference

Data Fields

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

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

12.4.1 Field Documentation

12.4.1.1 ActionCommand

[quickSpinCommandNode](#) `ActionCommand`

12.4.1.2 DeviceAccessStatus

[quickSpinEnumerationNode](#) `DeviceAccessStatus`

12.4.1.3 DeviceCount

`quickSpinIntegerNode` DeviceCount

12.4.1.4 DeviceID

`quickSpinStringNode` DeviceID

12.4.1.5 DeviceModelName

`quickSpinStringNode` DeviceModelName

12.4.1.6 DeviceSelector

`quickSpinIntegerNode` DeviceSelector

12.4.1.7 DeviceSerialNumber

`quickSpinStringNode` DeviceSerialNumber

12.4.1.8 DeviceUnlock

`quickSpinStringNode` DeviceUnlock

12.4.1.9 DeviceUpdateList

`quickSpinCommandNode` DeviceUpdateList

12.4.1.10 DeviceVendorName

`quickSpinStringNode` DeviceVendorName

12.4.1.11 FilterDriverStatus

`quickSpinEnumerationNode` FilterDriverStatus

12.4.1.12 GevActionDeviceKey

`quickSpinIntegerNode` GevActionDeviceKey

12.4.1.13 GevActionGroupKey

`quickSpinIntegerNode` GevActionGroupKey

12.4.1.14 GevActionGroupMask

`quickSpinIntegerNode` GevActionGroupMask

12.4.1.15 GevActionTime

`quickSpinIntegerNode` GevActionTime

12.4.1.16 GevDeviceAutoForceIP

`quickSpinCommandNode` GevDeviceAutoForceIP

12.4.1.17 GevDeviceForceGateway

`quickSpinIntegerNode` GevDeviceForceGateway

12.4.1.18 GevDeviceForceIP

`quickSpinCommandNode` GevDeviceForceIP

12.4.1.19 **GevDeviceForceIPAddress**

[quickSpinIntegerNode](#) `GevDeviceForceIPAddress`

12.4.1.20 **GevDeviceForceSubnetMask**

[quickSpinIntegerNode](#) `GevDeviceForceSubnetMask`

12.4.1.21 **GevDeviceGateway**

[quickSpinIntegerNode](#) `GevDeviceGateway`

12.4.1.22 **GevDeviceIPAddress**

[quickSpinIntegerNode](#) `GevDeviceIPAddress`

12.4.1.23 **GevDeviceMACAddress**

[quickSpinIntegerNode](#) `GevDeviceMACAddress`

12.4.1.24 **GevDeviceSubnetMask**

[quickSpinIntegerNode](#) `GevDeviceSubnetMask`

12.4.1.25 **GevInterfaceGateway**

[quickSpinIntegerNode](#) `GevInterfaceGateway`

12.4.1.26 **GevInterfaceGatewaySelector**

[quickSpinIntegerNode](#) `GevInterfaceGatewaySelector`

12.4.1.27 GevInterfaceMACAddress

`quickSpinIntegerNode` GevInterfaceMACAddress

12.4.1.28 GevInterfaceMTU

`quickSpinIntegerNode` GevInterfaceMTU

12.4.1.29 GevInterfaceReceiveLinkSpeed

`quickSpinIntegerNode` GevInterfaceReceiveLinkSpeed

12.4.1.30 GevInterfaceSubnetIPAddress

`quickSpinIntegerNode` GevInterfaceSubnetIPAddress

12.4.1.31 GevInterfaceSubnetMask

`quickSpinIntegerNode` GevInterfaceSubnetMask

12.4.1.32 GevInterfaceSubnetSelector

`quickSpinIntegerNode` GevInterfaceSubnetSelector

12.4.1.33 GevInterfaceTransmitLinkSpeed

`quickSpinIntegerNode` GevInterfaceTransmitLinkSpeed

12.4.1.34 HostAdapterDriverVersion

`quickSpinStringNode` HostAdapterDriverVersion

12.4.1.35 HostAdapterName

`quickSpinStringNode` HostAdapterName

12.4.1.36 HostAdapterVendor

`quickSpinStringNode` HostAdapterVendor

12.4.1.37 IncompatibleDeviceCount

`quickSpinIntegerNode` IncompatibleDeviceCount

12.4.1.38 IncompatibleDeviceID

`quickSpinStringNode` IncompatibleDeviceID

12.4.1.39 IncompatibleDeviceModelName

`quickSpinStringNode` IncompatibleDeviceModelName

12.4.1.40 IncompatibleDeviceSelector

`quickSpinIntegerNode` IncompatibleDeviceSelector

12.4.1.41 IncompatibleDeviceVendorName

`quickSpinStringNode` IncompatibleDeviceVendorName

12.4.1.42 IncompatibleGevDeviceIPAddress

`quickSpinIntegerNode` IncompatibleGevDeviceIPAddress

12.4.1.43 IncompatibleGevDeviceMACAddress

[quickSpinIntegerNode](#) IncompatibleGevDeviceMACAddress

12.4.1.44 IncompatibleGevDeviceSubnetMask

[quickSpinIntegerNode](#) IncompatibleGevDeviceSubnetMask

12.4.1.45 InterfaceDisplayName

[quickSpinStringNode](#) InterfaceDisplayName

12.4.1.46 InterfaceID

[quickSpinStringNode](#) InterfaceID

12.4.1.47 InterfaceType

[quickSpinEnumerationNode](#) InterfaceType

12.4.1.48 POEStatus

[quickSpinEnumerationNode](#) POEStatus

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

- [include/spinc/TransportLayerInterfaceC.h](#)

12.5 quickSpinTLStream Struct Reference

Data Fields

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

12.5.1 Field Documentation

12.5.1.1 StreamAnnounceBufferMinimum

[quickSpinIntegerNode](#) StreamAnnounceBufferMinimum

12.5.1.2 StreamAnnouncedBufferCount

[quickSpinIntegerNode](#) StreamAnnouncedBufferCount

12.5.1.3 StreamBlockTransferSize

[quickSpinIntegerNode](#) StreamBlockTransferSize

12.5.1.4 StreamBufferAlignment

[quickSpinIntegerNode](#) StreamBufferAlignment

12.5.1.5 StreamBufferCountManual

[quickSpinIntegerNode](#) StreamBufferCountManual

12.5.1.6 StreamBufferCountMax

[quickSpinIntegerNode](#) StreamBufferCountMax

12.5.1.7 StreamBufferCountMode

[quickSpinEnumerationNode](#) StreamBufferCountMode

12.5.1.8 StreamBufferCountResult

[quickSpinIntegerNode](#) StreamBufferCountResult

12.5.1.9 StreamBufferHandlingMode

[quickSpinEnumerationNode](#) StreamBufferHandlingMode

12.5.1.10 StreamChunkCountMaximum

[quickSpinIntegerNode](#) StreamChunkCountMaximum

12.5.1.11 StreamCRCCheckEnable

`quickSpinBooleanNode` StreamCRCCheckEnable

12.5.1.12 StreamDeliveredFrameCount

`quickSpinIntegerNode` StreamDeliveredFrameCount

12.5.1.13 StreamDroppedFrameCount

`quickSpinIntegerNode` StreamDroppedFrameCount

12.5.1.14 StreamID

`quickSpinStringNode` StreamID

12.5.1.15 StreamIncompleteFrameCount

`quickSpinIntegerNode` StreamIncompleteFrameCount

12.5.1.16 StreamInputBufferCount

`quickSpinIntegerNode` StreamInputBufferCount

12.5.1.17 StreamIsGrabbing

`quickSpinBooleanNode` StreamIsGrabbing

12.5.1.18 StreamLostFrameCount

`quickSpinIntegerNode` StreamLostFrameCount

12.5.1.19 StreamMissedPacketCount

`quickSpinIntegerNode` `StreamMissedPacketCount`

12.5.1.20 StreamMode

`quickSpinEnumerationNode` `StreamMode`

12.5.1.21 StreamOutputBufferCount

`quickSpinIntegerNode` `StreamOutputBufferCount`

12.5.1.22 StreamPacketResendEnable

`quickSpinBooleanNode` `StreamPacketResendEnable`

12.5.1.23 StreamPacketResendMaxRequests

`quickSpinIntegerNode` `StreamPacketResendMaxRequests`

12.5.1.24 StreamPacketResendReceivedPacketCount

`quickSpinIntegerNode` `StreamPacketResendReceivedPacketCount`

12.5.1.25 StreamPacketResendRequestCount

`quickSpinIntegerNode` `StreamPacketResendRequestCount`

12.5.1.26 StreamPacketResendRequestedPacketCount

`quickSpinIntegerNode` `StreamPacketResendRequestedPacketCount`

12.5.1.27 StreamPacketResendRequestSuccessCount

[quickSpinIntegerNode](#) StreamPacketResendRequestSuccessCount

12.5.1.28 StreamPacketResendTimeout

[quickSpinIntegerNode](#) StreamPacketResendTimeout

12.5.1.29 StreamReceivedFrameCount

[quickSpinIntegerNode](#) StreamReceivedFrameCount

12.5.1.30 StreamReceivedPacketCount

[quickSpinIntegerNode](#) StreamReceivedPacketCount

12.5.1.31 StreamStartedFrameCount

[quickSpinIntegerNode](#) StreamStartedFrameCount

12.5.1.32 StreamType

[quickSpinEnumerationNode](#) StreamType

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

- [include/spinc/TransportLayerStreamC.h](#)

12.6 quickSpinTLSystem Struct Reference

Data Fields

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

12.6.1 Field Documentation

12.6.1.1 EnumerateGen2Cameras

[quickSpinBooleanNode](#) EnumerateGen2Cameras

12.6.1.2 EnumerateGEVInterfaces

[quickSpinBooleanNode](#) EnumerateGEVInterfaces

12.6.1.3 EnumerateUSBInterfaces

`quickSpinBooleanNode` EnumerateUSBInterfaces

12.6.1.4 GenTLFNCVersionMajor

`quickSpinIntegerNode` GenTLFNCVersionMajor

12.6.1.5 GenTLFNCVersionMinor

`quickSpinIntegerNode` GenTLFNCVersionMinor

12.6.1.6 GenTLFNCVersionSubMinor

`quickSpinIntegerNode` GenTLFNCVersionSubMinor

12.6.1.7 GenTLVersionMajor

`quickSpinIntegerNode` GenTLVersionMajor

12.6.1.8 GenTLVersionMinor

`quickSpinIntegerNode` GenTLVersionMinor

12.6.1.9 GevInterfaceDefaultGateway

`quickSpinIntegerNode` GevInterfaceDefaultGateway

12.6.1.10 GevInterfaceDefaultIPAddress

`quickSpinIntegerNode` GevInterfaceDefaultIPAddress

12.6.1.11 **GevInterfaceDefaultSubnetMask**

`quickSpinIntegerNode` `GevInterfaceDefaultSubnetMask`

12.6.1.12 **GevInterfaceMACAddress**

`quickSpinIntegerNode` `GevInterfaceMACAddress`

12.6.1.13 **GevVersionMajor**

`quickSpinIntegerNode` `GevVersionMajor`

12.6.1.14 **GevVersionMinor**

`quickSpinIntegerNode` `GevVersionMinor`

12.6.1.15 **InterfaceDisplayName**

`quickSpinStringNode` `InterfaceDisplayName`

12.6.1.16 **InterfaceID**

`quickSpinStringNode` `InterfaceID`

12.6.1.17 **InterfaceSelector**

`quickSpinIntegerNode` `InterfaceSelector`

12.6.1.18 **InterfaceUpdateList**

`quickSpinCommandNode` `InterfaceUpdateList`

12.6.1.19 TLDisplayName

`quickSpinStringNode` TLDisplayName

12.6.1.20 TLFileName

`quickSpinStringNode` TLFileName

12.6.1.21 TLID

`quickSpinStringNode` TLID

12.6.1.22 TLModelName

`quickSpinStringNode` TLModelName

12.6.1.23 TLPath

`quickSpinStringNode` TLPath

12.6.1.24 TLType

`quickSpinEnumerationNode` TLType

12.6.1.25 TLVendorName

`quickSpinStringNode` TLVendorName

12.6.1.26 TLVersion

`quickSpinStringNode` TLVersion

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

- include/spinc/[TransportLayerSystemC.h](#)

12.7 spinAVIOption Struct Reference

Options for saving uncompressed videos.

Data Fields

- float [frameRate](#)
Frame rate of the stream.
- unsigned int [width](#)
Width of source image.
- unsigned int [height](#)
Height of source image.
- unsigned int [reserved](#) [192]

12.7.1 Detailed Description

Options for saving uncompressed videos.

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

12.7.2 Field Documentation

12.7.2.1 frameRate

`float frameRate`

Frame rate of the stream.

12.7.2.2 height

`unsigned int height`

Height of source image.

12.7.2.3 reserved

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

12.7.2.4 width

```
unsigned int width
```

Width of source image.

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

- include/spinc/[SpinnakerDefsC.h](#)

12.8 spinBMPOption Struct Reference

Options for saving BMP images.

Data Fields

- [bool8_t indexedColor_8bit](#)
- unsigned int [reserved](#) [16]

Reserved for future use.

12.8.1 Detailed Description

Options for saving BMP images.

Used in saving PPM images with a call to [spinImageSaveBmp\(\)](#).

12.8.2 Field Documentation

12.8.2.1 indexedColor_8bit

```
bool8_t indexedColor_8bit
```

12.8.2.2 reserved

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

Reserved for future use.

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

- [include/spinc/SpinnakerDefsC.h](#)

12.9 spinChunkData Struct Reference

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

Data Fields

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

12.9.1 Detailed Description

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

12.9.2 Field Documentation

12.9.2.1 m_blackLevel

```
double m_blackLevel
```

12.9.2.2 m_compressionMode

```
int64_t m_compressionMode
```

12.9.2.3 m_compressionRatio

```
double m_compressionRatio
```

12.9.2.4 m_counterValue

```
int64_t m_counterValue
```

12.9.2.5 m_cRC

```
int64_t m_cRC
```

12.9.2.6 m_encoderValue

```
int64_t m_encoderValue
```

12.9.2.7 m_exposureEndLineStatusAll

```
int64_t m_exposureEndLineStatusAll
```

12.9.2.8 m_exposureTime

```
double m_exposureTime
```

12.9.2.9 m_frameID

```
int64_t m_frameID
```

12.9.2.10 m_gain

```
double m_gain
```

12.9.2.11 m_height

```
int64_t m_height
```

12.9.2.12 m_image

```
int64_t m_image
```

12.9.2.13 m_inferenceConfidence

```
double m_inferenceConfidence
```

12.9.2.14 m_inferenceFrameId

```
int64_t m_inferenceFrameId
```

12.9.2.15 m_inferenceResult

```
int64_t m_inferenceResult
```

12.9.2.16 m_linePitch

```
int64_t m_linePitch
```

12.9.2.17 m_lineStatusAll

```
int64_t m_lineStatusAll
```

12.9.2.18 m_offsetX

```
int64_t m_offsetX
```

12.9.2.19 m_offsetY

```
int64_t m_offsetY
```

12.9.2.20 m_partSelector

```
int64_t m_partSelector
```

12.9.2.21 m_pixelDynamicRangeMax

```
int64_t m_pixelDynamicRangeMax
```

12.9.2.22 m_pixelDynamicRangeMin

```
int64_t m_pixelDynamicRangeMin
```

12.9.2.23 m_scan3dAxisMax

```
double m_scan3dAxisMax
```

12.9.2.24 m_scan3dAxisMin

```
double m_scan3dAxisMin
```

12.9.2.25 m_scan3dCoordinateOffset

```
double m_scan3dCoordinateOffset
```

12.9.2.26 m_scan3dCoordinateReferenceValue

```
double m_scan3dCoordinateReferenceValue
```

12.9.2.27 m_scan3dCoordinateScale

```
double m_scan3dCoordinateScale
```

12.9.2.28 m_scan3dInvalidDataValue

```
double m_scan3dInvalidDataValue
```

12.9.2.29 m_scan3dTransformValue

```
double m_scan3dTransformValue
```

12.9.2.30 m_scanLineSelector

```
int64_t m_scanLineSelector
```

12.9.2.31 m_sequencerSetActive

```
int64_t m_sequencerSetActive
```

12.9.2.32 m_serialDataLength

```
int64_t m_serialDataLength
```

12.9.2.33 m_streamChannelID

```
int64_t m_streamChannelID
```

12.9.2.34 m_timerValue

```
double m_timerValue
```

12.9.2.35 m_timestamp

```
int64_t m_timestamp
```

12.9.2.36 m_timestampLatchValue

```
int64_t m_timestampLatchValue
```

12.9.2.37 m_transferBlockID

```
int64_t m_transferBlockID
```

12.9.2.38 m_transferQueueCurrentBlockCount

```
int64_t m_transferQueueCurrentBlockCount
```

12.9.2.39 m_width

```
int64_t m_width
```

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

- include/spinc/[ChunkDataDefC.h](#)

12.10 spinH264Option Struct Reference

Options for saving H264 videos.

Data Fields

- float [frameRate](#)
Frame rate of the stream.
- unsigned int [width](#)
Width of source image.
- unsigned int [height](#)
Height of source image.
- unsigned int [bitrate](#)
Bitrate to encode at.
- unsigned int [reserved](#) [256]
Reserved for future use.

12.10.1 Detailed Description

Options for saving H264 videos.

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

12.10.2 Field Documentation

12.10.2.1 bitrate

```
unsigned int bitrate
```

Bitrate to encode at.

12.10.2.2 frameRate

```
float frameRate
```

Frame rate of the stream.

12.10.2.3 height

```
unsigned int height
```

Height of source image.

12.10.2.4 reserved

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

Reserved for future use.

12.10.2.5 width

```
unsigned int width
```

Width of source image.

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

- [include/spinc/SpinnakerDefsC.h](#)

12.11 spinJPEGOption Struct Reference

Options for saving JPEG images.

Data Fields

- [bool8_t progressive](#)
Whether to save as a progressive JPEG file.
- unsigned int [quality](#)
JPEG image quality in range (0-100).
- unsigned int [reserved](#) [16]
Reserved for future use.

12.11.1 Detailed Description

Options for saving JPEG images.

Used in saving PPM images with a call to [spinImageSaveJpeg\(\)](#).

12.11.2 Field Documentation

12.11.2.1 progressive

`bool8_t` progressive

Whether to save as a progressive JPEG file.

12.11.2.2 quality

`unsigned int` quality

JPEG image quality in range (0-100).

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

12.11.2.3 reserved

`unsigned int` reserved[16]

Reserved for future use.

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

- `include/spinc/SpinnakerDefsC.h`

12.12 spinJPG2Option Struct Reference

Options for saving JPEG 2000 images.

Data Fields

- unsigned int [quality](#)
JPEG saving quality in range (1-512).
- unsigned int [reserved](#) [16]
Reserved for future use.

12.12.1 Detailed Description

Options for saving JPEG 2000 images.

Used in saving PPM images with a call to [spinImageSaveJpg2\(\)](#).

12.12.2 Field Documentation

12.12.2.1 [quality](#)

```
unsigned int quality
```

JPEG saving quality in range (1-512).

12.12.2.2 [reserved](#)

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

Reserved for future use.

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

- include/spinc/[SpinnakerDefsC.h](#)

12.13 spinLibraryVersion Struct Reference

Provides easier access to the current version of Spinnaker.

Data Fields

- unsigned int [major](#)
Major version of the library.
- unsigned int [minor](#)
Minor version of the library.
- unsigned int [type](#)
Version type of the library.
- unsigned int [build](#)
Build number of the library.

12.13.1 Detailed Description

Provides easier access to the current version of Spinnaker.

12.13.2 Field Documentation

12.13.2.1 build

```
unsigned int build
```

Build number of the library.

12.13.2.2 major

```
unsigned int major
```

Major version of the library.

12.13.2.3 minor

```
unsigned int minor
```

Minor version of the library.

12.13.2.4 type

```
unsigned int type
```

Version type of the library.

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

- [include/spinc/SpinnakerDefsC.h](#)

12.14 spinMJPGOption Struct Reference

Options for saving MJPG videos.

Data Fields

- float [frameRate](#)
Frame rate of the stream.
- unsigned int [quality](#)
Image quality (1-100)
- unsigned int [width](#)
Width of source image.
- unsigned int [height](#)
Height of source image.
- unsigned int [reserved](#) [192]

12.14.1 Detailed Description

Options for saving MJPG videos.

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

12.14.2 Field Documentation

12.14.2.1 frameRate

```
float frameRate
```

Frame rate of the stream.

12.14.2.2 height

```
unsigned int height
```

Height of source image.

12.14.2.3 quality

```
unsigned int quality
```

Image quality (1-100)

12.14.2.4 reserved

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

12.14.2.5 width

```
unsigned int width
```

Width of source image.

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

- include/spinc/[SpinnakerDefsC.h](#)

12.15 spinPGMOption Struct Reference

Options for saving PGM images.

Data Fields

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

12.15.1 Detailed Description

Options for saving PGM images.

12.15.2 Field Documentation

12.15.2.1 binaryFile

```
bool8_t binaryFile
```

Whether to save the PPM as a binary file.

12.15.2.2 reserved

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

Reserved for future use.

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

- include/spinc/[SpinnakerDefsC.h](#)

12.16 spinPNGOption Struct Reference

Options for saving PNG images.

Data Fields

- [bool8_t interlaced](#)
Whether to save the PNG as interlaced.
- unsigned int [compressionLevel](#)
Compression level (0-9).
- unsigned int [reserved](#) [16]
Reserved for future use.

12.16.1 Detailed Description

Options for saving PNG images.

Used in saving PNG images with a call to [spinImageSavePng\(\)](#).

12.16.2 Field Documentation

12.16.2.1 compressionLevel

```
unsigned int compressionLevel
```

Compression level (0-9).

0 is no compression, 9 is best compression.

12.16.2.2 interlaced

```
bool8_t interlaced
```

Whether to save the PNG as interlaced.

12.16.2.3 reserved

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

Reserved for future use.

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

- include/spinc/[SpinnakerDefsC.h](#)

12.17 spinPPMOption Struct Reference

Options for saving PPM images.

Data Fields

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

12.17.1 Detailed Description

Options for saving PPM images.

Used in saving PPM images with a call to [spinImageSavePpm\(\)](#).

12.17.2 Field Documentation

12.17.2.1 binaryFile

```
bool8_t binaryFile
```

Whether to save the PPM as a binary file.

12.17.2.2 reserved

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

Reserved for future use.

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

- include/spinc/[SpinnakerDefsC.h](#)

12.18 spinTIFFOption Struct Reference

Options for saving TIFF images.

Data Fields

- [spinTIFFCompressionMethod](#) *compression*
Compression method to use for encoding TIFF images.
- unsigned int [reserved](#) [16]
Reserved for future use.

12.18.1 Detailed Description

Options for saving TIFF images.

Used in saving PPM images with a call to [spinImageSaveTiff\(\)](#).

12.18.2 Field Documentation

12.18.2.1 compression

[spinTIFFCompressionMethod](#) *compression*

Compression method to use for encoding TIFF images.

12.18.2.2 reserved

unsigned int `reserved[16]`

Reserved for future use.

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

- `include/spinc/SpinnakerDefsC.h`

Chapter 13

File Documentation

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

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

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

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

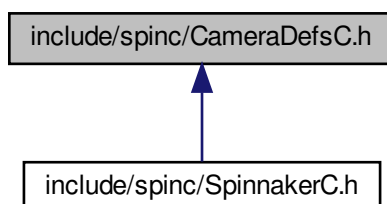
13.5 `doc/spindocs/shared/GenICamGenTL.dox` File Reference

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

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

13.8 `include/spinc/CameraDefsC.h` File Reference

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



Enumerations

- enum [spinLUTSelectorEnums](#) {
[LUTSelector_LUT1](#) ,
[NUM_LUTSELECTOR](#) }

The enum definitions for camera nodes.

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

```

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

```

- enum `spinColorTransformationSelectorEnums` {
`ColorTransformationSelector_RGBtoRGB` ,
`ColorTransformationSelector_RGBtoYUV` ,
`NUM_COLORTRANSFORMATIONSELECTOR` }
- enum `spinRgbTransformLightSourceEnums` {
`RgbTransformLightSource_General` ,
`RgbTransformLightSource_Tungsten2800K` ,
`RgbTransformLightSource_WarmFluorescent3000K` ,
`RgbTransformLightSource_CoolFluorescent4000K` ,
`RgbTransformLightSource_Daylight5000K` ,
`RgbTransformLightSource_Cloudy6500K` ,
`RgbTransformLightSource_Shade8000K` ,
`RgbTransformLightSource_Custom` ,
`NUM_RGBTRANSFORMLIGHTSOURCE` }
- enum `spinColorTransformationValueSelectorEnums` {
`ColorTransformationValueSelector_Gain00` ,
`ColorTransformationValueSelector_Gain01` ,
`ColorTransformationValueSelector_Gain02` ,
`ColorTransformationValueSelector_Gain10` ,
`ColorTransformationValueSelector_Gain11` ,
`ColorTransformationValueSelector_Gain12` ,
`ColorTransformationValueSelector_Gain20` ,
`ColorTransformationValueSelector_Gain21` ,
`ColorTransformationValueSelector_Gain22` ,
`ColorTransformationValueSelector_Offset0` ,
`ColorTransformationValueSelector_Offset1` ,
`ColorTransformationValueSelector_Offset2` ,
`NUM_COLORTRANSFORMATIONVALUESELECTOR` }
- enum `spinDeviceRegistersEndiannessEnums` {
`DeviceRegistersEndianness_Little` ,
`DeviceRegistersEndianness_Big` ,
`NUM_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_DEVICECTLTYPE` }
- 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 ,
PixelFormat_GR12 ,
PixelFormat_GB12 ,
UNKNOWN_PIXELFORMAT ,
NUM_PIXELFORMAT }
• enum spinDecimationVerticalModeEnums {
    DecimationVerticalMode_Discard ,

```

```

    NUM_DECIMATIONVERTICALMODE }

• enum spinLineModeEnums {
    LineMode_Input ,
    LineMode_Output ,
    NUM_LINEMODE }

• enum spinLineSourceEnums {
    LineSource_Off ,
    LineSource_Line0 ,
    LineSource_Line1 ,
    LineSource_Line2 ,
    LineSource_Line3 ,
    LineSource_UserOutput0 ,
    LineSource_UserOutput1 ,
    LineSource_UserOutput2 ,
    LineSource_UserOutput3 ,
    LineSource_Counter0Active ,
    LineSource_Counter1Active ,
    LineSource_LogicBlock0 ,
    LineSource_LogicBlock1 ,
    LineSource_ExposureActive ,
    LineSource_FrameTriggerWait ,
    LineSource_SerialPort0 ,
    LineSource_PPSSignal ,
    LineSource_AllPixel ,
    LineSource_AnyPixel ,
    NUM_LINESOURCE }

• enum spinLineInputFilterSelectorEnums {
    LineInputFilterSelector_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_DEVICSERIALPORTSELECTOR }
• enum spinDeviceSerialPortBaudRateEnums {
DeviceSerialPortBaudRate_Baud9600 ,
DeviceSerialPortBaudRate_Baud19200 ,
DeviceSerialPortBaudRate_Baud38400 ,
DeviceSerialPortBaudRate_Baud57600 ,
DeviceSerialPortBaudRate_Baud115200 ,
DeviceSerialPortBaudRate_Baud230400 ,
DeviceSerialPortBaudRate_Baud460800 ,
DeviceSerialPortBaudRate_Baud921600 ,
NUM_DEVICSERIALPORTBAUDRATE }
• 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_LLCCBayerRG8 ,
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 {

```

```

 ClConfiguration_Base ,
 ClConfiguration_Medium ,
 ClConfiguration_Full ,
 ClConfiguration_DualBase ,
 ClConfiguration_EightyBit ,
 NUM_CLCONFIGURATION }

• enum spinClTimeSlotsCountEnums {
 ClTimeSlotsCount_One ,
 ClTimeSlotsCount_Two ,
 ClTimeSlotsCount_Three ,
 NUM_CLTIMESLOTSCOUNT }

• enum spinCxpLinkConfigurationStatusEnums {
 CxpLinkConfigurationStatus_None ,
 CxpLinkConfigurationStatus_Pending ,
 CxpLinkConfigurationStatus_CXP1_X1 ,
 CxpLinkConfigurationStatus_CXP2_X1 ,
 CxpLinkConfigurationStatus_CXP3_X1 ,
 CxpLinkConfigurationStatus_CXP5_X1 ,
 CxpLinkConfigurationStatus_CXP6_X1 ,
 CxpLinkConfigurationStatus_CXP1_X2 ,
 CxpLinkConfigurationStatus_CXP2_X2 ,
 CxpLinkConfigurationStatus_CXP3_X2 ,
 CxpLinkConfigurationStatus_CXP5_X2 ,
 CxpLinkConfigurationStatus_CXP6_X2 ,
 CxpLinkConfigurationStatus_CXP1_X3 ,
 CxpLinkConfigurationStatus_CXP2_X3 ,
 CxpLinkConfigurationStatus_CXP3_X3 ,
 CxpLinkConfigurationStatus_CXP5_X3 ,
 CxpLinkConfigurationStatus_CXP6_X3 ,
 CxpLinkConfigurationStatus_CXP1_X4 ,
 CxpLinkConfigurationStatus_CXP2_X4 ,
 CxpLinkConfigurationStatus_CXP3_X4 ,
 CxpLinkConfigurationStatus_CXP5_X4 ,
 CxpLinkConfigurationStatus_CXP6_X4 ,
 CxpLinkConfigurationStatus_CXP1_X5 ,
 CxpLinkConfigurationStatus_CXP2_X5 ,
 CxpLinkConfigurationStatus_CXP3_X5 ,
 CxpLinkConfigurationStatus_CXP5_X5 ,
 CxpLinkConfigurationStatus_CXP6_X5 ,
 CxpLinkConfigurationStatus_CXP1_X6 ,
 CxpLinkConfigurationStatus_CXP2_X6 ,
 CxpLinkConfigurationStatus_CXP3_X6 ,
 CxpLinkConfigurationStatus_CXP5_X6 ,
 CxpLinkConfigurationStatus_CXP6_X6 ,
 NUM_CXPLINKCONFIGURATIONSTATUS }

• enum spinCxpLinkConfigurationPreferredEnums {
 CxpLinkConfigurationPreferred_CXP1_X1 ,
 CxpLinkConfigurationPreferred_CXP2_X1 ,
 CxpLinkConfigurationPreferred_CXP3_X1 ,
 CxpLinkConfigurationPreferred_CXP5_X1 ,
 CxpLinkConfigurationPreferred_CXP6_X1 ,
 CxpLinkConfigurationPreferred_CXP1_X2 ,
 CxpLinkConfigurationPreferred_CXP2_X2 ,
 CxpLinkConfigurationPreferred_CXP3_X2 ,
 CxpLinkConfigurationPreferred_CXP5_X2 ,
 CxpLinkConfigurationPreferred_CXP6_X2 ,
 CxpLinkConfigurationPreferred_CXP1_X3 ,
 CxpLinkConfigurationPreferred_CXP2_X3 ,

```



```

CxpLinkConfigurationPreferred_CXP3_X3 ,
CxpLinkConfigurationPreferred_CXP5_X3 ,
CxpLinkConfigurationPreferred_CXP6_X3 ,
CxpLinkConfigurationPreferred_CXP1_X4 ,
CxpLinkConfigurationPreferred_CXP2_X4 ,
CxpLinkConfigurationPreferred_CXP3_X4 ,
CxpLinkConfigurationPreferred_CXP5_X4 ,
CxpLinkConfigurationPreferred_CXP6_X4 ,
CxpLinkConfigurationPreferred_CXP1_X5 ,
CxpLinkConfigurationPreferred_CXP2_X5 ,
CxpLinkConfigurationPreferred_CXP3_X5 ,
CxpLinkConfigurationPreferred_CXP5_X5 ,
CxpLinkConfigurationPreferred_CXP6_X5 ,
CxpLinkConfigurationPreferred_CXP1_X6 ,
CxpLinkConfigurationPreferred_CXP2_X6 ,
CxpLinkConfigurationPreferred_CXP3_X6 ,
CxpLinkConfigurationPreferred_CXP5_X6 ,
CxpLinkConfigurationPreferred_CXP6_X6 ,
NUM_CXPLINKCONFIGURATIONPREFERRED }
• enum spinCxpLinkConfigurationEnums {
 CxpLinkConfiguration_Auto ,
 CxpLinkConfiguration_CXP1_X1 ,
 CxpLinkConfiguration_CXP2_X1 ,
 CxpLinkConfiguration_CXP3_X1 ,
 CxpLinkConfiguration_CXP5_X1 ,
 CxpLinkConfiguration_CXP6_X1 ,
 CxpLinkConfiguration_CXP1_X2 ,
 CxpLinkConfiguration_CXP2_X2 ,
 CxpLinkConfiguration_CXP3_X2 ,
 CxpLinkConfiguration_CXP5_X2 ,
 CxpLinkConfiguration_CXP6_X2 ,
 CxpLinkConfiguration_CXP1_X3 ,
 CxpLinkConfiguration_CXP2_X3 ,
 CxpLinkConfiguration_CXP3_X3 ,
 CxpLinkConfiguration_CXP5_X3 ,
 CxpLinkConfiguration_CXP6_X3 ,
 CxpLinkConfiguration_CXP1_X4 ,
 CxpLinkConfiguration_CXP2_X4 ,
 CxpLinkConfiguration_CXP3_X4 ,
 CxpLinkConfiguration_CXP5_X4 ,
 CxpLinkConfiguration_CXP6_X4 ,
 CxpLinkConfiguration_CXP1_X5 ,
 CxpLinkConfiguration_CXP2_X5 ,
 CxpLinkConfiguration_CXP3_X5 ,
 CxpLinkConfiguration_CXP5_X5 ,
 CxpLinkConfiguration_CXP6_X5 ,
 CxpLinkConfiguration_CXP1_X6 ,
 CxpLinkConfiguration_CXP2_X6 ,
 CxpLinkConfiguration_CXP3_X6 ,
 CxpLinkConfiguration_CXP5_X6 ,
 CxpLinkConfiguration_CXP6_X6 ,
 NUM_CXPLINKCONFIGURATION }
• enum spinCxpConnectionTestModeEnums {
 CxpConnectionTestMode_Off ,
 CxpConnectionTestMode_Mode1 ,
 NUM_CXPCONNECTIONTESTMODE }
• enum spinCxpPoCxpStatusEnums {
 CxpPoCxpStatus_Auto ,

```

```
CxpPoCxpStatus_Off ,
CxpPoCxpStatus_Tripped ,
NUM_CXPPOCXPSTATUS }
```

## 13.8.1 Enumeration Type Documentation

### 13.8.1.1 spinAcquisitionModeEnums

```
enum spinAcquisitionModeEnums
```

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

#### Enumerator

|                             |  |
|-----------------------------|--|
| AcquisitionMode_Continuous  |  |
| AcquisitionMode_SingleFrame |  |
| AcquisitionMode_MultiFrame  |  |
| NUM_ACQUISITIONMODE         |  |

### 13.8.1.2 spinAcquisitionStatusSelectorEnums

```
enum spinAcquisitionStatusSelectorEnums
```

< Selects the internal acquisition signal to read using AcquisitionStatus.

#### Enumerator

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

### 13.8.1.3 spinActionUnconditionalModeEnums

enum `spinActionUnconditionalModeEnums`

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

Enumerator

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

### 13.8.1.4 spinAdcBitDepthEnums

enum `spinAdcBitDepthEnums`

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

Enumerator

|                   |  |
|-------------------|--|
| AdcBitDepth_Bit8  |  |
| AdcBitDepth_Bit10 |  |
| AdcBitDepth_Bit12 |  |
| AdcBitDepth_Bit14 |  |
| NUM_ADCBITDEPTH   |  |

### 13.8.1.5 spinAutoAlgorithmSelectorEnums

enum `spinAutoAlgorithmSelectorEnums`

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

Enumerator

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

### 13.8.1.6 spinAutoExposureControlPriorityEnums

enum `spinAutoExposureControlPriorityEnums`

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

#### Enumerator

|                                                       |  |
|-------------------------------------------------------|--|
| <code>AutoExposureControlPriority_Gain</code>         |  |
| <code>AutoExposureControlPriority_ExposureTime</code> |  |
| <code>NUM_AUTOEXPOSURECONTROLPRIORITY</code>          |  |

### 13.8.1.7 spinAutoExposureLightingModeEnums

enum `spinAutoExposureLightingModeEnums`

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

#### Enumerator

|                                                  |  |
|--------------------------------------------------|--|
| <code>AutoExposureLightingMode_AutoDetect</code> |  |
| <code>AutoExposureLightingMode_Backlight</code>  |  |
| <code>AutoExposureLightingMode_Frontlight</code> |  |
| <code>AutoExposureLightingMode_Normal</code>     |  |
| <code>NUM_AUTOEXPOSURELIGHTINGMODE</code>        |  |

### 13.8.1.8 spinAutoExposureMeteringModeEnums

enum `spinAutoExposureMeteringModeEnums`

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

## Enumerator

|                                         |  |
|-----------------------------------------|--|
| AutoExposureMeteringMode_Average        |  |
| AutoExposureMeteringMode_Spot           |  |
| AutoExposureMeteringMode_Partial        |  |
| AutoExposureMeteringMode_CenterWeighted |  |
| AutoExposureMeteringMode_HistogramPeak  |  |
| NUM_AUTOEXPOSUREMETERINGMODE            |  |

## 13.8.1.9 spinAutoExposureTargetGreyValueAutoEnums

enum [spinAutoExposureTargetGreyValueAutoEnums](#)

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

## Enumerator

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

## 13.8.1.10 spinBalanceRatioSelectorEnums

enum [spinBalanceRatioSelectorEnums](#)

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

## Enumerator

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

## 13.8.1.11 spinBalanceWhiteAutoEnums

enum [spinBalanceWhiteAutoEnums](#)

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

#### Enumerator

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

### 13.8.1.12 spinBalanceWhiteAutoProfileEnums

```
enum spinBalanceWhiteAutoProfileEnums
```

< Selects the profile used by BalanceWhiteAuto.

#### Enumerator

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

### 13.8.1.13 spinBinningHorizontalModeEnums

```
enum spinBinningHorizontalModeEnums
```

<

#### Enumerator

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

**13.8.1.14 spinBinningSelectorEnums**enum `spinBinningSelectorEnums`

&lt; Selects which binning engine is controlled by the BinningHorizontal and BinningVertical features.

**Enumerator**

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

**13.8.1.15 spinBinningVerticalModeEnums**enum `spinBinningVerticalModeEnums`

&lt;

**Enumerator**

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

**13.8.1.16 spinBlackLevelAutoBalanceEnums**enum `spinBlackLevelAutoBalanceEnums`

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

**Enumerator**

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

### 13.8.1.17 spinBlackLevelAutoEnums

enum `spinBlackLevelAutoEnums`

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

#### Enumerator

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

### 13.8.1.18 spinBlackLevelSelectorEnums

enum `spinBlackLevelSelectorEnums`

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

#### Enumerator

|                            |  |
|----------------------------|--|
| BlackLevelSelector_All     |  |
| BlackLevelSelector_Analog  |  |
| BlackLevelSelector_Digital |  |
| NUM_BLACKLEVELSELECTOR     |  |

### 13.8.1.19 spinChunkBlackLevelSelectorEnums

enum `spinChunkBlackLevelSelectorEnums`

< Selects which black level to retrieve

#### Enumerator

|                             |  |
|-----------------------------|--|
| ChunkBlackLevelSelector_All |  |
| NUM_CHUNKBLACKLEVELSELECTOR |  |

### 13.8.1.20 spinChunkCounterSelectorEnums

enum `spinChunkCounterSelectorEnums`



< Selects which counter to retrieve data from.

#### Enumerator

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

### 13.8.1.21 spinChunkEncoderSelectorEnums

```
enum spinChunkEncoderSelectorEnums
```

< Selects which Encoder to retrieve data from.

#### Enumerator

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

### 13.8.1.22 spinChunkEncoderStatusEnums

```
enum spinChunkEncoderStatusEnums
```

< Returns the motion status of the selected encoder.

#### Enumerator

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

### 13.8.1.23 spinChunkExposureTimeSelectorEnums

```
enum spinChunkExposureTimeSelectorEnums
```

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

**Enumerator**

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

**13.8.1.24 spinChunkGainSelectorEnums**

enum `spinChunkGainSelectorEnums`

< Selects which gain to retrieve

**Enumerator**

|                         |  |
|-------------------------|--|
| ChunkGainSelector_All   |  |
| ChunkGainSelector_Red   |  |
| ChunkGainSelector_Green |  |
| ChunkGainSelector_Blue  |  |
| NUM_CHUNKGAINSELECTOR   |  |

**13.8.1.25 spinChunkImageComponentEnums**

enum `spinChunkImageComponentEnums`

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

**Enumerator**

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

## Enumerator

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

**13.8.1.26 spinChunkPixelFormatEnums**

enum [spinChunkPixelFormatEnums](#)

< Format of the pixel provided by the camera

## Enumerator

|                                        |  |
|----------------------------------------|--|
| ChunkPixelFormat_Mono8                 |  |
| ChunkPixelFormat_Mono12Packed          |  |
| ChunkPixelFormat_Mono16                |  |
| ChunkPixelFormat_RGB8Packed            |  |
| ChunkPixelFormat_YUV422Packed          |  |
| ChunkPixelFormat_BayerGR8              |  |
| ChunkPixelFormat_BayerRG8              |  |
| ChunkPixelFormat_BayerGB8              |  |
| ChunkPixelFormat_BayerBG8              |  |
| ChunkPixelFormat_YCbCr601_422_8_CbYCrY |  |
| NUM_CHUNKPIXELFORMAT                   |  |

**13.8.1.27 spinChunkRegionIDEnums**

enum [spinChunkRegionIDEnums](#)

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

## Enumerator

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

**13.8.1.28 spinChunkScan3dCoordinateReferenceSelectorEnums**

enum [spinChunkScan3dCoordinateReferenceSelectorEnums](#)

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

#### Enumerator

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

### 13.8.1.29 spinChunkScan3dCoordinateSelectorEnums

enum [spinChunkScan3dCoordinateSelectorEnums](#)

< Selects which Coordinate to retrieve data from.

#### Enumerator

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

### 13.8.1.30 spinChunkScan3dCoordinateSystemEnums

enum [spinChunkScan3dCoordinateSystemEnums](#)

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

#### Enumerator

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

### 13.8.1.31 spinChunkScan3dCoordinateSystemReferenceEnums

enum [spinChunkScan3dCoordinateSystemReferenceEnums](#)

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

#### Enumerator

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

### 13.8.1.32 spinChunkScan3dCoordinateTransformSelectorEnums

enum `spinChunkScan3dCoordinateTransformSelectorEnums`

< Selector for transform values.

#### Enumerator

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

### 13.8.1.33 spinChunkScan3dDistanceUnitEnums

enum `spinChunkScan3dDistanceUnitEnums`

< Returns the Distance Unit of the payload image.

#### Enumerator

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

### 13.8.1.34 spinChunkScan3dOutputModeEnums

enum `spinChunkScan3dOutputModeEnums`

< Returns the Calibrated Mode of the payload image.

#### Enumerator

|                                                               |                                                                                                                                                                                                                                                                                                             |
|---------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>ChunkScan3dOutputMode_UncalibratedC</code>              | 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.                                                                                                                                                                 |
| <code>ChunkScan3dOutputMode_CalibratedABC_Grid</code>         | 3 Coordinates in grid organization. The full 3 coordinate data with the grid array organization from the sensor kept.                                                                                                                                                                                       |
| <code>ChunkScan3dOutputMode_CalibratedABC_Point↔ Cloud</code> | 3 Coordinates without organization. The full 3 coordinate data without any organization of data points. Typically only valid points transmitted giving varying image size.                                                                                                                                  |
| <code>ChunkScan3dOutputMode_CalibratedAC</code>               | 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.                                                                                                                                      |
| <code>ChunkScan3dOutputMode_CalibratedAC_Linescan</code>      | 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.                                                                                                                                                       |
| <code>ChunkScan3dOutputMode_CalibratedC</code>                | 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.                                                                                                                                                  |
| <code>ChunkScan3dOutputMode_CalibratedC_Linescan</code>       | 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.                                                                                                                             |
| <code>ChunkScan3dOutputMode_RectifiedC</code>                 | 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. |
| <code>ChunkScan3dOutputMode_RectifiedC_Linescan</code>        | Rectified 2.5D Depth map with varying B sampling. The data is sent as rectified 1D profiles using <code>Coord3D_C</code> pixels. The B (Y) axis comes from the encoder chunk value.                                                                                                                         |
| <code>ChunkScan3dOutputMode_DisparityC</code>                 | Disparity 2.5D Depth map. The distance is inversely proportional to the pixel (disparity) value.                                                                                                                                                                                                            |
| <code>ChunkScan3dOutputMode_DisparityC_Linescan</code>        | 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.                                                                                                                                 |
| <code>NUM_CHUNKSCAN3DOUTPUTMODE</code>                        |                                                                                                                                                                                                                                                                                                             |

**13.8.1.35 spinChunkSelectorEnums**

enum `spinChunkSelectorEnums`

< Selects which chunk data to enable or disable.

Enumerator

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

**13.8.1.36 spinChunkSourceIDEnums**

enum `spinChunkSourceIDEnums`

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

Enumerator

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

**13.8.1.37 spinChunkTimerSelectorEnums**

enum `spinChunkTimerSelectorEnums`

< Selects which Timer to retrieve data from.

**Enumerator**

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

**13.8.1.38 spinChunkTransferStreamIDEnums**

enum `spinChunkTransferStreamIDEnums`

< Returns identifier of the stream that generated this block.

**Enumerator**

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

**13.8.1.39 spinClConfigurationEnums**

enum `spinClConfigurationEnums`

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

**Enumerator**

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



**13.8.1.40 spinCITimeSlotsCountEnums**enum `spinCITimeSlotsCountEnums`

< 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   |       |

**13.8.1.41 spinColorTransformationSelectorEnums**enum `spinColorTransformationSelectorEnums`

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

**Enumerator**

|                                      |  |
|--------------------------------------|--|
| ColorTransformationSelector_RGBtoRGB |  |
| ColorTransformationSelector_RGBtoYUV |  |
| NUM_COLORTRANSFORMATIONSELECTOR      |  |

**13.8.1.42 spinColorTransformationValueSelectorEnums**enum `spinColorTransformationValueSelectorEnums`

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

**Enumerator**

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

### 13.8.1.43 spinCompressionSaturationPriorityEnums

enum `spinCompressionSaturationPriorityEnums`

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

Enumerator

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

### 13.8.1.44 spinCounterEventActivationEnums

enum `spinCounterEventActivationEnums`

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

Enumerator

|                                    |  |
|------------------------------------|--|
| CounterEventActivation_LevelLow    |  |
| CounterEventActivation_LevelHigh   |  |
| CounterEventActivation_FallingEdge |  |
| CounterEventActivation_RisingEdge  |  |
| CounterEventActivation_AnyEdge     |  |
| NUM_COUNTEREVENTACTIVATION         |  |

### 13.8.1.45 spinCounterEventSourceEnums

enum `spinCounterEventSourceEnums`

< Selects the event that will increment the counter

Enumerator

|                            |         |
|----------------------------|---------|
| CounterEventSource_Off     | Off     |
| CounterEventSource_MHzTick | MHzTick |
| CounterEventSource_Line0   | Line0   |

## Enumerator

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

## 13.8.1.46 spinCounterResetActivationEnums

```
enum spinCounterResetActivationEnums
```

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

## Enumerator

|                                    |  |
|------------------------------------|--|
| CounterResetActivation_LevelLow    |  |
| CounterResetActivation_LevelHigh   |  |
| CounterResetActivation_FallingEdge |  |
| CounterResetActivation_RisingEdge  |  |
| CounterResetActivation_AnyEdge     |  |
| NUM_COUNTERRESETACTIVATION         |  |

## 13.8.1.47 spinCounterResetSourceEnums

```
enum spinCounterResetSourceEnums
```

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

## Enumerator

|                        |     |
|------------------------|-----|
| CounterResetSource_Off | Off |
|------------------------|-----|

## Enumerator

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

## 13.8.1.48 spinCounterSelectorEnums

```
enum spinCounterSelectorEnums
```

< Selects which counter to configure

## Enumerator

|                          |  |
|--------------------------|--|
| CounterSelector_Counter0 |  |
| CounterSelector_Counter1 |  |
| NUM_COUNTERSELECTOR      |  |

## 13.8.1.49 spinCounterStatusEnums

```
enum spinCounterStatusEnums
```

< Returns the current status of the counter.

## Enumerator

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

**13.8.1.50 spinCounterTriggerActivationEnums**

```
enum spinCounterTriggerActivationEnums
```

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

**Enumerator**

|                                      |  |
|--------------------------------------|--|
| CounterTriggerActivation_LevelLow    |  |
| CounterTriggerActivation_LevelHigh   |  |
| CounterTriggerActivation_FallingEdge |  |
| CounterTriggerActivation_RisingEdge  |  |
| CounterTriggerActivation_AnyEdge     |  |
| NUM_COUNTERTRIGGERACTIVATION         |  |

**13.8.1.51 spinCounterTriggerSourceEnums**

```
enum spinCounterTriggerSourceEnums
```

< Selects the source of the trigger to start the counter

**Enumerator**

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

### 13.8.1.52 spinCxpConnectionTestModeEnums

enum `spinCxpConnectionTestModeEnums`

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

#### Enumerator

|                                           |        |
|-------------------------------------------|--------|
| <code>CxpConnectionTestMode_Off</code>    | Off    |
| <code>CxpConnectionTestMode_Mode1</code>  | Mode 1 |
| <code>NUM_CXP_CONNECTION_TEST_MODE</code> |        |

### 13.8.1.53 spinCxpLinkConfigurationEnums

enum `spinCxpLinkConfigurationEnums`

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

#### Enumerator

|                                           |                                                                        |
|-------------------------------------------|------------------------------------------------------------------------|
| <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     |                                                                        |

## 13.8.1.54 spinCxpLinkConfigurationPreferredEnums

```
enum spinCxpLinkConfigurationPreferredEnums
```

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

## Enumerator

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

## Enumerator

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

## 13.8.1.55 spinCxpLinkConfigurationStatusEnums

```
enum spinCxpLinkConfigurationStatusEnums
```

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

## Enumerator

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



## Enumerator

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

## 13.8.1.56 spinCxpPoCxpStatusEnums

```
enum spinCxpPoCxpStatusEnums
```

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

## Enumerator

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

## 13.8.1.57 spinDecimationHorizontalModeEnums

```
enum spinDecimationHorizontalModeEnums
```

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

## Enumerator

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

## 13.8.1.58 spinDecimationSelectorEnums

```
enum spinDecimationSelectorEnums
```

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

## Enumerator

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

## 13.8.1.59 spinDecimationVerticalModeEnums

```
enum spinDecimationVerticalModeEnums
```

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

## Enumerator

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

## 13.8.1.60 spinDefectCorrectionModeEnums

```
enum spinDefectCorrectionModeEnums
```

< Controls the method used for replacing defective pixels.

## Enumerator

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

## 13.8.1.61 spinDeinterlacingEnums

```
enum spinDeinterlacingEnums
```

< Controls how the device performs de-interlacing.

## Enumerator

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

## 13.8.1.62 spinDeviceCharacterSetEnums

```
enum spinDeviceCharacterSetEnums
```

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

## Enumerator

|                          |  |
|--------------------------|--|
| DeviceCharacterSet_UTF8  |  |
| DeviceCharacterSet_ASCII |  |
| NUM_DEVICECHARACTERSET   |  |

## 13.8.1.63 spinDeviceClockSelectorEnums

```
enum spinDeviceClockSelectorEnums
```

< Selects the clock frequency to access from the device.

## Enumerator

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

## 13.8.1.64 spinDeviceConnectionStatusEnums

```
enum spinDeviceConnectionStatusEnums
```

< Indicates the status of the specified Connection.

## Enumerator

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

### 13.8.1.65 spinDeviceIndicatorModeEnums

enum `spinDeviceIndicatorModeEnums`

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

#### Enumerator

|                                 |  |
|---------------------------------|--|
| DeviceIndicatorMode_Inactive    |  |
| DeviceIndicatorMode_Active      |  |
| DeviceIndicatorMode_ErrorStatus |  |
| NUM_DEVICEINDICATORMODE         |  |

### 13.8.1.66 spinDeviceLinkHeartbeatModeEnums

enum `spinDeviceLinkHeartbeatModeEnums`

< Activate or deactivate the Link's heartbeat.

#### Enumerator

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

### 13.8.1.67 spinDeviceLinkThroughputLimitModeEnums

enum `spinDeviceLinkThroughputLimitModeEnums`

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

#### Enumerator

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

### 13.8.1.68 spinDevicePowerSupplySelectorEnums

enum [spinDevicePowerSupplySelectorEnums](#)

< Selects the power supply source to control or read.

Enumerator

|                                    |  |
|------------------------------------|--|
| DevicePowerSupplySelector_External |  |
| NUM_DEVICEPOWERSUPPLYSELECTOR      |  |

### 13.8.1.69 spinDeviceRegistersEndiannessEnums

enum [spinDeviceRegistersEndiannessEnums](#)

< Endianness of the registers of the device.

Enumerator

|                                  |  |
|----------------------------------|--|
| DeviceRegistersEndianness_Little |  |
| DeviceRegistersEndianness_Big    |  |
| NUM_DEVICEREGISTERSENDIANNESSE   |  |

### 13.8.1.70 spinDeviceScanTypeEnums

enum [spinDeviceScanTypeEnums](#)

< Scan type of the sensor of the device.

Enumerator

|                         |  |
|-------------------------|--|
| DeviceScanType_Areascan |  |
| NUM_DEVICEASCANTYPE     |  |

### 13.8.1.71 spinDeviceSerialPortBaudRateEnums

enum [spinDeviceSerialPortBaudRateEnums](#)

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

## Enumerator

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

**13.8.1.72 spinDeviceSerialPortSelectorEnums**

```
enum spinDeviceSerialPortSelectorEnums
```

< Selects which serial port of the device to control.

## Enumerator

|                                     |                                                       |
|-------------------------------------|-------------------------------------------------------|
| DeviceSerialPortSelector_CameraLink | Serial port associated to the Camera link connection. |
| NUM_DEVICESERIALPORTSELECTOR        |                                                       |

**13.8.1.73 spinDeviceStreamChannelEndiannessEnums**

```
enum spinDeviceStreamChannelEndiannessEnums
```

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

## Enumerator

|                                      |                                       |
|--------------------------------------|---------------------------------------|
| DeviceStreamChannelEndianness_Big    | Stream channel data is big Endian.    |
| DeviceStreamChannelEndianness_Little | Stream channel data is little Endian. |
| NUM_DEVICESTREAMCHANNELENDIANNES     |                                       |

**13.8.1.74 spinDeviceStreamChannelTypeEnums**

```
enum spinDeviceStreamChannelTypeEnums
```

< Reports the type of the stream channel.

## Enumerator

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

## 13.8.1.75 spinDeviceTapGeometryEnums

enum [spinDeviceTapGeometryEnums](#)

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

## Enumerator

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

## Enumerator

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

## 13.8.1.76 spinDeviceTemperatureSelectorEnums

enum [spinDeviceTemperatureSelectorEnums](#)

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

## Enumerator

|                                  |  |
|----------------------------------|--|
| DeviceTemperatureSelector_Sensor |  |
| NUM_DEVICETEMPERATURESELECTOR    |  |

## 13.8.1.77 spinDeviceTLTypeEnums

enum [spinDeviceTLTypeEnums](#)

< Transport Layer type of the device.

## Enumerator

|                           |  |
|---------------------------|--|
| DeviceTLType_GigEVision   |  |
| DeviceTLType_CameraLink   |  |
| DeviceTLType_CameraLinkHS |  |
| DeviceTLType_CoaXPress    |  |
| DeviceTLType_USB3Vision   |  |
| DeviceTLType_Custom       |  |
| NUM_DEVICETLTYPE          |  |



**13.8.1.78 spinDeviceTypeEnums**

enum [spinDeviceTypeEnums](#)

< Returns the device type.

**Enumerator**

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

**13.8.1.79 spinEncoderModeEnums**

enum [spinEncoderModeEnums](#)

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

**Enumerator**

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

**13.8.1.80 spinEncoderOutputModeEnums**

enum [spinEncoderOutputModeEnums](#)

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

**Enumerator**

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

## Enumerator

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

## 13.8.1.81 spinEncoderResetActivationEnums

```
enum spinEncoderResetActivationEnums
```

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

## Enumerator

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

## 13.8.1.82 spinEncoderResetSourceEnums

```
enum spinEncoderResetSourceEnums
```

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

## Enumerator

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

## Enumerator

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

## 13.8.1.83 spinEncoderSelectorEnums

```
enum spinEncoderSelectorEnums
```

< Selects which Encoder to configure.

## Enumerator

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

### 13.8.1.84 spinEncoderSourceAEnums

enum [spinEncoderSourceAEnums](#)

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

#### Enumerator

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

### 13.8.1.85 spinEncoderSourceBEnums

enum [spinEncoderSourceBEnums](#)

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

#### Enumerator

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

### 13.8.1.86 spinEncoderStatusEnums

enum [spinEncoderStatusEnums](#)

< Returns the motion status of the encoder.

#### Enumerator

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

### 13.8.1.87 spinEventNotificationEnums

enum `spinEventNotificationEnums`

< Enables/Disables the selected event.

Enumerator

|                       |  |
|-----------------------|--|
| EventNotification_On  |  |
| EventNotification_Off |  |
| NUM_EVENTNOTIFICATION |  |

### 13.8.1.88 spinEventSelectorEnums

enum `spinEventSelectorEnums`

< Selects which Event to enable or disable.

Enumerator

|                                 |  |
|---------------------------------|--|
| EventSelector_Error             |  |
| EventSelector_ExposureEnd       |  |
| EventSelector_SerialPortReceive |  |
| NUM_EVENTSELECTOR               |  |

### 13.8.1.89 spinExposureActiveModeEnums

enum `spinExposureActiveModeEnums`

< Control sensor active exposure mode.

Enumerator

|                              |  |
|------------------------------|--|
| ExposureActiveMode_Line1     |  |
| ExposureActiveMode_AnyPixels |  |
| ExposureActiveMode_AllPixels |  |
| NUM_EXPOSUREACTIVEMODE       |  |

### 13.8.1.90 spinExposureAutoEnums

enum `spinExposureAutoEnums`

< Sets the automatic exposure mode

#### Enumerator

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

### 13.8.1.91 spinExposureModeEnums

enum `spinExposureModeEnums`

< Sets the operation mode of the Exposure.

#### Enumerator

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

### 13.8.1.92 spinExposureTimeModeEnums

enum `spinExposureTimeModeEnums`

< Sets the configuration mode of the ExposureTime feature.

#### Enumerator

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

**13.8.1.93 spinExposureTimeSelectorEnums**

enum `spinExposureTimeSelectorEnums`

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

**Enumerator**

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

**13.8.1.94 spinFileOpenModeEnums**

enum `spinFileOpenModeEnums`

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

**Enumerator**

|                        |  |
|------------------------|--|
| FileOpenMode_Read      |  |
| FileOpenMode_Write     |  |
| FileOpenMode_ReadWrite |  |
| NUM_FILEOPENMODE       |  |

**13.8.1.95 spinFileOperationSelectorEnums**

enum `spinFileOperationSelectorEnums`

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

**Enumerator**

|                              |  |
|------------------------------|--|
| FileOperationSelector_Open   |  |
| FileOperationSelector_Close  |  |
| FileOperationSelector_Read   |  |
| FileOperationSelector_Write  |  |
| FileOperationSelector_Delete |  |
| NUM_FILEOPERATIONSELECTOR    |  |

**13.8.1.96 spinFileOperationStatusEnums**

enum `spinFileOperationStatusEnums`

< Represents the file operation execution status.

**Enumerator**

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

**13.8.1.97 spinFileSelectorEnums**

enum `spinFileSelectorEnums`

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

**Enumerator**

|                             |  |
|-----------------------------|--|
| FileSelector_UserSetDefault |  |
| FileSelector_UserSet0       |  |
| FileSelector_UserSet1       |  |
| FileSelector_UserFile1      |  |
| FileSelector_SerialPort0    |  |
| NUM_FILESELECTOR            |  |

**13.8.1.98 spinGainAutoBalanceEnums**

enum `spinGainAutoBalanceEnums`



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

**Enumerator**

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

**13.8.1.99 spinGainAutoEnums**

enum `spinGainAutoEnums`

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

**Enumerator**

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

**13.8.1.100 spinGainSelectorEnums**

enum `spinGainSelectorEnums`

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

**Enumerator**

|                  |  |
|------------------|--|
| GainSelector_All |  |
| NUM_GAINSELECTOR |  |

**13.8.1.101 spinGevCCPEnums**

enum `spinGevCCPEnums`

< Controls the device access privilege of an application.

## Enumerator

|                        |  |
|------------------------|--|
| GevCCP_OpenAccess      |  |
| GevCCP_ExclusiveAccess |  |
| GevCCP_ControlAccess   |  |
| NUM_GEVCCP             |  |

**13.8.1.102 spinGevCurrentPhysicalLinkConfigurationEnums**

enum `spinGevCurrentPhysicalLinkConfigurationEnums`

< Indicates the current physical link configuration of the device.

## Enumerator

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

**13.8.1.103 spinGevGVCPExtendedStatusCodesSelectorEnums**

enum `spinGevGVCPExtendedStatusCodesSelectorEnums`

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

## Enumerator

|                                               |             |
|-----------------------------------------------|-------------|
| GevGVCPExtendedStatusCodesSelector_Version1_1 | Version 1 1 |
| GevGVCPExtendedStatusCodesSelector_Version2_0 | Version 2 0 |
| NUM_GEVGVCPEXTENDEDSTATUSCODESSELECTOR        |             |

**13.8.1.104 spinGevGVSPExtendedIDModeEnums**

enum `spinGevGVSPExtendedIDModeEnums`

< Enables the extended IDs mode.

## Enumerator

|                           |     |
|---------------------------|-----|
| GevGVSPExtendedIDMode_Off | Off |
| GevGVSPExtendedIDMode_On  | On  |
| NUM_GEVGVSPEXTENDEDIDMODE |     |

**13.8.1.105 spinGevIEEE1588ClockAccuracyEnums**

enum [spinGevIEEE1588ClockAccuracyEnums](#)

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

## Enumerator

|                                  |                  |
|----------------------------------|------------------|
| GevIEEE1588ClockAccuracy_Unknown | Unknown Accuracy |
| NUM_GEVIEEE1588CLOCKACCURACY     |                  |

**13.8.1.106 spinGevIEEE1588ModeEnums**

enum [spinGevIEEE1588ModeEnums](#)

< Provides the mode of the IEEE 1588 clock.

## Enumerator

|                           |            |
|---------------------------|------------|
| GevIEEE1588Mode_Auto      | Automatic  |
| GevIEEE1588Mode_SlaveOnly | Slave Only |
| NUM_GEVIEEE1588MODE       |            |

**13.8.1.107 spinGevIEEE1588StatusEnums**

enum [spinGevIEEE1588StatusEnums](#)

< Provides the status of the IEEE 1588 clock.

## Enumerator

|                                |              |
|--------------------------------|--------------|
| GevIEEE1588Status_Initializing | Initializing |
| GevIEEE1588Status_Faulty       | Faulty       |
| GevIEEE1588Status_Disabled     | Disabled     |

## Enumerator

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

**13.8.1.108 spinGevIPConfigurationStatusEnums**

enum `spinGevIPConfigurationStatusEnums`

< Reports the current IP configuration status.

## Enumerator

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

**13.8.1.109 spinGevPhysicalLinkConfigurationEnums**

enum `spinGevPhysicalLinkConfigurationEnums`

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

## Enumerator

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

**13.8.1.110 spinGevSupportedOptionSelectorEnums**

enum `spinGevSupportedOptionSelectorEnums`

< Selects the GEV option to interrogate for existing support.

**Enumerator**

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

**13.8.1.111 spinImageComponentSelectorEnums**

enum `spinImageComponentSelectorEnums`

< Selects a component to activate data streaming from.

**Enumerator**

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

## Enumerator

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

## 13.8.1.112 spinImageCompressionJPEGFormatOptionEnums

enum `spinImageCompressionJPEGFormatOptionEnums`

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

## Enumerator

|                                                         |                                                                                                                                                                           |
|---------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ImageCompressionJPEGFormatOption_Lossless               | Selects lossless JPEG compression based on a predictive coding model.                                                                                                     |
| ImageCompressionJPEGFormatOption_Baseline↔<br>Standard  | Indicates this is a baseline sequential (single-scan) DCT-based JPEG.                                                                                                     |
| ImageCompressionJPEGFormatOption_Baseline↔<br>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_↔<br>IMAGECOMPRESSIONJPEGFORMATOPTION               |                                                                                                                                                                           |

## 13.8.1.113 spinImageCompressionModeEnums

enum `spinImageCompressionModeEnums`

<

## Enumerator

|                               |  |
|-------------------------------|--|
| ImageCompressionMode_Off      |  |
| ImageCompressionMode_Lossless |  |
| NUM_IMAGECOMPRESSIONMODE      |  |

**13.8.1.114 spinImageCompressionRateOptionEnums**

enum [spinImageCompressionRateOptionEnums](#)

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

## Enumerator

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

**13.8.1.115 spinLineFormatEnums**

enum [spinLineFormatEnums](#)

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

## Enumerator

|                        |  |
|------------------------|--|
| LineFormat_NoConnect   |  |
| LineFormat_TriState    |  |
| LineFormat_TTL         |  |
| LineFormat_LVDS        |  |
| LineFormat_RS422       |  |
| LineFormat_OptoCoupled |  |
| LineFormat_OpenDrain   |  |
| NUM_LINEFORMAT         |  |

**13.8.1.116 spinLineInputFilterSelectorEnums**

enum [spinLineInputFilterSelectorEnums](#)



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

#### Enumerator

|                                  |  |
|----------------------------------|--|
| LineInputFilterSelector_Deglitch |  |
| LineInputFilterSelector_Debounce |  |
| NUM_LINEINPUTFILTERSELECTOR      |  |

### 13.8.1.117 spinLineModeEnums

enum [spinLineModeEnums](#)

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

#### Enumerator

|                 |  |
|-----------------|--|
| LineMode_Input  |  |
| LineMode_Output |  |
| NUM_LINEMODE    |  |

### 13.8.1.118 spinLineSelectorEnums

enum [spinLineSelectorEnums](#)

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

#### Enumerator

|                    |  |
|--------------------|--|
| LineSelector_Line0 |  |
| LineSelector_Line1 |  |
| LineSelector_Line2 |  |
| LineSelector_Line3 |  |
| NUM_LINESELECTOR   |  |

### 13.8.1.119 spinLineSourceEnums

enum [spinLineSourceEnums](#)

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

## Enumerator

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

**13.8.1.120 spinLogicBlockLUTInputActivationEnums**

```
enum spinLogicBlockLUTInputActivationEnums
```

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

## Enumerator

|                                          |  |
|------------------------------------------|--|
| LogicBlockLUTInputActivation_LevelLow    |  |
| LogicBlockLUTInputActivation_LevelHigh   |  |
| LogicBlockLUTInputActivation_FallingEdge |  |
| LogicBlockLUTInputActivation_RisingEdge  |  |
| LogicBlockLUTInputActivation_AnyEdge     |  |
| NUM_LOGICBLOCKLUTINPUTACTIVATION         |  |

**13.8.1.121 spinLogicBlockLUTInputSelectorEnums**

```
enum spinLogicBlockLUTInputSelectorEnums
```

< Controls which LogicBlockLUT Input Source & Activation to access.

## Enumerator

|                                   |  |
|-----------------------------------|--|
| LogicBlockLUTInputSelector_Input0 |  |
| LogicBlockLUTInputSelector_Input1 |  |
| LogicBlockLUTInputSelector_Input2 |  |
| LogicBlockLUTInputSelector_Input3 |  |
| NUM_LOGICBLOCKLUTINPUTSELECTOR    |  |

## 13.8.1.122 spinLogicBlockLUTInputSourceEnums

enum [spinLogicBlockLUTInputSourceEnums](#)

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

## Enumerator

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

## 13.8.1.123 spinLogicBlockLUTSelectorEnums

enum [spinLogicBlockLUTSelectorEnums](#)

< Selects which LogicBlock LUT to configure

**Enumerator**

|                              |  |
|------------------------------|--|
| LogicBlockLUTSelector_Value  |  |
| LogicBlockLUTSelector_Enable |  |
| NUM_LOGICBLOCKLUTSELECTOR    |  |

**13.8.1.124 spinLogicBlockSelectorEnums**

```
enum spinLogicBlockSelectorEnums
```

< Selects which LogicBlock to configure

**Enumerator**

|                                |  |
|--------------------------------|--|
| LogicBlockSelector_LogicBlock0 |  |
| LogicBlockSelector_LogicBlock1 |  |
| NUM_LOGICBLOCKSELECTOR         |  |

**13.8.1.125 spinLUTSelectorEnums**

```
enum spinLUTSelectorEnums
```

The enum definitions for camera nodes.

< Selects which LUT to control.

**Enumerator**

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

**13.8.1.126 spinPixelColorFilterEnums**

```
enum spinPixelColorFilterEnums
```

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

**Enumerator**

|                          |                         |
|--------------------------|-------------------------|
| PixelColorFilter_None    | No color filter.        |
| PixelColorFilter_BayerRG | Bayer Red Green filter. |

## Enumerator

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

## 13.8.1.127 spinPixelFormatEnums

enum `spinPixelFormatEnums`

< Format of the pixel provided by the camera.

## Enumerator

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

## Enumerator

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

## Enumerator

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

## Enumerator

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



## Enumerator

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

## Enumerator

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

## 13.8.1.128 spinPixelFormatInfoSelectorEnums

```
enum spinPixelFormatInfoSelectorEnums
```

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

## Enumerator

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

## Enumerator

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

## Enumerator

|                                               |                                                               |
|-----------------------------------------------|---------------------------------------------------------------|
| PixelFormatInfoSelector_Coord3D_ABC16         | 3D coordinate A-B-C 16-bit                                    |
| PixelFormatInfoSelector_Coord3D_ABC16_Planar  | 3D coordinate A-B-C 16-bit planar                             |
| PixelFormatInfoSelector_Coord3D_ABC32f        | 3D coordinate A-B-C 32-bit floating point                     |
| PixelFormatInfoSelector_Coord3D_ABC32f_Planar | 3D coordinate A-B-C 32-bit floating point planar              |
| PixelFormatInfoSelector_Coord3D_AC8           | 3D coordinate A-C 8-bit                                       |
| PixelFormatInfoSelector_Coord3D_AC8_Planar    | 3D coordinate A-C 8-bit planar                                |
| PixelFormatInfoSelector_Coord3D_AC10p         | 3D coordinate A-C 10-bit packed                               |
| PixelFormatInfoSelector_Coord3D_AC10p_Planar  | 3D coordinate A-C 10-bit packed planar                        |
| PixelFormatInfoSelector_Coord3D_AC12p         | 3D coordinate A-C 12-bit packed                               |
| PixelFormatInfoSelector_Coord3D_AC12p_Planar  | 3D coordinate A-C 12-bit packed planar                        |
| PixelFormatInfoSelector_Coord3D_AC16          | 3D coordinate A-C 16-bit                                      |
| PixelFormatInfoSelector_Coord3D_AC16_Planar   | 3D coordinate A-C 16-bit planar                               |
| PixelFormatInfoSelector_Coord3D_AC32f         | 3D coordinate A-C 32-bit floating point                       |
| PixelFormatInfoSelector_Coord3D_AC32f_Planar  | 3D coordinate A-C 32-bit floating point planar                |
| PixelFormatInfoSelector_Coord3D_A8            | 3D coordinate A 8-bit                                         |
| PixelFormatInfoSelector_Coord3D_A10p          | 3D coordinate A 10-bit packed                                 |
| PixelFormatInfoSelector_Coord3D_A12p          | 3D coordinate A 12-bit packed                                 |
| PixelFormatInfoSelector_Coord3D_A16           | 3D coordinate A 16-bit                                        |
| PixelFormatInfoSelector_Coord3D_A32f          | 3D coordinate A 32-bit floating point                         |
| PixelFormatInfoSelector_Coord3D_B8            | 3D coordinate B 8-bit                                         |
| PixelFormatInfoSelector_Coord3D_B10p          | 3D coordinate B 10-bit packed                                 |
| PixelFormatInfoSelector_Coord3D_B12p          | 3D coordinate B 12-bit packed                                 |
| PixelFormatInfoSelector_Coord3D_B16           | 3D coordinate B 16-bit                                        |
| PixelFormatInfoSelector_Coord3D_B32f          | 3D coordinate B 32-bit floating point                         |
| PixelFormatInfoSelector_Coord3D_C8            | 3D coordinate C 8-bit                                         |
| PixelFormatInfoSelector_Coord3D_C10p          | 3D coordinate C 10-bit packed                                 |
| PixelFormatInfoSelector_Coord3D_C12p          | 3D coordinate C 12-bit packed                                 |
| PixelFormatInfoSelector_Coord3D_C16           | 3D coordinate C 16-bit                                        |
| PixelFormatInfoSelector_Coord3D_C32f          | 3D coordinate C 32-bit floating point                         |
| PixelFormatInfoSelector_Confidence1           | Confidence 1-bit unpacked                                     |
| PixelFormatInfoSelector_Confidence1p          | Confidence 1-bit packed                                       |
| PixelFormatInfoSelector_Confidence8           | Confidence 8-bit                                              |
| PixelFormatInfoSelector_Confidence16          | Confidence 16-bit                                             |
| PixelFormatInfoSelector_Confidence32f         | Confidence 32-bit floating point                              |
| PixelFormatInfoSelector_BiColorBGRG8          | Bi-color Blue/Green - Red/Green 8-bit                         |
| PixelFormatInfoSelector_BiColorBGRG10         | Bi-color Blue/Green - Red/Green 10-bit unpacked               |
| PixelFormatInfoSelector_BiColorBGRG10p        | Bi-color Blue/Green - Red/Green 10-bit packed                 |
| PixelFormatInfoSelector_BiColorBGRG12         | Bi-color Blue/Green - Red/Green 12-bit unpacked               |
| PixelFormatInfoSelector_BiColorBGRG12p        | Bi-color Blue/Green - Red/Green 12-bit packed                 |
| PixelFormatInfoSelector_BiColorRGBG8          | Bi-color Red/Green - Blue/Green 8-bit                         |
| PixelFormatInfoSelector_BiColorRGBG10         | Bi-color Red/Green - Blue/Green 10-bit unpacked               |
| PixelFormatInfoSelector_BiColorRGBG10p        | Bi-color Red/Green - Blue/Green 10-bit packed                 |
| PixelFormatInfoSelector_BiColorRGBG12         | Bi-color Red/Green - Blue/Green 12-bit unpacked               |
| PixelFormatInfoSelector_BiColorRGBG12p        | Bi-color Red/Green - Blue/Green 12-bit packed                 |
| PixelFormatInfoSelector_SCF1BWBG8             | Sparse Color Filter #1 White-Blue-White-Green 8-bit           |
| PixelFormatInfoSelector_SCF1BWBG10            | Sparse Color Filter #1 White-Blue-White-Green 10-bit unpacked |

## Enumerator

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

## Enumerator

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

## Enumerator

|                                                  |                                                   |
|--------------------------------------------------|---------------------------------------------------|
| PixelFormatInfoSelector_YCbCr709_422_10p_Cb↔YCrY | YCbCr 4:2:2 10-bit packed BT.709                  |
| PixelFormatInfoSelector_YCbCr709_422_12          | YCbCr 4:2:2 12-bit unpacked BT.709                |
| PixelFormatInfoSelector_YCbCr709_422_12_Cb↔YCrY  | YCbCr 4:2:2 12-bit unpacked BT.709                |
| PixelFormatInfoSelector_YCbCr709_422_12p         | YCbCr 4:2:2 12-bit packed BT.709                  |
| PixelFormatInfoSelector_YCbCr709_422_12p_Cb↔YCrY | YCbCr 4:2:2 12-bit packed BT.709                  |
| PixelFormatInfoSelector_YUV8_UYV                 | YUV 4:4:4 8-bit                                   |
| PixelFormatInfoSelector_YUV411_8_UYVYY           | YUV 4:1:1 8-bit                                   |
| PixelFormatInfoSelector_YUV422_8                 | YUV 4:2:2 8-bit                                   |
| PixelFormatInfoSelector_YUV422_8_UYVY            | YUV 4:2:2 8-bit                                   |
| PixelFormatInfoSelector_Polarized8               | Monochrome Polarized 8-bit                        |
| PixelFormatInfoSelector_Polarized10p             | Monochrome Polarized 10-bit packed                |
| PixelFormatInfoSelector_Polarized12p             | Monochrome Polarized 12-bit packed                |
| PixelFormatInfoSelector_Polarized16              | Monochrome Polarized 16-bit                       |
| PixelFormatInfoSelector_BayerRGPolarized8        | Polarized Bayer Red Green filter 8-bit            |
| PixelFormatInfoSelector_BayerRGPolarized10p      | Polarized Bayer Red Green filter 10-bit packed    |
| PixelFormatInfoSelector_BayerRGPolarized12p      | Polarized Bayer Red Green filter 12-bit packed    |
| PixelFormatInfoSelector_BayerRGPolarized16       | Polarized Bayer Red Green filter 16-bit           |
| PixelFormatInfoSelector_LLCMono8                 | Lossless Compression Monochrome 8-bit             |
| PixelFormatInfoSelector_LLCBayerRG8              | Lossless Compression Bayer Red Green filter 8-bit |
| PixelFormatInfoSelector_JPEGMono8                | JPEG Monochrome 8-bit                             |
| PixelFormatInfoSelector_JPEGColor8               | JPEG Color 8-bit                                  |
| NUM_PIXELFORMATINFOSELECTOR                      |                                                   |

## 13.8.1.129 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. |



**Enumerator**

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

**13.8.1.130 spinRegionDestinationEnums**

enum [spinRegionDestinationEnums](#)

< Control the destination of the selected region.

**Enumerator**

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

**13.8.1.131 spinRegionModeEnums**

enum [spinRegionModeEnums](#)

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

**Enumerator**

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

**13.8.1.132 spinRegionSelectorEnums**

enum [spinRegionSelectorEnums](#)

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

**Enumerator**

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

**13.8.1.133 spinRgbTransformLightSourceEnums**

enum `spinRgbTransformLightSourceEnums`

< Used to select from a set of RGBtoRGB transform 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**

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

**13.8.1.134 spinScan3dCoordinateReferenceSelectorEnums**

enum `spinScan3dCoordinateReferenceSelectorEnums`

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

## Enumerator

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

## 13.8.1.135 spinScan3dCoordinateSelectorEnums

enum `spinScan3dCoordinateSelectorEnums`

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

## Enumerator

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

## 13.8.1.136 spinScan3dCoordinateSystemEnums

enum `spinScan3dCoordinateSystemEnums`

< Specifies the Coordinate system to use for the device.

## Enumerator

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

## 13.8.1.137 spinScan3dCoordinateSystemReferenceEnums

enum `spinScan3dCoordinateSystemReferenceEnums`

< Defines coordinate system reference location.

## Enumerator

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

**13.8.1.138 spinScan3dCoordinateTransformSelectorEnums**

enum [spinScan3dCoordinateTransformSelectorEnums](#)

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

## Enumerator

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

**13.8.1.139 spinScan3dDistanceUnitEnums**

enum [spinScan3dDistanceUnitEnums](#)

< Specifies the unit used when delivering calibrated distance data.

## Enumerator

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

**13.8.1.140 spinScan3dOutputModeEnums**

enum [spinScan3dOutputModeEnums](#)

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

## Enumerator

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

## 13.8.1.141 spinSensorDigitizationTapsEnums

```
enum spinSensorDigitizationTapsEnums
```

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

## Enumerator

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

## Enumerator

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

## 13.8.1.142 spinSensorShutterModeEnums

enum `spinSensorShutterModeEnums`

< Sets the shutter mode of the device.

## Enumerator

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

## 13.8.1.143 spinSensorTapsEnums

enum `spinSensorTapsEnums`

< Number of taps of the camera sensor.

## Enumerator

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

**13.8.1.144 spinSequencerConfigurationModeEnums**

enum `spinSequencerConfigurationModeEnums`

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

Enumerator

|                                |  |
|--------------------------------|--|
| SequencerConfigurationMode_Off |  |
| SequencerConfigurationMode_On  |  |
| NUM_SEQUENCERCONFIGURATIONMODE |  |

**13.8.1.145 spinSequencerConfigurationValidEnums**

enum `spinSequencerConfigurationValidEnums`

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

Enumerator

|                                 |  |
|---------------------------------|--|
| SequencerConfigurationValid_No  |  |
| SequencerConfigurationValid_Yes |  |
| NUM_SEQUENCERCONFIGURATIONVALID |  |

**13.8.1.146 spinSequencerModeEnums**

enum `spinSequencerModeEnums`

< Controls whether or not a sequencer is active.

Enumerator

|                   |  |
|-------------------|--|
| SequencerMode_Off |  |
| SequencerMode_On  |  |
| NUM_SEQUENCERMODE |  |

**13.8.1.147 spinSequencerSetValidEnums**

enum `spinSequencerSetValidEnums`

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



## Enumerator

|                       |  |
|-----------------------|--|
| SequencerSetValid_No  |  |
| SequencerSetValid_Yes |  |
| NUM_SEQUENCERSETVALID |  |

**13.8.1.148 spinSequencerTriggerActivationEnums**

enum `spinSequencerTriggerActivationEnums`

< Specifies the activation mode of the sequencer trigger.

## Enumerator

|                                        |  |
|----------------------------------------|--|
| SequencerTriggerActivation_RisingEdge  |  |
| SequencerTriggerActivation_FallingEdge |  |
| SequencerTriggerActivation_AnyEdge     |  |
| SequencerTriggerActivation_LevelHigh   |  |
| SequencerTriggerActivation_LevelLow    |  |
| NUM_SEQUENCERTRIGGERACTIVATION         |  |

**13.8.1.149 spinSequencerTriggerSourceEnums**

enum `spinSequencerTriggerSourceEnums`

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

## Enumerator

|                                   |  |
|-----------------------------------|--|
| SequencerTriggerSource_Off        |  |
| SequencerTriggerSource_FrameStart |  |
| NUM_SEQUENCERTRIGGERSOURCE        |  |

**13.8.1.150 spinSerialPortBaudRateEnums**

enum `spinSerialPortBaudRateEnums`

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

## Enumerator

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

**13.8.1.151 spinSerialPortParityEnums**

enum `spinSerialPortParityEnums`

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

## Enumerator

|                        |  |
|------------------------|--|
| SerialPortParity_None  |  |
| SerialPortParity_Odd   |  |
| SerialPortParity_Even  |  |
| SerialPortParity_Mark  |  |
| SerialPortParity_Space |  |
| NUM_SERIALPORTPARITY   |  |

**13.8.1.152 spinSerialPortSelectorEnums**

enum `spinSerialPortSelectorEnums`

< Selects which serial port of the device to control.

## Enumerator

|                                |  |
|--------------------------------|--|
| SerialPortSelector_SerialPort0 |  |
| NUM_SERIALPORTSELECTOR         |  |

**13.8.1.153 spinSerialPortSourceEnums**

enum `spinSerialPortSourceEnums`

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

**Enumerator**

|                        |  |
|------------------------|--|
| SerialPortSource_Line0 |  |
| SerialPortSource_Line1 |  |
| SerialPortSource_Line2 |  |
| SerialPortSource_Line3 |  |
| SerialPortSource_Off   |  |
| NUM_SERIALPORTSOURCE   |  |

**13.8.1.154 spinSerialPortStopBitsEnums**

enum `spinSerialPortStopBitsEnums`

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

**Enumerator**

|                                  |  |
|----------------------------------|--|
| SerialPortStopBits_Bits1         |  |
| SerialPortStopBits_Bits1AndAHalf |  |
| SerialPortStopBits_Bits2         |  |
| NUM_SERIALPORTSTOPBITS           |  |

**13.8.1.155 spinSoftwareSignalSelectorEnums**

enum `spinSoftwareSignalSelectorEnums`

< Selects which Software Signal features to control.

**Enumerator**

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

**13.8.1.156 spinSourceSelectorEnums**

enum `spinSourceSelectorEnums`

< Selects the source to control.

**Enumerator**

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

**13.8.1.157 spinTestPatternEnums**

enum `spinTestPatternEnums`

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

**Enumerator**

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

**13.8.1.158 spinTestPatternGeneratorSelectorEnums**

enum `spinTestPatternGeneratorSelectorEnums`

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

**Enumerator**

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

**13.8.1.159 spinTimerSelectorEnums**

```
enum spinTimerSelectorEnums
```

< Selects which Timer to configure.

**Enumerator**

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

**13.8.1.160 spinTimerStatusEnums**

```
enum spinTimerStatusEnums
```

< Returns the current status of the Timer.

**Enumerator**

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

**13.8.1.161 spinTimerTriggerActivationEnums**

```
enum spinTimerTriggerActivationEnums
```

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

**Enumerator**

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

### 13.8.1.162 spinTimerTriggerSourceEnums

enum `spinTimerTriggerSourceEnums`

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

#### Enumerator

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

## Enumerator

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

## 13.8.1.163 spinTransferComponentSelectorEnums

enum `spinTransferComponentSelectorEnums`

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

## Enumerator

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

## 13.8.1.164 spinTransferControlModeEnums

enum `spinTransferControlModeEnums`

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

## Enumerator

|                                    |                 |
|------------------------------------|-----------------|
| TransferControlMode_Basic          | Basic           |
| TransferControlMode_Automatic      | Automatic       |
| TransferControlMode_UserControlled | User Controlled |
| NUM_TRANSFERCONTROLMODE            |                 |

**13.8.1.165 spinTransferOperationModeEnums**

enum `spinTransferOperationModeEnums`

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

## Enumerator

|                                  |             |
|----------------------------------|-------------|
| TransferOperationMode_Continuous | Continuous  |
| TransferOperationMode_MultiBlock | Multi Block |
| NUM_TRANSFEROPERATIONMODE        |             |

**13.8.1.166 spinTransferQueueModeEnums**

enum `spinTransferQueueModeEnums`

< Specifies the operation mode of the transfer queue.

## Enumerator

|                                   |                                            |
|-----------------------------------|--------------------------------------------|
| TransferQueueMode_FirstInFirstOut | Blocks first In are transferred Out first. |
| NUM_TRANSFERQUEUEMODE             |                                            |

**13.8.1.167 spinTransferSelectorEnums**

enum `spinTransferSelectorEnums`

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

## Enumerator

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



**13.8.1.168 spinTransferStatusSelectorEnums**

enum `spinTransferStatusSelectorEnums`

< Selects which status of the transfer module to read.

**Enumerator**

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

**13.8.1.169 spinTransferTriggerActivationEnums**

enum `spinTransferTriggerActivationEnums`

< Specifies the activation mode of the transfer control trigger.

**Enumerator**

|                                                    |                                                                                                                                                               |
|----------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>TransferTriggerActivation_RisingEdge</code>  | Specifies that the trigger is considered valid on the rising edge of the source signal.                                                                       |
| <code>TransferTriggerActivation_FallingEdge</code> | Specifies that the trigger is considered valid on the falling edge of the source signal.                                                                      |
| <code>TransferTriggerActivation_AnyEdge</code>     | Specifies that the trigger is considered valid on the falling or rising edge of the source signal.                                                            |
| <code>TransferTriggerActivation_LevelHigh</code>   | 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. |
| <code>TransferTriggerActivation_LevelLow</code>    | 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.  |
| <code>NUM_TRANSFERTRIGGERACTIVATION</code>         |                                                                                                                                                               |

**13.8.1.170 spinTransferTriggerModeEnums**

enum `spinTransferTriggerModeEnums`

< Controls if the selected trigger is active.

## Enumerator

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

**13.8.1.171 spinTransferTriggerSelectorEnums**

```
enum spinTransferTriggerSelectorEnums
```

< Selects the type of transfer trigger to configure.

## Enumerator

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

**13.8.1.172 spinTransferTriggerSourceEnums**

```
enum spinTransferTriggerSourceEnums
```

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

## Enumerator

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

## Enumerator

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

## 13.8.1.173 spinTriggerActivationEnums

```
enum spinTriggerActivationEnums
```

< Specifies the activation mode of the trigger.

## Enumerator

|                               |  |
|-------------------------------|--|
| TriggerActivation_LevelLow    |  |
| TriggerActivation_LevelHigh   |  |
| TriggerActivation_FallingEdge |  |
| TriggerActivation_RisingEdge  |  |
| TriggerActivation_AnyEdge     |  |
| NUM_TRIGGERACTIVATION         |  |

### 13.8.1.174 spinTriggerModeEnums

enum `spinTriggerModeEnums`

< Controls whether or not trigger is active.

#### Enumerator

|                 |  |
|-----------------|--|
| TriggerMode_Off |  |
| TriggerMode_On  |  |
| NUM_TRIGGERMODE |  |

### 13.8.1.175 spinTriggerOverlapEnums

enum `spinTriggerOverlapEnums`

< Specifies the overlap mode of the trigger.

#### Enumerator

|                              |  |
|------------------------------|--|
| TriggerOverlap_Off           |  |
| TriggerOverlap_ReadOut       |  |
| TriggerOverlap_PreviousFrame |  |
| NUM_TRIGGEROVERLAP           |  |

### 13.8.1.176 spinTriggerSelectorEnums

enum `spinTriggerSelectorEnums`

< Selects the type of trigger to configure.

#### Enumerator

|                                  |  |
|----------------------------------|--|
| TriggerSelector_AcquisitionStart |  |
| TriggerSelector_FrameStart       |  |
| TriggerSelector_FrameBurstStart  |  |
| NUM_TRIGGERSELECTOR              |  |

**13.8.1.177 spinTriggerSourceEnums**

enum [spinTriggerSourceEnums](#)

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

**Enumerator**

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

**13.8.1.178 spinUserOutputSelectorEnums**

enum [spinUserOutputSelectorEnums](#)

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

**Enumerator**

|                                |  |
|--------------------------------|--|
| UserOutputSelector_UserOutput0 |  |
| UserOutputSelector_UserOutput1 |  |
| UserOutputSelector_UserOutput2 |  |
| UserOutputSelector_UserOutput3 |  |
| NUM_USEROUTPUTSELECTOR         |  |

**13.8.1.179 spinUserSetDefaultEnums**

enum [spinUserSetDefaultEnums](#)

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

**Enumerator**

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

**13.8.1.180 spinUserSetSelectorEnums**

enum `spinUserSetSelectorEnums`

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

**Enumerator**

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

**13.8.1.181 spinWhiteClipSelectorEnums**

enum `spinWhiteClipSelectorEnums`

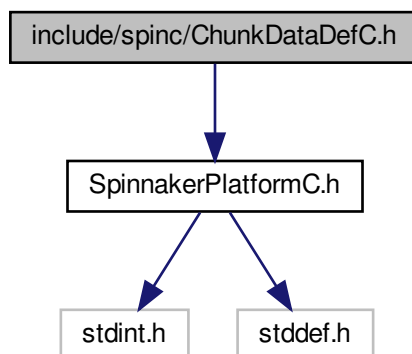
< Selects which White Clip to control.

**Enumerator**

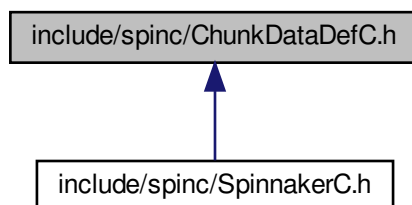
|                         |                                                     |
|-------------------------|-----------------------------------------------------|
| WhiteClipSelector_All   | White Clip will be applied to all channels or taps. |
| WhiteClipSelector_Red   | White Clip will be applied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

## 13.9 include/spinc/ChunkDataDefC.h File Reference

Include dependency graph for ChunkDataDefC.h:



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



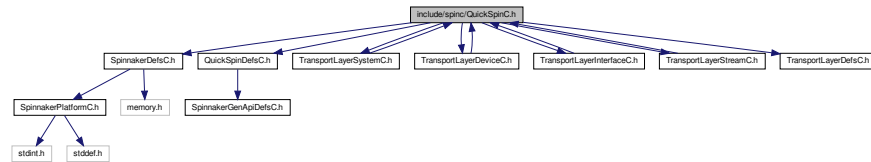
### Data Structures

- struct [spinChunkData](#)

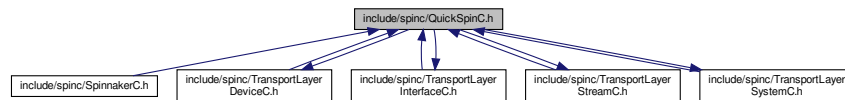
*The type of information that can be obtained from image chunk data.*

## 13.10 include/spinc/QuickSpinC.h File Reference

Include dependency graph for QuickSpinC.h:



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



### Functions

- [SPINNAKERC\\_API quickSpinInit](#) ([spinCamera](#) hCamera, [quickSpin](#) \*pQuickSpin)
- [SPINNAKERC\\_API quickSpinInitEx](#) ([spinCamera](#) hCamera, [quickSpin](#) \*pQuickSpin, [quickSpinTLDevice](#) \*pQuickSpinTLDevice, [quickSpinTLStream](#) \*pQuickSpinTLStream)
- [SPINNAKERC\\_API quickSpinTLDeviceInit](#) ([spinCamera](#) hCamera, [quickSpinTLDevice](#) \*pQuickSpin↔TLDevice)
- [SPINNAKERC\\_API quickSpinTLStreamInit](#) ([spinCamera](#) hCamera, [quickSpinTLStream](#) \*pQuickSpin↔TLStream)
- [SPINNAKERC\\_API quickSpinTLInterfaceInit](#) ([spinInterface](#) hInterface, [quickSpinTLInterface](#) \*pQuickSpin↔TLInterface)
- [SPINNAKERC\\_API quickSpinTLSystemInit](#) ([spinSystem](#) hSystem, [quickSpinTLSystem](#) \*pQuickSpin↔TLSystem)

### 13.10.1 Function Documentation

#### 13.10.1.1 quickSpinInit()

```

SPINNAKERC_API quickSpinInit (
 spinCamera hCamera,
 quickSpin * pQuickSpin)

```



### 13.10.1.2 quickSpinInitEx()

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

### 13.10.1.3 quickSpinTLDeviceInit()

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

### 13.10.1.4 quickSpinTLInterfaceInit()

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

### 13.10.1.5 quickSpinTLStreamInit()

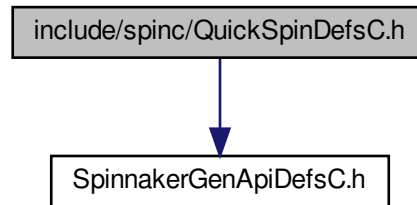
```
SPINNAKERC_API quickSpinTLStreamInit (
 spinCamera hCamera,
 quickSpinTLStream * pQuickSpinTLStream)
```

### 13.10.1.6 quickSpinTLSystemInit()

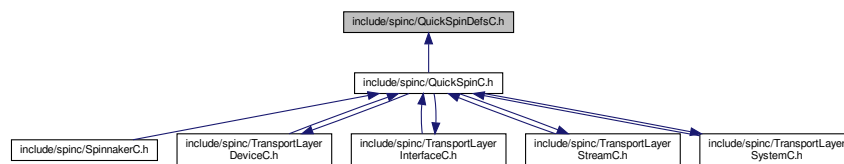
```
SPINNAKERC_API quickSpinTLSystemInit (
 spinSystem hSystem,
 quickSpinTLSystem * pQuickSpinTLSystem)
```

## 13.11 include/spinc/QuickSpinDefsC.h File Reference

Include dependency graph for QuickSpinDefsC.h:



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



### 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

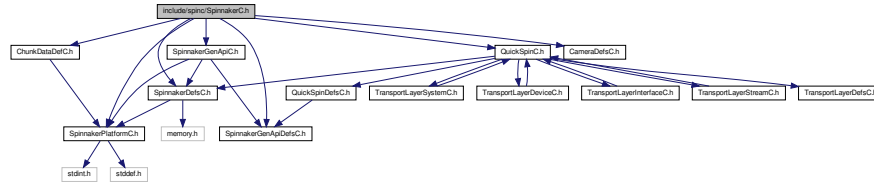
```
typedef spinNodeHandle quickSpinRegisterNode
```

#### 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](#) hLogEventHandler)  
*Registers a log event handler for the system.*

- Registers a logging event handler to the system (event handlers registered in this way must be unregistered)*

  - [SPINNAKERC\\_API spinSystemUnregisterLogEventHandler](#) ([spinSystem](#) hSystem, [spinLogEventHandler](#) hLogEventHandler)

*Unregisters a selected logging event handler from the system.*
- [SPINNAKERC\\_API spinSystemUnregisterAllLogEventHandlers](#) ([spinSystem](#) hSystem)

*Unregisters all logging event handlers from the system.*
- [SPINNAKERC\\_API spinSystemIsInUse](#) ([spinSystem](#) hSystem, [bool8\\_t](#) \*pbIsInUse)

*Checks whether a system is currently in use.*
- [SPINNAKERC\\_API spinSystemRegisterDeviceArrivalEventHandler](#) ([spinSystem](#) hSystem, [spinDeviceArrivalEventHandler](#) hDeviceArrivalEventHandler)

*Registers a device arrival event handler to every interface on the system (event handlers registered this way must be unregistered)*
- [SPINNAKERC\\_API spinSystemRegisterDeviceRemovalEventHandler](#) ([spinSystem](#) hSystem, [spinDeviceRemovalEventHandler](#) hDeviceRemovalEventHandler)

*Registers a device removal event handler to the system to every interface on the system (event handlers registered this way must be unregistered)*
- [SPINNAKERC\\_API spinSystemUnregisterDeviceArrivalEventHandler](#) ([spinSystem](#) hSystem, [spinDeviceArrivalEventHandler](#) hDeviceArrivalEventHandler)

*Unregisters a device arrival event handler from the system.*
- [SPINNAKERC\\_API spinSystemUnregisterDeviceRemovalEventHandler](#) ([spinSystem](#) hSystem, [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](#) 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.*
- [SPINNAKERC\\_API spinImageListCreateEmpty](#) ([spinImageList](#) \*phImageList)  
*Creates an empty image list (image lists created this way must be destroyed)*
- [SPINNAKERC\\_API spinImageListDestroy](#) ([spinImageList](#) hImageList)  
*Destroys a image list.*
- [SPINNAKERC\\_API spinImageListGetSize](#) ([spinImageList](#) hImageList, [size\\_t](#) \*pSize)  
*Retrieves the number of images in an image list.*
- [SPINNAKERC\\_API spinImageListGet](#) ([spinImageList](#) hImageList, [size\\_t](#) index, [spinImage](#) \*phImage)  
*Retrieves a image from a image list using an index.*
- [SPINNAKERC\\_API spinImageListClear](#) ([spinImageList](#) hImageList)  
*Clears a image list.*
- [SPINNAKERC\\_API spinImageListRemove](#) ([spinImageList](#) hImageList, [size\\_t](#) index)  
*Removes a image from a image list using its index.*
- [SPINNAKERC\\_API spinImageListAppend](#) ([spinImageList](#) hImageListBase, [spinImageList](#) hImageListToAppend)  
*Appends all the images from one image list to another.*
- [SPINNAKERC\\_API spinImageListGetByPixelFormat](#) ([spinImageList](#) hImageList, [spinPixelFormatEnums](#) pixelFormat, [spinImage](#) \*phImage)  
*Retrieves a image from a image list given its pixel format.*
- [SPINNAKERC\\_API spinImageListRemoveByPixelFormat](#) ([spinImageList](#) hImageList, [spinPixelFormatEnums](#) pixelFormat)  
*Removes a image from a image list using its pixel format.*
- [SPINNAKERC\\_API spinImageListRelease](#) ([spinImageList](#) hImageList)
- [SPINNAKERC\\_API spinImageListSave](#) ([spinImageList](#) hImageList, [const char](#) \*fileName)  
*Saves an image list as an object to a file.*
- [SPINNAKERC\\_API spinImageListLoad](#) ([spinImageList](#) \*phImageList, [const char](#) \*fileName)  
*Creates an image list object from file.*
- [SPINNAKERC\\_API spinInterfaceUpdateCameras](#) ([spinInterface](#) hInterface, [bool8\\_t](#) \*pbChanged)  
*Checks whether any cameras have been connected or disconnected on an interface.*

- [SPINNAKERC\\_API spinInterfaceGetCameras](#) ([spinInterface](#) hInterface, [spinCameraList](#) hCameraList)  
*Retrieves a camera list from an interface; camera lists must be created and destroy.*
- [SPINNAKERC\\_API spinInterfaceGetCamerasEx](#) ([spinInterface](#) hInterface, [bool8\\_t](#) bUpdateCameras, [spinCameraList](#) hCameraList)  
*Retrieves a camera list from an interface; manually set whether to update the cameras; camera lists must be created and destroyed.*
- [SPINNAKERC\\_API spinInterfaceGetTLNodeMap](#) ([spinInterface](#) hInterface, [spinNodeMapHandle](#) \*phNode↔Map)  
*Retrieves the transport layer nodemap from an interface.*
- [SPINNAKERC\\_API spinInterfaceRegisterDeviceArrivalEventHandler](#) ([spinInterface](#) hInterface, [spinDeviceArrivalEventHandler](#) hDeviceArrivalEventHandler)  
*Registers a device arrival event handler on an interface (event handlers registered in this way must be unregistered)*
- [SPINNAKERC\\_API spinInterfaceRegisterDeviceRemovalEventHandler](#) ([spinInterface](#) hInterface, [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.*
- [SPINNAKERC\\_API spinCameraInit](#) ([spinCamera](#) hCamera)  
*Initializes a camera, allowing for much more interaction.*
- [SPINNAKERC\\_API spinCameraDeInit](#) ([spinCamera](#) hCamera)  
*Deinitializes a camera, greatly reducing functionality.*
- [SPINNAKERC\\_API spinCameraGetNodeMap](#) ([spinCamera](#) hCamera, [spinNodeMapHandle](#) \*phNodeMap)  
*Retrieves the GenICam nodemap from a camera.*
- [SPINNAKERC\\_API spinCameraGetTLDeviceNodeMap](#) ([spinCamera](#) hCamera, [spinNodeMapHandle](#) \*ph↔NodeMap)  
*Retrieves the transport layer device nodemap from a camera.*
- [SPINNAKERC\\_API spinCameraGetTLStreamNodeMap](#) ([spinCamera](#) hCamera, [spinNodeMapHandle](#) \*ph↔NodeMap)  
*Retrieves the transport layer stream nodemap from a camera.*
- [SPINNAKERC\\_API spinCameraGetAccessMode](#) ([spinCamera](#) hCamera, [spinAccessMode](#) \*pAccessMode)  
*Retrieves the access mode of a camera (as an enum, spinAccessMode)*
- [SPINNAKERC\\_API spinCameraReadPort](#) ([spinCamera](#) hCamera, [uint64\\_t](#) iAddress, [void](#) \*pBuffer, [size\\_t](#) iSize)
- [SPINNAKERC\\_API spinCameraWritePort](#) ([spinCamera](#) hCamera, [uint64\\_t](#) iAddress, [void](#) \*pBuffer, [size\\_t](#) iSize)
- [SPINNAKERC\\_API spinCameraBeginAcquisition](#) ([spinCamera](#) hCamera)

- Has a camera start acquiring images.*

  - [SPINNAKERC\\_API spinCameraEndAcquisition](#) ([spinCamera](#) hCamera)
- Has a camera stop acquiring images.*

  - [SPINNAKERC\\_API spinCameraGetNextImage](#) ([spinCamera](#) hCamera, [spinImage](#) \*pImage)
- Retrieves an image from a camera.*

  - [SPINNAKERC\\_API spinCameraGetNextImageEx](#) ([spinCamera](#) hCamera, uint64\_t grabTimeout, [spinImage](#) \*pImage)
- Retrieves an image from a camera; manually set the timeout in milliseconds.*

  - [SPINNAKERC\\_API spinCameraGetNextImageSync](#) ([spinCamera](#) hCamera, uint64\_t grabTimeout, [spinImageList](#) \*pImageList)
- If a camera supports one or more streams, this function gets one image from each of the camera's streams, and returns the images in a list.*

  - [SPINNAKERC\\_API spinCameraGetUniqueID](#) ([spinCamera](#) hCamera, char \*pBuf, size\_t \*pBufLen)
- Retrieves a unique identifier for a camera.*

  - [SPINNAKERC\\_API spinCamerasStreaming](#) ([spinCamera](#) hCamera, [bool8\\_t](#) \*pbIsStreaming)
- Checks whether a camera is currently acquiring images.*

  - [SPINNAKERC\\_API spinCameraGetGuiXml](#) ([spinCamera](#) hCamera, char \*pBuf, size\_t \*pBufLen)
- Retrieves the GUI XML from a camera.*

  - [SPINNAKERC\\_API spinCameraRegisterDeviceEventHandler](#) ([spinCamera](#) hCamera, [spinDeviceEventHandler](#) hDeviceEventHandler)
- Registers a universal device event handler (every device event type) to a camera.*

  - [SPINNAKERC\\_API spinCameraRegisterDeviceEventHandlerEx](#) ([spinCamera](#) hCamera, [spinDeviceEventHandler](#) hDeviceEventHandler, const char \*pName)
- Registers a specific device event handler (only one device event type) to a camera.*

  - [SPINNAKERC\\_API spinCameraUnregisterDeviceEventHandler](#) ([spinCamera](#) hCamera, [spinDeviceEventHandler](#) hDeviceEventHandler)
- Unregisters a device event handler from a camera.*

  - [SPINNAKERC\\_API spinCameraRegisterImageEventHandler](#) ([spinCamera](#) hCamera, [spinImageEventHandler](#) hImageEventHandler)
- Registers an image event handler to a camera.*

  - [SPINNAKERC\\_API spinCameraRegisterImageEventHandlerEx](#) ([spinCamera](#) hCamera, [spinImageEventHandler](#) hImageEventHandler, uint64\_t streamIndex)
- Registers an image event handler to a camera Registers a specific stream handler for the camera given a stream index.*

  - [SPINNAKERC\\_API spinCameraUnregisterImageEventHandler](#) ([spinCamera](#) hCamera, [spinImageEventHandler](#) hImageEventHandler)
- Unregisters an image event handler from a camera.*

  - [SPINNAKERC\\_API spinCameraRegisterImageListEventHandler](#) ([spinCamera](#) hCamera, [spinImageListEventHandler](#) hImageListEventHandler)
- Registers an image list event handler to a camera.*

  - [SPINNAKERC\\_API spinCameraUnregisterImageListEventHandler](#) ([spinCamera](#) hCamera, [spinImageListEventHandler](#) hImageListEventHandler)
- Unregisters an image list event handler from a camera.*

  - [SPINNAKERC\\_API spinCameraRelease](#) ([spinCamera](#) hCamera)
- Releases a camera.*

  - [SPINNAKERC\\_API 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.*

  - [SPINNAKERC\\_API spinCameraDiscoverMaxPacketSize](#) ([spinCamera](#) hCamera, unsigned int \*pMax↔PacketSize)
- Returns the largest packet size that can be safely used on the interface that device is connected to.*

  - [SPINNAKERC\\_API spinCameraForceIP](#) ()



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

  - [SPINNAKERC\\_API spinImageCreateEmpty](#) ([spinImage](#) \*phImage)

*Creates an empty image; images created this way must be destroyed.*
- [SPINNAKERC\\_API spinImageCreate](#) ([spinImage](#) hSrcImage, [spinImage](#) \*phDestImage)

*Creates an image from another; images created this way must be destroyed.*
- [SPINNAKERC\\_API spinImageCreateEx](#) ([spinImage](#) \*phImage, [size\\_t](#) width, [size\\_t](#) height, [size\\_t](#) offsetX, [size\\_t](#) offsetY, [spinPixelFormatEnums](#) pixelFormat, void \*pData)

*Creates an image with some set properties; images created this way must be destroyed.*
- [SPINNAKERC\\_API spinImageCreateEx2](#) ([spinImage](#) \*phImage, [size\\_t](#) width, [size\\_t](#) height, [size\\_t](#) offsetX, [size\\_t](#) offsetY, [spinPixelFormatEnums](#) pixelFormat, void \*pData, [spinTLPayloadType](#) dataPayloadType, [size\\_t](#) dataSize)

*Creates an image with some set properties; images created this way must be destroyed.*
- [SPINNAKERC\\_API spinImageDestroy](#) ([spinImage](#) hImage)

*Destroys an image.*
- [SPINNAKERC\\_API spinImageGetColorProcessing](#) ([spinImage](#) hImage, [spinColorProcessingAlgorithm](#) \*pAlgorithm)

*Retrieves the color processing algorithm of a specific image.*
- [SPINNAKERC\\_API spinImageReset](#) ([spinImage](#) hImage, [size\\_t](#) width, [size\\_t](#) height, [size\\_t](#) offsetX, [size\\_t](#) offsetY, [spinPixelFormatEnums](#) pixelFormat)

*Resets an image with some set properties.*
- [SPINNAKERC\\_API spinImageResetEx](#) ([spinImage](#) hImage, [size\\_t](#) width, [size\\_t](#) height, [size\\_t](#) offsetX, [size\\_t](#) offsetY, [spinPixelFormatEnums](#) pixelFormat, void \*pData)

*Resets an image with some set properties and image data.*
- [SPINNAKERC\\_API spinImageGetID](#) ([spinImage](#) hImage, [uint64\\_t](#) \*pId)

*Retrieves the ID of an image.*
- [SPINNAKERC\\_API spinImageGetData](#) ([spinImage](#) hImage, void \*\*ppData)

*Retrieves the image data of an image.*
- [SPINNAKERC\\_API spinImageGetPrivateData](#) ([spinImage](#) hImage, void \*\*ppData)

*Retrieves the private data of an image.*
- [SPINNAKERC\\_API spinImageGetBufferSize](#) ([spinImage](#) hImage, [size\\_t](#) \*pSize)

*Retrieves the buffer size of an image.*
- [SPINNAKERC\\_API spinImageDeepCopy](#) ([spinImage](#) hSrcImage, [spinImage](#) hDestImage)

*Creates a deep copy of an image (the destination image must be created as an empty image prior to the deep copy)*
- [SPINNAKERC\\_API spinImageGetWidth](#) ([spinImage](#) hImage, [size\\_t](#) \*pWidth)

*Retrieves the width of an image.*
- [SPINNAKERC\\_API spinImageGetHeight](#) ([spinImage](#) hImage, [size\\_t](#) \*pHeight)

*Retrieves the height of an image.*
- [SPINNAKERC\\_API spinImageGetOffsetX](#) ([spinImage](#) hImage, [size\\_t](#) \*pOffsetX)

*Retrieves the offset of an image along its X axis.*
- [SPINNAKERC\\_API spinImageGetOffsetY](#) ([spinImage](#) hImage, [size\\_t](#) \*pOffsetY)

*Retrieves the offset of an image along its Y axis.*
- [SPINNAKERC\\_API spinImageGetPaddingX](#) ([spinImage](#) hImage, [size\\_t](#) \*pPaddingX)

*Retrieves the padding of an image along its X axis.*
- [SPINNAKERC\\_API spinImageGetPaddingY](#) ([spinImage](#) hImage, [size\\_t](#) \*pPaddingY)

*Retrieves the padding of an image along its Y axis.*
- [SPINNAKERC\\_API spinImageGetFrameID](#) ([spinImage](#) hImage, [uint64\\_t](#) \*pFrameID)

*Retrieves the frame ID of an image.*
- [SPINNAKERC\\_API spinImageGetTimeStamp](#) ([spinImage](#) hImage, [uint64\\_t](#) \*pTimeStamp)

*Retrieves the timestamp of an image.*
- [SPINNAKERC\\_API spinImageGetPayloadType](#) ([spinImage](#) hImage, [size\\_t](#) \*pPayloadType)

*Retrieves the payload type of an image (as an enum, [spinPayloadTypeInfolds](#))*

- [SPINNAKERC\\_API spinImageGetTLPayloadType](#) ([spinImage](#) hImage, [spinTLPayloadType](#) \*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, [spinTLPixelFormatNamespace](#) \*pPixelFormatNamespace)  
*Retrieves the transport layer pixel format namespace of an image (as an enum, [spinPixelFormatNamespaceID](#))*
- [SPINNAKERC\\_API spinImageGetPixelFormatName](#) ([spinImage](#) hImage, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves the pixel format of an image (as a symbolic)*
- [SPINNAKERC\\_API spinImageIsIncomplete](#) ([spinImage](#) hImage, [bool8\\_t](#) \*pblsIncomplete)  
*Checks whether an image is incomplete.*
- [SPINNAKERC\\_API spinImageGetValidPayloadSize](#) ([spinImage](#) hImage, [size\\_t](#) \*pSize)  
*Retrieves the valid payload size of an image.*
- [SPINNAKERC\\_API spinImageSave](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [spinImageFileFormat](#) format)  
*Saves an image using a specified file format (using an enum, [spinImageFileFormat](#))*
- [SPINNAKERC\\_API spinImageSaveFromExt](#) ([spinImage](#) hImage, [const char](#) \*pFilename)  
*Saves an image using a specified file format (using the extension of the filename)*
- [SPINNAKERC\\_API spinImageSavePng](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinPNGOption](#) \*pOption)  
*Saves an image as a PNG image.*
- [SPINNAKERC\\_API spinImageSavePpm](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinPPMOption](#) \*pOption)  
*Saves an image as a PPM image.*
- [SPINNAKERC\\_API spinImageSavePgm](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinPGMOption](#) \*pOption)  
*Saves an image as an PGM image.*
- [SPINNAKERC\\_API spinImageSaveTiff](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinTIFFOption](#) \*pOption)  
*Saves an image as a TIFF image.*
- [SPINNAKERC\\_API spinImageSaveJpeg](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinJPEGOption](#) \*pOption)  
*Saves an image as a JPEG image.*
- [SPINNAKERC\\_API spinImageSaveJpg2](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinJPG2Option](#) \*pOption)  
*Saves an image as a JPEG 2000 image.*
- [SPINNAKERC\\_API spinImageSaveBmp](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinBMPOption](#) \*pOption)  
*Saves an image as a BMP image.*
- [SPINNAKERC\\_API spinImageGetChunkLayoutID](#) ([spinImage](#) hImage, [uint64\\_t](#) \*pId)  
*Retrieves the chunk layout ID of an image.*
- [SPINNAKERC\\_API spinImageCalculateStatistics](#) ([spinImage](#) hImage, [const spinImageStatistics](#) hStatistics)  
*Calculates the image statistics of an image.*
- [SPINNAKERC\\_API spinImageGetStatus](#) ([spinImage](#) hImage, [spinImageStatus](#) \*pStatus)  
*Retrieves the image status of an image.*
- [SPINNAKERC\\_API spinImageGetStatusDescription](#) ([spinImageStatus](#) status, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves the description of image status.*
- [SPINNAKERC\\_API spinImageRelease](#) ([spinImage](#) hImage)  
*Releases an image.*
- [SPINNAKERC\\_API spinImageHasCRC](#) ([spinImage](#) hImage, [bool8\\_t](#) \*pbHasCRC)

- Checks whether an image has CRC.*

  - [SPINNAKERC\\_API spinImageCheckCRC](#) ([spinImage](#) hImage, [bool8\\_t](#) \*pbCheckCRC)

*Checks whether the CRC of an image is correct.*
- [SPINNAKERC\\_API spinImageGetBitsPerPixel](#) ([spinImage](#) hImage, [size\\_t](#) \*pBitsPerPixel)

*Retrieves the number of bits per pixel of an image.*
- [SPINNAKERC\\_API spinImageGetSize](#) ([spinImage](#) hImage, [size\\_t](#) \*pImageSize)

*Retrieves the size of an image.*
- [SPINNAKERC\\_API spinImageGetStride](#) ([spinImage](#) hImage, [size\\_t](#) \*pStride)

*Retrieves the stride of an image.*
- [SPINNAKERC\\_API spinImageProcessorCreate](#) ([spinImageProcessor](#) \*phImageProcessor)

*Creates an image processor.*
- [SPINNAKERC\\_API spinImageProcessorDestroy](#) ([spinImageProcessor](#) hImageProcessor)

*Destroys a image list.*
- [SPINNAKERC\\_API spinImageProcessorSetColorProcessing](#) ([spinImageProcessor](#) hImageProcessor, [spinColorProcessingAlgorithm](#) colorAlgorithm)

*Sets the color processing algorithm used at the time of the [spinImageProcessorConvert\(\)](#) call, therefore the most recent execution of this function will take precedence.*
- [SPINNAKERC\\_API spinImageProcessorGetColorProcessing](#) ([spinImageProcessor](#) hImageProcessor, [spinColorProcessingAlgorithm](#) \*pColorAlgorithm)

*Gets the default color processing algorithm.*
- [SPINNAKERC\\_API spinImageProcessorSetNumDecompressionThreads](#) ([spinImageProcessor](#) hImage↵Processor, unsigned int numThreads)

*Sets the default number of threads used for image decompression during [spinImageProcessorConvert\(\)](#).*
- [SPINNAKERC\\_API spinImageProcessorGetNumDecompressionThreads](#) ([spinImageProcessor](#) hImage↵Processor, unsigned int \*pNumThreads)

*Gets the number of threads used for image decompression during [spinImageProcessorConvert\(\)](#).*
- [SPINNAKERC\\_API spinImageProcessorConvert](#) ([spinImageProcessor](#) hImageProcessor, [spinImage](#) hSrc↵Image, [spinImage](#) hDestImage, [spinPixelFormatEnums](#) destFormat)

*Converts the source image buffer to the specified destination pixel format and stores the result in the destination image.*
- [SPINNAKERC\\_API spinImageProcessorConvertImageList](#) ([spinImageProcessor](#) hImageProcessor, [spinImageList](#) hSrcImageList, [spinImage](#) hDestImage, [spinPixelFormatEnums](#) destFormat)

*Converts the source list of image buffers to the specified output pixel format and returns the result in a new image.*
- [SPINNAKERC\\_API spinImageProcessorApplyGamma](#) ([spinImageProcessor](#) hImageProcessor, [spinImage](#) hSrcImage, [spinImage](#) hDestImage, float gamma, [bool8\\_t](#) applyGammaInverse)

*Applies gamma correction to the source image and stores the result in the destination image.*
- [SPINNAKERC\\_API spinDeviceEventHandlerCreate](#) ([spinDeviceEventHandler](#) \*phDeviceEventHandler, [spinDeviceEventFunction](#) pFunction, void \*pUserData)

*Creates a device event handler.*
- [SPINNAKERC\\_API spinDeviceEventHandlerDestroy](#) ([spinDeviceEventHandler](#) hDeviceEventHandler)

*Destroys a device event handler.*
- [SPINNAKERC\\_API spinImageEventHandlerCreate](#) ([spinImageEventHandler](#) \*phImageEventHandler, [spinImageEventFunction](#) pFunction, void \*pUserData)

*Creates an image event handler.*
- [SPINNAKERC\\_API spinImageEventHandlerDestroy](#) ([spinImageEventHandler](#) hImageEventHandler)

*Destroys an image event handler.*
- [SPINNAKERC\\_API spinImageListEventHandlerCreate](#) ([spinImageListEventHandler](#) \*phImageEventHandler, [spinImageListEventFunction](#) pFunction, void \*pUserData)

*Creates an image list event handler.*
- [SPINNAKERC\\_API spinImageListEventHandlerDestroy](#) ([spinImageListEventHandler](#) hImageListEvent↵Handler)

*Destroys an image list event handler.*

- [SPINNAKERC\\_API spinDeviceArrivalEventHandlerCreate](#) ([spinDeviceArrivalEventHandler](#) \*phDevice↔ArrivalEventHandler, [spinArrivalEventFunction](#) pFunction, void \*pUserData)  
*Creates a device arrival event handler.*
- [SPINNAKERC\\_API spinDeviceArrivalEventHandlerDestroy](#) ([spinDeviceArrivalEventHandler](#) hDevice↔ArrivalEventHandler)  
*Destroys a device arrival event handler.*
- [SPINNAKERC\\_API spinDeviceRemovalEventHandlerCreate](#) ([spinDeviceRemovalEventHandler](#) \*phDevice↔RemovalEventHandler, [spinRemovalEventFunction](#) pFunction, void \*pUserData)  
*Creates a device removal event handler.*
- [SPINNAKERC\\_API spinDeviceRemovalEventHandlerDestroy](#) ([spinDeviceRemovalEventHandler](#) hDevice↔RemovalEventHandler)  
*Destroys a device removal event handler.*
- [SPINNAKERC\\_API spinInterfaceEventHandlerCreate](#) ([spinInterfaceEventHandler](#) \*phInterfaceEvent↔Handler, [spinArrivalEventFunction](#) pArrivalFunction, [spinRemovalEventFunction](#) pRemovalFunction, void \*pUserData)  
*Creates an interface event handler (both device arrival and device removal)*
- [SPINNAKERC\\_API spinInterfaceEventHandlerDestroy](#) ([spinInterfaceEventHandler](#) hInterfaceEventHandler)  
*Destroys an interface event handler (both device arrival and device removal)*
- [SPINNAKERC\\_API spinLogEventHandlerCreate](#) ([spinLogEventHandler](#) \*phLogEventHandler, [spinLogEventFunction](#) pFunction, void \*pUserData)  
*Creates a log event handler.*
- [SPINNAKERC\\_API spinLogEventHandlerDestroy](#) ([spinLogEventHandler](#) hLogEventHandler)  
*Destroys a log event handler.*
- [SPINNAKERC\\_API spinImageStatisticsCreate](#) ([spinImageStatistics](#) \*phStatistics)  
*Creates an image statistics context.*
- [SPINNAKERC\\_API spinImageStatisticsDestroy](#) ([spinImageStatistics](#) hStatistics)  
*Destroys an image statistics context.*
- [SPINNAKERC\\_API spinImageStatisticsEnableAll](#) ([spinImageStatistics](#) hStatistics)  
*Enables all channels of an image statistics context.*
- [SPINNAKERC\\_API spinImageStatisticsDisableAll](#) ([spinImageStatistics](#) hStatistics)  
*Disables all channels of an image statistics context.*
- [SPINNAKERC\\_API spinImageStatisticsEnableGreyOnly](#) ([spinImageStatistics](#) hStatistics)  
*Disables all channels of an image statistics context except grey-scale.*
- [SPINNAKERC\\_API spinImageStatisticsEnableRgbOnly](#) ([spinImageStatistics](#) hStatistics)  
*Disables all channels of an image statistics context except red, blue, and green.*
- [SPINNAKERC\\_API spinImageStatisticsEnableHslOnly](#) ([spinImageStatistics](#) hStatistics)  
*Disables all channels of an image statistics context except hue, saturation, and lightness.*
- [SPINNAKERC\\_API spinImageStatisticsGetChannelStatus](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, [bool8\\_t](#) \*pbEnabled)  
*Checks whether an image statistics context is enabled.*
- [SPINNAKERC\\_API spinImageStatisticsSetChannelStatus](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, [bool8\\_t](#) bEnable)  
*Sets the status of an image statistics channel.*
- [SPINNAKERC\\_API spinImageStatisticsGetRange](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, unsigned int \*pMin, unsigned int \*pMax)  
*Retrieves the range of an image statistics channel.*
- [SPINNAKERC\\_API spinImageStatisticsGetPixelValueRange](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, unsigned int \*pMin, unsigned int \*pMax)  
*Retrieves the pixel value range of an image statistics channel.*
- [SPINNAKERC\\_API spinImageStatisticsGetNumPixelValues](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, unsigned int \*pNumValues)  
*Retrieves the number of pixel values of an image statistics channel.*

- [SPINNAKERC\\_API spinImageStatisticsGetMean](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, float \*pMean)  
*Retrieves the mean of pixel values of an image statistics channel.*
- [SPINNAKERC\\_API spinImageStatisticsGetHistogram](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, int \*\*ppHistogram)  
*Retrieves a histogram of an image statistics channel.*
- [SPINNAKERC\\_API spinImageStatisticsGetAll](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, unsigned int \*pRangeMin, unsigned int \*pRangeMax, unsigned int \*pPixelValueMin, unsigned int \*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)
- [SPINNAKERC\\_API spinImageChunkDataGetFloatValue](#) ([spinImage](#) hImage, const char \*pName, double \*pValue)

### 13.12.1 Function Documentation

### 13.12.1.1 spinCameraBeginAcquisition()

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

Has a camera start acquiring images.

See also

[spinError](#)

#### Parameters

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

#### Returns

spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

### 13.12.1.2 spinCameraDeInit()

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

Deinitializes a camera, greatly reducing functionality.

See also

[spinError](#)

#### Parameters

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

#### Returns

spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

### 13.12.1.3 spinCameraDiscoverMaxPacketSize()

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

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

See also

[spinError](#)

Parameters

|                       |                                  |
|-----------------------|----------------------------------|
| <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

#### 13.12.1.4 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

#### 13.12.1.5 spinCameraForceIP()

```
SPINNAKERC_API spinCameraForceIP ()
```

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

Returns

*spinError* The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

### 13.12.1.6 spinCameraGetAccessMode()

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

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

See also

[spinError](#)

[spinAccessMode](#)

#### Parameters

|                    |                                                                   |
|--------------------|-------------------------------------------------------------------|
| <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

### 13.12.1.7 spinCameraGetGuiXml()

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

Retrieves the GUI XML from a camera.

See also

[spinError](#)

#### Parameters

|                |                                                                                                                     |
|----------------|---------------------------------------------------------------------------------------------------------------------|
| <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



### 13.12.1.8 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

### 13.12.1.9 spinCameraGetNextImageEx()

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

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

See also

[spinError](#)

#### Parameters

|                    |                                                         |
|--------------------|---------------------------------------------------------|
| <i>hCamera</i>     | The camera of the image to retrieve                     |
| <i>grabTimeout</i> | A 64bit value that represents a timeout in milliseconds |
| <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

### 13.12.1.10 spinCameraGetNextImageSync()

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

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

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

#### See also

[spinCameraInit\(\)](#)  
[spinCameraBeginAcquisition\(\)](#)  
[spinCameraEndAcquisition\(\)](#)

#### Parameters

|                    |                                                         |
|--------------------|---------------------------------------------------------|
| <i>hCamera</i>     | The camera of the image to retrieve                     |
| <i>grabTimeout</i> | A 64bit value that represents a timeout in milliseconds |

#### Returns

spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

### 13.12.1.11 spinCameraGetNodeMap()

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

Retrieves the GenICam nodemap from a camera.

#### See also

[spinError](#)

#### Parameters

|                  |                                                             |
|------------------|-------------------------------------------------------------|
| <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

### 13.12.1.12 spinCameraGetTLDeviceNodeMap()

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

Retrieves the transport layer device nodemap from a camera.

See also

[spinError](#)

#### Parameters

|                  |                                                                       |
|------------------|-----------------------------------------------------------------------|
| <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

### 13.12.1.13 spinCameraGetTLStreamNodeMap()

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

Retrieves the transport layer stream nodemap from a camera.

See also

[spinError](#)

#### Parameters

|                  |                                                                          |
|------------------|--------------------------------------------------------------------------|
| <i>hCamera</i>   | The camera from which the transport layer streaming 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

#### 13.12.1.14 spinCameraGetUniqueID()

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

Retrieves a unique identifier for a camera.

See also

[spinError](#)

##### Parameters

|                |                                                                                                                     |
|----------------|---------------------------------------------------------------------------------------------------------------------|
| <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

#### 13.12.1.15 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

### 13.12.1.16 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

### 13.12.1.17 spinCameraIsStreaming()

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

Checks whether a camera is currently acquiring images.

See also

[spinError](#)

#### Parameters

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| <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

### 13.12.1.18 spinCameraIsValid()

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

Checks whether a camera is still valid for use.

See also

[spinError](#)

Parameters

|                |                                                                  |
|----------------|------------------------------------------------------------------|
| <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

#### 13.12.1.19 `spinCameraListAppend()`

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

Appends all the cameras from one camera list to another.

See also

[spinError](#)

Parameters

|                            |                                      |
|----------------------------|--------------------------------------|
| <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

#### 13.12.1.20 `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

**13.12.1.21 spinCameraListCreateEmpty()**

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

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

## See also

[spinError](#)

## Parameters

|                     |                                                                           |
|---------------------|---------------------------------------------------------------------------|
| <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

**13.12.1.22 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

**13.12.1.23 spinCameraListGet()**

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

Retrieves a camera from a camera list using an index.

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

**See also**

[spinError](#)

**Parameters**

|                    |                                                           |
|--------------------|-----------------------------------------------------------|
| <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

**13.12.1.24 spinCameraListGetBySerial()**

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

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

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

**See also**

[spinError](#)



## Parameters

|                    |                                                           |
|--------------------|-----------------------------------------------------------|
| <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

**13.12.1.25 spinCameraListGetSize()**

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

Retrieves the number of cameras on a camera list.

## See also

[spinError](#)

## Parameters

|                    |                                                                         |
|--------------------|-------------------------------------------------------------------------|
| <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

**13.12.1.26 spinCameraListRemove()**

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

Removes a camera from a camera list using its index.

## See also

[spinError](#)

## Parameters

|                    |                                         |
|--------------------|-----------------------------------------|
| <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

**13.12.1.27 spinCameraListRemoveBySerial()**

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

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

## See also

[spinError](#)

## Parameters

|                    |                                           |
|--------------------|-------------------------------------------|
| <i>hCameraList</i> | The camera list 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

**13.12.1.28 spinCameraReadPort()**

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

### 13.12.1.29 spinCameraRegisterDeviceEventHandler()

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

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

See also

[spinError](#)

#### Parameters

|                            |                                                                    |
|----------------------------|--------------------------------------------------------------------|
| <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

### 13.12.1.30 spinCameraRegisterDeviceEventHandlerEx()

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

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

See also

[spinError](#)

#### Parameters

|                            |                                                                   |
|----------------------------|-------------------------------------------------------------------|
| <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

### 13.12.1.31 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

### 13.12.1.32 spinCameraRegisterImageEventHandlerEx()

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

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

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

See also

[spinError](#)

#### Parameters

|                           |                                                                     |
|---------------------------|---------------------------------------------------------------------|
| <i>hCamera</i>            | The camera on which to register the image event handler             |
| <i>hImageEventHandler</i> | The image event handler to register                                 |
| <i>streamIndex</i>        | The index of the stream of where this handler will be registered to |

#### Returns

spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

### 13.12.1.33 spinCameraRegisterImageListEventHandler()

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

Registers an image list event handler to a camera.

See also

[spinError](#)

Parameters

|                               |                                                         |
|-------------------------------|---------------------------------------------------------|
| <i>hCamera</i>                | The camera on which to register the image event handler |
| <i>hImageListEventHandler</i> | The image list event handler to register                |

Returns

spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

### 13.12.1.34 spinCameraRelease()

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

Releases a camera.

See also

[spinError](#)

Parameters

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

Returns

spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

### 13.12.1.35 spinCameraUnregisterDeviceEventHandler()

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

Unregisters a device event handler from a camera.

See also

[spinError](#)

#### Parameters

|                            |                                                              |
|----------------------------|--------------------------------------------------------------|
| <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

### 13.12.1.36 `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

### 13.12.1.37 `spinCameraUnregisterImageListEventHandler()`

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

Unregisters an image list event handler from a camera.

See also

[spinError](#)

## Parameters

|                           |                                                             |
|---------------------------|-------------------------------------------------------------|
| <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

**13.12.1.38 spinCameraWritePort()**

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

**13.12.1.39 spinDeviceArrivalEventHandlerCreate()**

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

Creates a device arrival event handler.

## See also

[spinError](#)

## Parameters

|                                    |                                                                                                                                                  |
|------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| <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: <code>void(&lt;em&gt;spinArrivalEventFunction)(void pUserData)</code> |
| <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

#### 13.12.1.40 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

#### 13.12.1.41 spinDeviceEventGetId()

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

Retrieves the event ID of a device event.

See also

[spinError](#)

Parameters

|                         |                                                                |
|-------------------------|----------------------------------------------------------------|
| <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

#### 13.12.1.42 spinDeviceEventGetName()

```
SPINNAKERC_API spinDeviceEventGetName (
 spinDeviceEventData hDeviceEventData,
```



```
char * pBuf,
size_t * pBufLen)
```

Retrieves the event name of a device event.

See also

[spinError](#)

#### Parameters

|                         |                                                                                                                     |
|-------------------------|---------------------------------------------------------------------------------------------------------------------|
| <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

#### 13.12.1.43 spinDeviceEventGetPayloadData()

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

Retrieves the payload data of a device event.

See also

[spinError](#)

#### Parameters

|                         |                                                                           |
|-------------------------|---------------------------------------------------------------------------|
| <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

#### 13.12.1.44 spinDeviceEventGetPayloadDataSize()

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

Retrieves the payload data size of a device event.

See also

[spinError](#)

##### Parameters

|                         |                                                                           |
|-------------------------|---------------------------------------------------------------------------|
| <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

#### 13.12.1.45 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:<br>void(<em>spinDeviceEventFunction)(const spinDeviceEventData hEventData,<br>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

#### 13.12.1.46 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

#### 13.12.1.47 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(<em>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

**13.12.1.48 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

**13.12.1.49 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

**13.12.1.50 spinErrorGetLastBuildDate()**

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

Retrieves the build date of the last error.

See also

[spinError](#)

#### Parameters

|                |                                                                                                                     |
|----------------|---------------------------------------------------------------------------------------------------------------------|
| <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

### 13.12.1.51 spinErrorGetLastBuildTime()

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

Retrieves the build time of the last error.

See also

[spinError](#)

#### Parameters

|                |                                                                                                                     |
|----------------|---------------------------------------------------------------------------------------------------------------------|
| <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

### 13.12.1.52 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

**13.12.1.53 spinErrorGetLastFullMessage()**

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

Retrieves the full error message of the last error.

## See also

[spinError](#)

## Parameters

|                |                                                                                                                     |
|----------------|---------------------------------------------------------------------------------------------------------------------|
| <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

**13.12.1.54 spinErrorGetLastFunctionName()**

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

Retrieves the function name of the last error.

## See also

[spinError](#)

## Parameters

|                |                                                                                                                     |
|----------------|---------------------------------------------------------------------------------------------------------------------|
| <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

**13.12.1.55 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

**13.12.1.56 spinErrorGetLastMessage()**

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

Retrieves the error message of the last error.

## See also

[spinError](#)

## Parameters

|                |                                                                                                                     |
|----------------|---------------------------------------------------------------------------------------------------------------------|
| <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

**13.12.1.57 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

**13.12.1.58 spinImageCheckCRC()**

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

Checks whether the CRC of an image is correct.

## See also

[spinError](#)



## Parameters

|                   |                                                            |
|-------------------|------------------------------------------------------------|
| <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

**13.12.1.59 spinImageChunkDataGetFloatValue()**

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

**13.12.1.60 spinImageChunkDataGetIntValue()**

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

**13.12.1.61 spinImageCreate()**

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

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

## See also

[spinError](#)

## Parameters

|                    |                                                     |
|--------------------|-----------------------------------------------------|
| <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

**13.12.1.62 spinImageCreateEmpty()**

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

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

**See also**

[spinError](#)

**Parameters**

|                |                                                               |
|----------------|---------------------------------------------------------------|
| <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

**13.12.1.63 spinImageCreateEx()**

```
SPINNAKERC_API spinImageCreateEx (
 spinImage * phImage,
 size_t width,
 size_t height,
 size_t offsetX,
 size_t offsetY,
 spinPixelFormatEnums pixelFormat,
 void * pData)
```

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

**See also**

[spinError](#)

**Parameters**

|                    |                                                         |
|--------------------|---------------------------------------------------------|
| <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

13.12.1.64 `spinImageCreateEx2()`

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

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

## See also

[spinError](#)

[spinImageGetTLPayloadType](#)

## Parameters

|                        |                                                                                                                                                          |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <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 <a href="#">spinImageGetTLPayloadType()</a> 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

13.12.1.65 `spinImageDeepCopy()`

```
SPINNAKERC_API spinImageDeepCopy (
 spinImage hSrcImage,
 spinImage hDestImage)
```

Creates a deep copy of an image (the destination image must be created as an empty image prior to the deep copy)

See also

[spinError](#)

Parameters

|                   |                                               |
|-------------------|-----------------------------------------------|
| <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

#### 13.12.1.66 [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

#### 13.12.1.67 [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:<br>void(<em>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

**13.12.1.68 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

**13.12.1.69 spinImageGetBitsPerPixel()**

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

Retrieves the number of bits per pixel of an image.

## See also

[spinError](#)

## Parameters

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| <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

**13.12.1.70 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

**13.12.1.71 spinImageGetChunkLayoutID()**

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

Retrieves the chunk layout ID of an image.

## See also

[spinError](#)

## Parameters

|               |                                                                       |
|---------------|-----------------------------------------------------------------------|
| <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

**13.12.1.72 spinImageGetColorProcessing()**

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

Retrieves the color processing algorithm of a specific image.

## See also

[spinError](#)

## Parameters

|                   |                                                                                            |
|-------------------|--------------------------------------------------------------------------------------------|
| <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

**13.12.1.73 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

**13.12.1.74 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

**13.12.1.75 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

**13.12.1.76 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

**13.12.1.77 spinImageGetOffsetX()**

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

Retrieves the offset of an image along its X axis.

## See also

[spinError](#)

## Parameters

|                 |                                                                               |
|-----------------|-------------------------------------------------------------------------------|
| <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

**13.12.1.78 spinImageGetOffsetY()**

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

Retrieves the offset of an image along its Y axis.

## See also

[spinError](#)

## Parameters

|                 |                                                                               |
|-----------------|-------------------------------------------------------------------------------|
| <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

**13.12.1.79 spinImageGetPaddingX()**

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

Retrieves the padding of an image along its X axis.

## See also

[spinError](#)

## Parameters

|                  |                                                                                |
|------------------|--------------------------------------------------------------------------------|
| <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

**13.12.1.80 spinImageGetPaddingY()**

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

Retrieves the padding of an image along its Y axis.

## See also

[spinError](#)

## Parameters

|                  |                                                                                |
|------------------|--------------------------------------------------------------------------------|
| <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

**13.12.1.81 spinImageGetPayloadType()**

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

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

## See also

[spinError](#)

[spinPayloadTypeInfolds](#)

## 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

**13.12.1.82 spinImageGetPixelFormat()**

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

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

## See also

[spinError](#)

[spinPixelFormatEnums](#)

## Parameters

|                     |                                                                     |
|---------------------|---------------------------------------------------------------------|
| <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

**13.12.1.83 spinImageGetPixelFormatName()**

```
SPINNAKERC_API spinImageGetPixelFormatName (
 spinImage hImage,
 char * pBuf,
 size_t * pBufLen)
```

Retrieves the pixel format of an image (as a symbolic)

## See also

[spinError](#)

## Parameters

|                |                                                                                                                     |
|----------------|---------------------------------------------------------------------------------------------------------------------|
| <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

**13.12.1.84 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

**13.12.1.85 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

**13.12.1.86 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

**13.12.1.87 spinImageGetStatusDescription()**

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

Retrieves the description of image status.

## See also

[spinError](#)

## Parameters

|                |                                                                                                                                                                                   |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <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 pBuf is NULL, minimum length of string buffer is returned |

## Returns

spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

**13.12.1.88 spinImageGetStride()**

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

Retrieves the stride of an image.

## See also

[spinError](#)

## Parameters

|                |                                                              |
|----------------|--------------------------------------------------------------|
| <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

**13.12.1.89 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

**13.12.1.90 spinImageGetTLPayloadType()**

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

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

## See also

[spinError](#)

`spinPayloadTypeIn folds`

## 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

**13.12.1.91 spinImageGetTLPixelFormat()**

```
SPINNAKERC_API spinImageGetTLPixelFormat (
 spinImage hImage,
 uint64_t * pPixelFormat)
```

Retrieves the transport layer pixel format of an image (as an unsigned integer)

## See also

[spinError](#)



## Parameters

|                     |                                                                       |
|---------------------|-----------------------------------------------------------------------|
| <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

**13.12.1.92 spinImageGetTLPixelFormatNamespace()**

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

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

## See also

[spinError](#)

`spinPixelFormatNamespaceID`

## Parameters

|                              |                                                                                    |
|------------------------------|------------------------------------------------------------------------------------|
| <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

**13.12.1.93 spinImageGetValidPayloadSize()**

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

Retrieves the valid payload size of an image.

## See also

[spinError](#)

## Parameters

|               |                                                                                 |
|---------------|---------------------------------------------------------------------------------|
| <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

**13.12.1.94 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

**13.12.1.95 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

**13.12.1.96 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

**13.12.1.97 spinImageListAppend()**

```
SPINNAKERC_API spinImageListAppend (
 spinImageList hImageListBase,
 spinImageList hImageListToAppend)
```

Appends all the images from one image list to another.

## See also

[spinError](#)

## Parameters

|                           |                                     |
|---------------------------|-------------------------------------|
| <i>hImageListBase</i>     | The image list to receive the other |
| <i>hImageListToAppend</i> | The image list to add to the other  |

## Returns

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

**13.12.1.98 spinImageListClear()**

```
SPINNAKERC_API spinImageListClear (
 spinImageList hImageList)
```

Clears a image list.

## See also

[spinError](#)

## Parameters

|                   |                         |
|-------------------|-------------------------|
| <i>hImageList</i> | The image list to clear |
|-------------------|-------------------------|

## Returns

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

**13.12.1.99 spinImageListCreateEmpty()**

```
SPINNAKERC_API spinImageListCreateEmpty (
 spinImageList * phImageList)
```

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

## See also

[spinError](#)

## Parameters

|                    |                                                                         |
|--------------------|-------------------------------------------------------------------------|
| <i>phImageList</i> | The image list handle pointer in which the empty image list is returned |
|--------------------|-------------------------------------------------------------------------|

**Returns**

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

**13.12.1.100 spinImageListDestroy()**

```
SPINNAKERC_API spinImageListDestroy (
 spinImageList hImageList)
```

Destroys a image list.

**See also**

[spinError](#)

**Parameters**

|                   |                           |
|-------------------|---------------------------|
| <i>hImageList</i> | The image list to destroy |
|-------------------|---------------------------|

**Returns**

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

**13.12.1.101 spinImageListEventHandlerCreate()**

```
SPINNAKERC_API spinImageListEventHandlerCreate (
 spinImageListEventHandler * phImageEventHandler,
 spinImageListEventFunction pFunction,
 void * pUserData)
```

Creates an image list event handler.

**See also**

[spinError](#)

**Parameters**

|                                |                                                                                                                                                                                    |
|--------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>phImageListEventHandler</i> | The image list event handler pointer in which the image list event context is created                                                                                              |
| <i>pFunction</i>               | The function to be called at image list event occurrences; signature to match: <code>void(&lt;em&gt;spinImageListEventFunction)(const spinListImage hImage, void pUserData)</code> |
| <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

**13.12.1.102 spinImageListEventHandlerDestroy()**

```
SPINNAKERC_API spinImageListEventHandlerDestroy (
 spinImageListEventHandler hImageListEventHandler)
```

Destroys an image list event handler.

**See also**

[spinError](#)

**Parameters**

|                               |                                         |
|-------------------------------|-----------------------------------------|
| <i>hImageListEventHandler</i> | The image list event handler to destroy |
|-------------------------------|-----------------------------------------|

**Returns**

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

**13.12.1.103 spinImageListGet()**

```
SPINNAKERC_API spinImageListGet (
 spinImageList hImageList,
 size_t index,
 spinImage * phImage)
```

Retrieves a image from a image list using an index.

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

**See also**

[spinError](#)

**Parameters**

|                   |                                                         |
|-------------------|---------------------------------------------------------|
| <i>hImageList</i> | The image list of the image to retrieve                 |
| <i>index</i>      | The index of the 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

**13.12.1.104 spinImageListGetByPixelFormat()**

```
SPINNAKERC_API spinImageListGetByPixelFormat (
 spinImageList hImageList,
 spinPixelFormatEnums pixelFormat,
 spinImage * phImage)
```

Retrieves a image from a image list given its pixel format.

This function will return a NULL `spinImage` pointer if no matching image pixel format is found.

**See also**

[spinError](#)

**Parameters**

|                    |                                                         |
|--------------------|---------------------------------------------------------|
| <i>hImageList</i>  | The image list of the image to retrieve                 |
| <i>pixelFormat</i> | The pixel format 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

**13.12.1.105 spinImageListGetSize()**

```
SPINNAKERC_API spinImageListGetSize (
 spinImageList hImageList,
 size_t * pSize)
```

Retrieves the number of images in an image list.

**See also**

[spinError](#)

**Parameters**

|                   |                                                                        |
|-------------------|------------------------------------------------------------------------|
| <i>hImageList</i> | The image list where the images to be counted are                      |
| <i>pSize</i>      | The unsigned integer pointer in which the number of images is returned |

**Returns**

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

**13.12.1.106 spinImageListLoad()**

```
SPINNAKERC_API spinImageListLoad (
 spinImageList * phImageList,
 const char * fileName)
```

Creates an image list object from file.

**See also**

[spinImageListSave\(\)](#)

[spinError](#)

**Parameters**

|                    |                                                                         |
|--------------------|-------------------------------------------------------------------------|
| <i>phImageList</i> | The image list handle pointer in which the empty image list is returned |
| <i>fileName</i>    | Name of the file to load an image object from.                          |

**Returns**

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

**13.12.1.107 spinImageListRelease()**

```
SPINNAKERC_API spinImageListRelease (
 spinImageList hImageList)
```

**13.12.1.108 spinImageListRemove()**

```
SPINNAKERC_API spinImageListRemove (
 spinImageList hImageList,
 size_t index)
```

Removes a image from a image list using its index.

**See also**

[spinError](#)



## Parameters

|                   |                                        |
|-------------------|----------------------------------------|
| <i>hImageList</i> | The image list of the camera to remove |
| <i>index</i>      | The index of the image to remove       |

## Returns

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

**13.12.1.109 spinImageListRemoveByPixelFormat()**

```
SPINNAKERC_API spinImageListRemoveByPixelFormat (
 spinImageList hImageList,
 spinPixelFormatEnums pixelFormat)
```

Removes a image from a image list using its pixel format.

## See also

[spinError](#)

## Parameters

|                    |                                         |
|--------------------|-----------------------------------------|
| <i>hImageList</i>  | The image list of the image to remove   |
| <i>pixelFormat</i> | The pixel format of the image to remove |

## Returns

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

**13.12.1.110 spinImageListSave()**

```
SPINNAKERC_API spinImageListSave (
 spinImageList hImageList,
 const char * fileName)
```

Saves an image list as an object to a file.

## See also

[spinImageListLoad\(\)](#)

[spinError](#)

## Parameters

|                   |                                                                                                               |
|-------------------|---------------------------------------------------------------------------------------------------------------|
| <i>hImageList</i> | The image list of the image to remove                                                                         |
| <i>fileName</i>   | Name of the file to save the current image list object to. It is recommended to use the file extension 'sil'. |

## Returns

spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

**13.12.1.111 spinImageProcessorApplyGamma()**

```
SPINNAKERC_API spinImageProcessorApplyGamma (
 spinImageProcessor hImageProcessor,
 spinImage hSrcImage,
 spinImage hDestImage,
 float gamma,
 bool8_t applyGammaInverse)
```

Applies gamma correction to the source image and stores the result in the destination image.

## Parameters

|                          |                                                                                                                                                   |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>hImageProcessor</i>   | The image processor context                                                                                                                       |
| <i>hSrcImage</i>         | The source image from which to apply gamma on.                                                                                                    |
| <i>hDestImage</i>        | The destination image in which the gamma applied image data will be stored.                                                                       |
| <i>gamma</i>             | Gamma value to apply. A value between 0.5 and 4 is acceptable. (Default assuming image-to-screen)                                                 |
| <i>applyGammaInverse</i> | Converts a gamma corrected source image back to the original image using the inverse of the gamma value (used for applying screen-to-image gamma) |

**13.12.1.112 spinImageProcessorConvert()**

```
SPINNAKERC_API spinImageProcessorConvert (
 spinImageProcessor hImageProcessor,
 spinImage hSrcImage,
 spinImage hDestImage,
 spinPixelFormatEnums destFormat)
```

Converts the source image buffer to the specified destination pixel format and stores the result in the destination image.

The destination image needs to be configured to have the correct buffer size before calling this function. See [spinImageReset\(\)](#) to setup the correct buffer size according to specified pixel format.

Note that compressed images are decompressed before any further color processing or conversion during this call. Decompression is multi-threaded and defaults to utilizing one less than the number of concurrent threads supported by the system. The default number of decompression threads can be set with [spinImageProcessorSetNumDecompressionThreads\(\)](#).

See also

[spinPixelFormatEnums](#)  
[spinImageReset](#)  
[spinImageProcessorSetNumDecompressionThreads](#)

Parameters

|                        |                                                                         |
|------------------------|-------------------------------------------------------------------------|
| <i>hImageProcessor</i> | The image processor context                                             |
| <i>srcImage</i>        | The source image from which to convert the image from.                  |
| <i>destImage</i>       | The destination image in which the converted image data will be stored. |
| <i>destFormat</i>      | Output format of the converted image.                                   |

### 13.12.1.113 spinImageProcessorConvertImageList()

```
SPINNAKERC_API spinImageProcessorConvertImageList (
 spinImageProcessor hImageProcessor,
 spinImageList hSrcImageList,
 spinImage hDestImage,
 spinPixelFormatEnums destFormat)
```

Converts the source list of image buffers to the specified output pixel format and returns the result in a new image.

The conversion could encompasses decompression, interleaving and conversion of image data depending on the source pixel format of images in the source image list. The destination image needs to be configured to have the correct buffer size before calling this function. See [spinImageReset\(\)](#) to setup the correct buffer size according to specified pixel format.

Note that compressed images are decompressed before any further color processing, interleaving or conversion is performed. Decompression is multi-threaded and defaults to utilizing one less than the number of concurrent threads supported by the system. The default number of decompression threads can be set with [SetNumDecompressionThreads\(\)](#).

Note not all the supported image pixel formats described in the class description are supported in this function.

List of supported image pixel formats for the source image list:

- PixelFormat\_R12
- PixelFormat\_GR12
- PixelFormat\_GB12
- PixelFormat\_B12
- PixelFormat\_R12\_Jpeg
- PixelFormat\_GR12\_Jpeg
- PixelFormat\_GB12\_Jpeg
- PixelFormat\_B12\_Jpeg

See also

[spinPixelFormatEnums](#)  
[spinImageReset](#)  
[spinImageProcessorSetNumDecompressionThreads](#)

## Parameters

|                        |                                                                         |
|------------------------|-------------------------------------------------------------------------|
| <i>hImageProcessor</i> | The image processor context                                             |
| <i>hSrcImageList</i>   | List of images from which to convert the images from.                   |
| <i>hDestImage</i>      | The destination image in which the converted image data will be stored. |
| <i>destFormat</i>      | Output format of the converted image.                                   |

**13.12.1.114 spinImageProcessorCreate()**

```
SPINNAKERC_API spinImageProcessorCreate (
 spinImageProcessor * phImageProcessor)
```

Creates an image processor.

## See also

[spinError](#)

## Parameters

|                         |                                                                                     |
|-------------------------|-------------------------------------------------------------------------------------|
| <i>phImageProcessor</i> | The image processor handle pointer in which the image processor context is returned |
|-------------------------|-------------------------------------------------------------------------------------|

## Returns

spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

**13.12.1.115 spinImageProcessorDestroy()**

```
SPINNAKERC_API spinImageProcessorDestroy (
 spinImageProcessor hImageProcessor)
```

Destroys a image list.

## See also

[spinError](#)

## Parameters

|                        |                                        |
|------------------------|----------------------------------------|
| <i>hImageProcessor</i> | The image processor context to destroy |
|------------------------|----------------------------------------|

## Returns

spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

**13.12.1.116 spinImageProcessorGetColorProcessing()**

```
SPINNAKERC_API spinImageProcessorGetColorProcessing (
 spinImageProcessor hImageProcessor,
 spinColorProcessingAlgorithm * pColorAlgorithm)
```

Gets the default color processing algorithm.

## Parameters

|                        |                                                                                            |
|------------------------|--------------------------------------------------------------------------------------------|
| <i>hImageProcessor</i> | The image processor context                                                                |
| <i>pColorAlgorithm</i> | The color processing algorithm pointer in which the color processing algorithm is returned |

## See also

[spinImageProcessorSetColorProcessing\(\)](#)

**13.12.1.117 spinImageProcessorGetNumDecompressionThreads()**

```
SPINNAKERC_API spinImageProcessorGetNumDecompressionThreads (
 spinImageProcessor hImageProcessor,
 unsigned int * pNumThreads)
```

Gets the number of threads used for image decompression during [spinImageProcessorConvert\(\)](#).

## Parameters

|                        |                                                                                                      |
|------------------------|------------------------------------------------------------------------------------------------------|
| <i>hImageProcessor</i> | The image processor context                                                                          |
| <i>pNumThreads</i>     | The unsigned integer pointer in which the number of parallel image decompression threads is returned |

## See also

[spinImageProcessorSetNumDecompressionThreads\(\)](#)

**13.12.1.118 spinImageProcessorSetColorProcessing()**

```
SPINNAKERC_API spinImageProcessorSetColorProcessing (
 spinImageProcessor hImageProcessor,
 spinColorProcessingAlgorithm colorAlgorithm)
```

Sets the color processing algorithm used at the time of the [spinImageProcessorConvert\(\)](#) call, therefore the most recent execution of this function will take precedence.

The DEFAULT algorithm is deprecated and should not be used in the ImageProcessor class.

#### Parameters

|                        |                                        |
|------------------------|----------------------------------------|
| <i>hImageProcessor</i> | The image processor context            |
| <i>colorAlgorithm</i>  | The color processing algorithm to set. |

#### See also

[spinImageProcessorGetColorProcessing\(\)](#)

### 13.12.1.119 spinImageProcessorSetNumDecompressionThreads()

```
SPINNAKERC_API spinImageProcessorSetNumDecompressionThreads (
 spinImageProcessor hImageProcessor,
 unsigned int numThreads)
```

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

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

#### Parameters

|                        |                                                           |
|------------------------|-----------------------------------------------------------|
| <i>hImageProcessor</i> | The image processor context                               |
| <i>numThreads</i>      | Number of parallel image decompression threads set to run |

#### See also

[spinImageProcessorConvert\(\)](#)

### 13.12.1.120 spinImageRelease()

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

Releases an image.

#### See also

[spinError](#)

## Parameters

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

## Returns

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

**13.12.1.121 spinImageReset()**

```
SPINNAKERC_API spinImageReset (
 spinImage hImage,
 size_t width,
 size_t height,
 size_t offsetX,
 size_t offsetY,
 spinPixelFormatEnums pixelFormat)
```

Resets an image with some set properties.

## See also

[spinError](#)

## Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <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

**13.12.1.122 spinImageResetEx()**

```
SPINNAKERC_API spinImageResetEx (
 spinImage hImage,
 size_t width,
 size_t height,
 size_t offsetX,
```

```

size_t offsetY,
spinPixelFormatEnums pixelFormat,
void * pData)

```

Resets an image with some set properties and image data.

See also

[spinError](#)

#### Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>hImage</i>      | The image to 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            |
| <i>pData</i>       | The image data to reset to                 |

#### Returns

spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

### 13.12.1.123 spinImageSave()

```

SPINNAKERC_API spinImageSave (
 spinImage hImage,
 const char * pFilename,
 spinImageFileFormat format)

```

Saves an image using a specified file format (using an enum, spinImageFileFormat)

See also

[spinError](#)

[spinImageFileFormat](#)

#### Parameters

|                  |                                                                                                                                                  |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| <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) @Param<br>format The file format to use to save the image |

#### Returns

spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error



### 13.12.1.124 spinImageSaveBmp()

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

Saves an image as a BMP image.

See also

[spinError](#)

#### Parameters

|                  |                                                                                        |
|------------------|----------------------------------------------------------------------------------------|
| <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

### 13.12.1.125 spinImageSaveFromExt()

```
SPINNAKERC_API spinImageSaveFromExt (
 spinImage hImage,
 const char * pFilename)
```

Saves an image using a specified file format (using the extension of the filename)

See also

[spinError](#)

#### Parameters

|                  |                                                                                        |
|------------------|----------------------------------------------------------------------------------------|
| <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

### 13.12.1.126 spinImageSaveJpeg()

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

Saves an image as a JPEG image.

See also

[spinError](#)

#### Parameters

|                  |                                                                                                  |
|------------------|--------------------------------------------------------------------------------------------------|
| <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

### 13.12.1.127 spinImageSaveJpg2()

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

Saves an image as a JPEG 2000 image.

See also

[spinError](#)

#### Parameters

|                  |                                                                                        |
|------------------|----------------------------------------------------------------------------------------|
| <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

**13.12.1.128 spinImageSavePgm()**

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

Saves an image as an PGM image.

See also

[spinError](#)

**Parameters**

|                  |                                                                                        |
|------------------|----------------------------------------------------------------------------------------|
| <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

**13.12.1.129 spinImageSavePng()**

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

Saves an image as a PNG image.

See also

[spinError](#)

**Parameters**

|                  |                                                                                                          |
|------------------|----------------------------------------------------------------------------------------------------------|
| <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

### 13.12.1.130 spinImageSavePpm()

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

Saves an image as a PPM image.

See also

[spinError](#)

#### Parameters

|                  |                                                                                        |
|------------------|----------------------------------------------------------------------------------------|
| <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

### 13.12.1.131 spinImageSaveTiff()

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

Saves an image as a TIFF image.

See also

[spinError](#)

#### Parameters

|                  |                                                                                        |
|------------------|----------------------------------------------------------------------------------------|
| <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

### 13.12.1.132 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

### 13.12.1.133 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

### 13.12.1.134 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

**13.12.1.135 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

**13.12.1.136 spinImageStatisticsEnableGreyOnly()**

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

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

## See also

[spinError](#)

## Parameters

|                    |                                                  |
|--------------------|--------------------------------------------------|
| <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

**13.12.1.137 spinImageStatisticsEnableHslOnly()**

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

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

**See also**

[`spinError`](#)

**Parameters**

|                    |                                                 |
|--------------------|-------------------------------------------------|
| <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

**13.12.1.138 spinImageStatisticsEnableRgbOnly()**

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

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

**See also**

[`spinError`](#)

**Parameters**

|                    |                                                 |
|--------------------|-------------------------------------------------|
| <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

**13.12.1.139 spinImageStatisticsGetAll()**

```
SPINNAKERC_API spinImageStatisticsGetAll (
 spinImageStatistics hStatistics,
 spinStatisticsChannel channel,
 unsigned int * pRangeMin,
 unsigned int * pRangeMax,
 unsigned int * pPixelValueMin,
 unsigned int * pPixelValueMax,
 unsigned int * pNumPixelValues,
 float * pPixelValueMean,
 int ** ppHistogram)
```

Retrieves all available information of an image statistics channel.

See also

[spinError](#)

**Parameters**

|                        |                                                                                        |
|------------------------|----------------------------------------------------------------------------------------|
| <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

**13.12.1.140 spinImageStatisticsGetChannelStatus()**

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

Checks whether an image statistics context is enabled.

See also

[spinError](#)



## Parameters

|                    |                                                                     |
|--------------------|---------------------------------------------------------------------|
| <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

**13.12.1.141 spinImageStatisticsGetHistogram()**

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

Retrieves a histogram of an image statistics channel.

## See also

[spinError](#)

## Parameters

|                    |                                                                            |
|--------------------|----------------------------------------------------------------------------|
| <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

**13.12.1.142 spinImageStatisticsGetMean()**

```
SPINNAKERC_API spinImageStatisticsGetMean (
 spinImageStatistics hStatistics,
 spinStatisticsChannel channel,
 float * pMean)
```

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

## See also

[spinError](#)

## Parameters

|                    |                                                             |
|--------------------|-------------------------------------------------------------|
| <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

**13.12.1.143 spinImageStatisticsGetNumPixelValues()**

```
SPINNAKERC_API spinImageStatisticsGetNumPixelValues (
 spinImageStatistics hStatistics,
 spinStatisticsChannel channel,
 unsigned int * pNumValues)
```

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

## See also

[spinError](#)

## Parameters

|                    |                                                                              |
|--------------------|------------------------------------------------------------------------------|
| <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

**13.12.1.144 spinImageStatisticsGetPixelValueRange()**

```
SPINNAKERC_API spinImageStatisticsGetPixelValueRange (
 spinImageStatistics hStatistics,
 spinStatisticsChannel channel,
 unsigned int * pMin,
 unsigned int * pMax)
```

Retrieves the pixel value range of an image statistics channel.

## See also

[spinError](#)

## Parameters

|                    |                                                                                              |
|--------------------|----------------------------------------------------------------------------------------------|
| <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

**13.12.1.145 spinImageStatisticsGetRange()**

```
SPINNAKERC_API spinImageStatisticsGetRange (
 spinImageStatistics hStatistics,
 spinStatisticsChannel channel,
 unsigned int * pMin,
 unsigned int * pMax)
```

Retrieves the range of an image statistics channel.

## See also

[spinError](#)

## Parameters

|                    |                                                                                  |
|--------------------|----------------------------------------------------------------------------------|
| <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

**13.12.1.146 spinImageStatisticsSetChannelStatus()**

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

Sets the status of an image statistics channel.

## See also

[spinError](#)

## Parameters

|                    |                                                        |
|--------------------|--------------------------------------------------------|
| <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

**13.12.1.147 spinInterfaceEventHandlerCreate()**

```
SPINNAKERC_API spinInterfaceEventHandlerCreate (
 spinInterfaceEventHandler * phInterfaceEventHandler,
 spinArrivalEventFunction pArrivalFunction,
 spinRemovalEventFunction pRemovalFunction,
 void * pUserData)
```

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

## See also

[spinError](#)

## Parameters

|                                |                                                                                                                                                                                   |
|--------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <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:<br><code>void(&lt;em&gt;spinArrivalEventFunction)(void pUserData)</code>                              |
| <i>hRemovalFunction</i>        | The function to be called at removal event occurrences; signature to match:<br><code>void(&lt;em&gt;spinRemovalEventFunction)(uint64_t deviceSerialNumber, void pUserData)</code> |
| <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

**13.12.1.148 spinInterfaceEventHandlerDestroy()**

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

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

## See also

[spinError](#)

## Parameters

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

## Returns

*spinError* The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

**13.12.1.149 spinInterfaceGetCameras()**

```
SPINNAKERC_API spinInterfaceGetCameras (
 spinInterface hInterface,
 spinCameraList hCameraList)
```

Retrieves a camera list from an interface; camera lists must be created and destroy.

## See also

[spinCameraListCreateEmpty\(\)](#)  
[spinCameraListDestroy\(\)](#)  
[spinError](#)

## Parameters

|                    |                                                         |
|--------------------|---------------------------------------------------------|
| <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

**13.12.1.150 spinInterfaceGetCamerasEx()**

```
SPINNAKERC_API spinInterfaceGetCamerasEx (
 spinInterface hInterface,
 bool8_t bUpdateCameras,
 spinCameraList hCameraList)
```

Retrieves a camera list from an interface; manually set whether to update the cameras; camera lists must be created and destroyed.

## See also

[spinCameraListCreateEmpty\(\)](#)  
[spinCameraListDestroy\(\)](#)  
[spinError](#)

## Parameters

|                       |                                                         |
|-----------------------|---------------------------------------------------------|
| <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

**13.12.1.151 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

**13.12.1.152 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

**13.12.1.153 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

**13.12.1.154 spinInterfaceListCreateEmpty()**

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

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

## See also

[spinError](#)

## Parameters

|                        |                                                                                 |
|------------------------|---------------------------------------------------------------------------------|
| <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

**13.12.1.155 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

**13.12.1.156 spinInterfaceListGet()**

```
SPINNAKERC_API spinInterfaceListGet (
 spinInterfaceList hInterfaceList,
 size_t index,
 spinInterface * phInterface)
```

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

**See also**

[spinError](#)

**Parameters**

|                       |                                                                 |
|-----------------------|-----------------------------------------------------------------|
| <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

**13.12.1.157 spinInterfaceListGetSize()**

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

Retrieves the number of interfaces in an interface list.

**See also**

[spinError](#)

**Parameters**

|                       |                                                                            |
|-----------------------|----------------------------------------------------------------------------|
| <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](#)

**13.12.1.158 spinInterfaceRegisterDeviceArrivalEventHandler()**

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

Registers a device arrival event handler on an interface (event handlers registered in this way must be unregistered)

**See also**

[spinError](#)

**Parameters**

|                                   |                                                                     |
|-----------------------------------|---------------------------------------------------------------------|
| <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

**13.12.1.159 spinInterfaceRegisterDeviceRemovalEventHandler()**

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

Registers a device removal event handler on an interface (event handlers registered in this way must be unregistered)

**See also**

[spinError](#)

**Parameters**

|                                   |                                                                     |
|-----------------------------------|---------------------------------------------------------------------|
| <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

**13.12.1.160 spinInterfaceRegisterInterfaceEventHandler()**

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

Registers an interface event handler (both device arrival and device removal) on an interface.

**See also**

[spinError](#)

**Parameters**

|                               |                                                                |
|-------------------------------|----------------------------------------------------------------|
| <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

**13.12.1.161 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

**13.12.1.162 spinInterfaceSendActionCommand()**

```
SPINNAKERC_API spinInterfaceSendActionCommand (
 spinInterface hInterface,
 size_t iDeviceKey,
 size_t iGroupKey,
 size_t iGroupMask,
 size_t iActionTime,
 size_t * piResultSize,
 actionCommandResult results[])
```

Broadcast an Action Command to all devices on interface.

**See also**

[spinError](#)

**Parameters**

|                    |                                                                              |
|--------------------|------------------------------------------------------------------------------|
| <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. |

## Parameters

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <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

**13.12.1.163 spinInterfaceUnregisterDeviceArrivalEventHandler()**

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

Unregisters a device arrival event handler from an interface.

## See also

[spinError](#)

## Parameters

|                                   |                                                                         |
|-----------------------------------|-------------------------------------------------------------------------|
| <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

**13.12.1.164 spinInterfaceUnregisterDeviceRemovalEventHandler()**

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

Unregisters a device removal event handler from an interface.

## See also

[spinError](#)

## Parameters

|                                   |                                                                         |
|-----------------------------------|-------------------------------------------------------------------------|
| <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

**13.12.1.165 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

**13.12.1.166 spinInterfaceUpdateCameras()**

```
SPINNAKERC_API spinInterfaceUpdateCameras (
 spinInterface hInterface,
 bool8_t * pbChanged)
```

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

## See also

[spinError](#)

## Parameters

|                   |                                                                       |
|-------------------|-----------------------------------------------------------------------|
| <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

**13.12.1.167 spinLogDataGetCategoryName()**

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

Retrieves the category name of a log event.

## See also

[spinError](#)

## Parameters

|                      |                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------|
| <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

**13.12.1.168 spinLogDataGetLogMessage()**

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

Retrieves the log message of a log event.

## See also

[spinError](#)

## Parameters

|                      |                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------|
| <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

**13.12.1.169 spinLogDataGetNDC()**

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

Retrieves the NDC of a log event.

## See also

[spinError](#)

## Parameters

|                      |                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------|
| <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

**13.12.1.170 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

**13.12.1.171 spinLogDataGetPriorityName()**

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

Retrieves the priority name of a log event.

## See also

[spinError](#)

## Parameters

|                      |                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------|
| <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

**13.12.1.172 spinLogDataGetThreadName()**

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

Retrieves the thread name of a log event.

## See also

[spinError](#)



## Parameters

|                      |                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------|
| <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

**13.12.1.173 spinLogDataGetTimestamp()**

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

Retrieves the timestamp of a log event.

## See also

[spinError](#)

## Parameters

|                      |                                                                                                                     |
|----------------------|---------------------------------------------------------------------------------------------------------------------|
| <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

**13.12.1.174 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:<br>void(<em>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

**13.12.1.175 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

**13.12.1.176 spinSystemGetCameras()**

```
SPINNAKERC_API spinSystemGetCameras (
 spinSystem hSystem,
 spinCameraList hCameraList)
```

Retrieves a list of detected (and enumerable) cameras on the system; camera lists must be created and destroyed.

## See also

[spinCameraListCreateEmpty\(\)](#)

[spinCameraListDestroy\(\)](#)

[spinError](#)

## Parameters

|                    |                                                      |
|--------------------|------------------------------------------------------|
| <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

**13.12.1.177 spinSystemGetCamerasEx()**

```
SPINNAKERC_API spinSystemGetCamerasEx (
 spinSystem hSystem,
 bool8_t bUpdateInterfaces,
 bool8_t bUpdateCameras,
 spinCameraList hCameraList)
```

Retrieves a list of detected (and enumerable) cameras on the system; manually set whether to update the current interface and camera lists; camera lists must be created and destroyed.

## See also

[spinCameraListCreateEmpty\(\)](#)  
[spinCameraListDestroy\(\)](#)  
[spinError](#)

## Parameters

|                          |                                                      |
|--------------------------|------------------------------------------------------|
| <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

**13.12.1.178 spinSystemGetInstance()**

```
SPINNAKERC_API spinSystemGetInstance (
 spinSystem * phSystem)
```

Retrieves an instance of the system object; the system is a singleton, so there will only ever be one instance; system instance must be destroyed by calling `spinSystemReleaseInstance`.

See also

[spinSystemReleaseInstance](#)  
[spinError](#)

Parameters

|                 |                                                                    |
|-----------------|--------------------------------------------------------------------|
| <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

### 13.12.1.179 **spinSystemGetInterfaces()**

```
SPINNAKERC_API spinSystemGetInterfaces (
 spinSystem hSystem,
 spinInterfaceList hInterfaceList)
```

Retrieves a list of detected (and enumerable) interfaces on the system; interface lists must be created and destroyed.

See also

[spinInterfaceListCreateEmpty\(\)](#)  
[spinInterfaceListDestroy\(\)](#)  
[spinError](#)

Parameters

|                       |                                                            |
|-----------------------|------------------------------------------------------------|
| <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

### 13.12.1.180 **spinSystemGetLibraryVersion()**

```
SPINNAKERC_API spinSystemGetLibraryVersion (
 spinSystem hSystem,
 spinLibraryVersion * hLibraryVersion)
```

Get current library version of Spinnaker.

Returns

A struct containing the current version of Spinnaker(major, minor, type, build).

**13.12.1.181 spinSystemGetLoggingLevel()**

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

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

See also

[spinError](#)

**Parameters**

|                 |                                                                               |
|-----------------|-------------------------------------------------------------------------------|
| <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

**13.12.1.182 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

**13.12.1.183 spinSystemIsInUse()**

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

Checks whether a system is currently in use.

See also

[spinError](#)

**Parameters**

|                  |                                                                      |
|------------------|----------------------------------------------------------------------|
| <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

**13.12.1.184 spinSystemRegisterDeviceArrivalEventHandler()**

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

Registers a device arrival event handler to every interface on the system (event handlers registered this way must be unregistered)

See also

[spinError](#)

**Parameters**

|                                   |                                                                     |
|-----------------------------------|---------------------------------------------------------------------|
| <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

**13.12.1.185 spinSystemRegisterDeviceRemovalEventHandler()**

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

Registers a device removal event handler to the system to every interface on the system (event handlers registered this way must be unregistered)

See also

[spinError](#)

**Parameters**

|                                   |                                                                     |
|-----------------------------------|---------------------------------------------------------------------|
| <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

**13.12.1.186 spinSystemRegisterInterfaceEventHandler()**

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

Registers an interface event handler (device arrival and device removal) to every interface on the system (interface events registered this way must be unregistered) If new interfaces are detected by the system after [spinSystemRegisterInterfaceEventHandler\(\)](#) is called, those interfaces will be automatically registered with this event.

See also

[spinError](#)

[spinInterfaceEventHandler](#)

**Parameters**

|                               |                                                                                           |
|-------------------------------|-------------------------------------------------------------------------------------------|
| <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

**13.12.1.187 spinSystemRegisterLogEventHandler()**

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

Registers a logging event handler to the system (event handlers registered in this way must be unregistered)

See also

[spinError](#)

**Parameters**

|                         |                                                              |
|-------------------------|--------------------------------------------------------------|
| <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

**13.12.1.188 spinSystemReleaseInstance()**

```
SPINNAKERC_API spinSystemReleaseInstance (
 spinSystem hSystem)
```

Releases the system; make sure handle is cleaned up properly by setting it to NULL after system is released; the handle can only be used again after calling spinSystemGetInstance.

See also

[spinSystemGetInstance](#)

[spinError](#)

**Parameters**

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

**Returns**

spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error

**13.12.1.189 spinSystemSendActionCommand()**

```
SPINNAKERC_API spinSystemSendActionCommand (
 spinSystem hSystem,
```



```

size_t iDeviceKey,
size_t iGroupKey,
size_t iGroupMask,
size_t iActionTime,
size_t * piResultSize,
actionCommandResult results[])

```

Broadcast an Action Command to all devices on system.

See also

[spinError](#)

#### Parameters

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <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

### 13.12.1.190 spinSystemSetLoggingLevel()

```

SPINNAKERC_API spinSystemSetLoggingLevel (
 spinSystem hSystem,
 spinnakerLogLevel logLevel)

```

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

See also

[spinError](#)

#### Parameters

|                 |                                               |
|-----------------|-----------------------------------------------|
| <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

**13.12.1.191 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

**13.12.1.192 spinSystemUnregisterDeviceArrivalEventHandler()**

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

Unregisters a device arrival event handler from the system.

**See also**

[spinError](#)

[spinDeviceArrivalEventHandler](#)

**Parameters**

|                                   |                                                                         |
|-----------------------------------|-------------------------------------------------------------------------|
| <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

**13.12.1.193 spinSystemUnregisterDeviceRemovalEventHandler()**

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

Unregisters a device removal event handler from the system.

See also

[spinError](#)

[spinDeviceRemovalEventHandler](#)

**Parameters**

|                                   |                                                                         |
|-----------------------------------|-------------------------------------------------------------------------|
| <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

**13.12.1.194 spinSystemUnregisterInterfaceEventHandler()**

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

Unregisters an interface event handler from the system.

See also

[spinError](#)

[spinInterfaceEventHandler](#)

**Parameters**

|                               |                                                                                               |
|-------------------------------|-----------------------------------------------------------------------------------------------|
| <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

**13.12.1.195 spinSystemUnregisterLogEventHandler()**

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

Unregisters a selected logging event handler from the system.

**See also**

[spinError](#)

**Parameters**

|                         |                                                                  |
|-------------------------|------------------------------------------------------------------|
| <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

**13.12.1.196 spinSystemUpdateCameras()**

```
SPINNAKERC_API spinSystemUpdateCameras (
 spinSystem hSystem,
 bool8_t * pbChanged)
```

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

**See also**

[spinError](#)

**Parameters**

|                  |                                                                                               |
|------------------|-----------------------------------------------------------------------------------------------|
| <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

13.12.1.197 `spinSystemUpdateCamerasEx()`

```
SPINNAKERC_API spinSystemUpdateCamerasEx (
 spinSystem hSystem,
 bool8_t bUpdateInterfaces,
 bool8_t * pbChanged)
```

Updates the list of cameras on the system, informing whether there has been any changes; manually set whether to update the current interface lists.

## See also

[`spinError`](#)

## Parameters

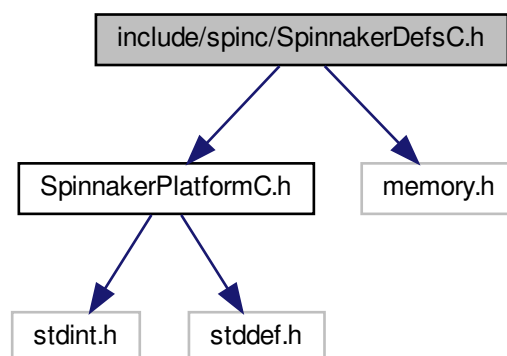
|                          |                                                                                            |
|--------------------------|--------------------------------------------------------------------------------------------|
| <i>hSystem</i>           | The system, on which the list 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

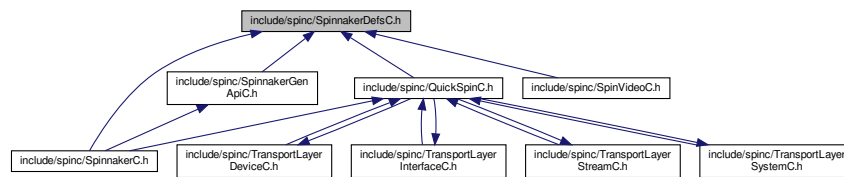
`spinError` The error code; returns `SPINNAKER_ERR_SUCCESS` (or 0) for no error

## 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 \* [spinImageList](#)
  - Handle for image list functionality.*
- typedef void \* [spinImageProcessor](#)
  - Handle for image processor functionality.*
- typedef void \* [spinImageStatistics](#)
  - Handle for image statistics functionality.*
- typedef void \* [spinDeviceEventHandler](#)
  - Handle for device event handler functionality.*
- typedef void \* [spinImageEventHandler](#)
  - Handle for image event handler functionality.*
- typedef void \* [spinImageListEventHandler](#)
  - Handle for image list event handler functionality.*
- typedef void \* [spinDeviceArrivalEventHandler](#)
  - Handle for arrival event handler functionality.*
- typedef void \* [spinDeviceRemovalEventHandler](#)
  - Handle for removal event handler functionality.*
- typedef void \* [spinInterfaceEventHandler](#)
  - Handle for interface event handler functionality.*
- typedef void \* [spinLogEventHandler](#)
  - Handle for logging event handler functionality.*
- typedef void \* [spinLogEventData](#)
  - Handle for logging event data functionality.*
- typedef void \* [spinDeviceEventData](#)
  - Handle for device event data functionality.*
- typedef void \* [spinVideo](#)
  - Handle for video recording functionality.*
- typedef void(\* [spinDeviceEventFunction](#)) (const [spinDeviceEventData](#) hEventData, const char \*pEventName, void \*pUserData)
- Function signatures are used to create and trigger callbacks and events.*
- typedef void(\* [spinImageEventFunction](#)) (const [spinImage](#) hImage, void \*pUserData)
- typedef void(\* [spinImageListEventFunction](#)) (const [spinImageList](#) hImage, void \*pUserData)
- typedef void(\* [spinArrivalEventFunction](#)) (const [spinCamera](#) hCamera, void \*pUserData)
- typedef void(\* [spinRemovalEventFunction](#)) (const [spinCamera](#) hCamera, void \*pUserData)
- typedef void(\* [spinLogEventFunction](#)) (const [spinLogEventData](#) hEventData, void \*pUserData)

## Enumerations

- enum [spinError](#) {
  - [SPINNAKER\\_ERR\\_SUCCESS](#) = 0 ,
  - [SPINNAKER\\_ERR\\_ERROR](#) = -1001 ,
  - [SPINNAKER\\_ERR\\_NOT\\_INITIALIZED](#) = -1002 ,
  - [SPINNAKER\\_ERR\\_NOT\\_IMPLEMENTED](#) = -1003 ,
  - [SPINNAKER\\_ERR\\_RESOURCE\\_IN\\_USE](#) = -1004 ,
  - [SPINNAKER\\_ERR\\_ACCESS\\_DENIED](#) = -1005 ,
  - [SPINNAKER\\_ERR\\_INVALID\\_HANDLE](#) = -1006 ,
  - [SPINNAKER\\_ERR\\_INVALID\\_ID](#) = -1007 ,
  - [SPINNAKER\\_ERR\\_NO\\_DATA](#) = -1008 ,
  - [SPINNAKER\\_ERR\\_INVALID\\_PARAMETER](#) = -1009 ,
  - [SPINNAKER\\_ERR\\_IO](#) = -1010 ,

```

SPINNAKER_ERR_TIMEOUT = -1011 ,
SPINNAKER_ERR_ABORT = -1012 ,
SPINNAKER_ERR_INVALID_BUFFER = -1013 ,
SPINNAKER_ERR_NOT_AVAILABLE = -1014 ,
SPINNAKER_ERR_INVALID_ADDRESS = -1015 ,
SPINNAKER_ERR_BUFFER_TOO_SMALL = -1016 ,
SPINNAKER_ERR_INVALID_INDEX = -1017 ,
SPINNAKER_ERR_PARSING_CHUNK_DATA = -1018 ,
SPINNAKER_ERR_INVALID_VALUE = -1019 ,
SPINNAKER_ERR_RESOURCE_EXHAUSTED = -1020 ,
SPINNAKER_ERR_OUT_OF_MEMORY = -1021 ,
SPINNAKER_ERR_BUSY = -1022 ,
SPINNAKER_ERR_GENICAM_INVALID_ARGUMENT = -2001 ,
SPINNAKER_ERR_GENICAM_OUT_OF_RANGE = -2002 ,
SPINNAKER_ERR_GENICAM_PROPERTY = -2003 ,
SPINNAKER_ERR_GENICAM_RUN_TIME = -2004 ,
SPINNAKER_ERR_GENICAM_LOGICAL = -2005 ,
SPINNAKER_ERR_GENICAM_ACCESS = -2006 ,
SPINNAKER_ERR_GENICAM_TIMEOUT = -2007 ,
SPINNAKER_ERR_GENICAM_DYNAMIC_CAST = -2008 ,
SPINNAKER_ERR_GENICAM_GENERIC = -2009 ,
SPINNAKER_ERR_GENICAM_BAD_ALLOCATION = -2010 ,
SPINNAKER_ERR_IM_CONVERT = -3001 ,
SPINNAKER_ERR_IM_COPY = -3002 ,
SPINNAKER_ERR_IM_MALLOC = -3003 ,
SPINNAKER_ERR_IM_NOT_SUPPORTED = -3004 ,
SPINNAKER_ERR_IM_HISTOGRAM_RANGE = -3005 ,
SPINNAKER_ERR_IM_HISTOGRAM_MEAN = -3006 ,
SPINNAKER_ERR_IM_MIN_MAX = -3007 ,
SPINNAKER_ERR_IM_COLOR_CONVERSION = -3008 ,
SPINNAKER_ERR_CUSTOM_ID = -10000 }

```

*The error codes used in Spinnaker C.*

- enum `spinColorProcessingAlgorithm` {
 

```

SPINNAKER_COLOR_PROCESSING_ALGORITHM_NONE ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_NEAREST_NEIGHBOR ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_NEAREST_NEIGHBOR_AVG ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_BILINEAR ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_EDGE_SENSING ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_HQ_LINEAR ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_IPP ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_DIRECTIONAL_FILTER ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_RIGOROUS ,
SPINNAKER_COLOR_PROCESSING_ALGORITHM_WEIGHTED_DIRECTIONAL_FILTER }

```

*Color processing algorithms.*

- enum `spinStatisticsChannel` {
 

```

SPINNAKER_STATISTICS_CHANNEL_GREY ,
SPINNAKER_STATISTICS_CHANNEL_RED ,
SPINNAKER_STATISTICS_CHANNEL_GREEN ,
SPINNAKER_STATISTICS_CHANNEL_BLUE ,
SPINNAKER_STATISTICS_CHANNEL_HUE ,
SPINNAKER_STATISTICS_CHANNEL_SATURATION ,
SPINNAKER_STATISTICS_CHANNEL_LIGHTNESS ,
SPINNAKER_STATISTICS_CHANNEL_NUM_CHANNELS }

```

*Channels that allow statistics to be calculated.*

- enum `spinImageFileFormat` {
 

```

SPINNAKER_IMAGE_FILE_FORMAT_FROM_FILE_EXT = -1 ,
SPINNAKER_IMAGE_FILE_FORMAT_PGM ,
SPINNAKER_IMAGE_FILE_FORMAT_PPM ,

```



```
SPINNAKER_IMAGE_FILE_FORMAT_BMP ,
SPINNAKER_IMAGE_FILE_FORMAT_JPEG ,
SPINNAKER_IMAGE_FILE_FORMAT_JPEG2000 ,
SPINNAKER_IMAGE_FILE_FORMAT_TIFF ,
SPINNAKER_IMAGE_FILE_FORMAT_PNG ,
SPINNAKER_IMAGE_FILE_FORMAT_RAW ,
SPINNAKER_IMAGE_FILE_FORMAT_FORCE_32BITS = 0x7FFFFFFF }
```

*File formats to be used for saving images to disk.*

- enum `spinTLPixelFormatNamespace` {
 

```
SPINNAKER_TLPixelFormatNamespace_UNKNOWN = 0 ,
SPINNAKER_TLPixelFormatNamespace_GEV = 1 ,
SPINNAKER_TLPixelFormatNamespace_IIDC = 2 ,
SPINNAKER_TLPixelFormatNamespace_PFNC_16BIT = 3 ,
SPINNAKER_TLPixelFormatNamespace_PFNC_32BIT = 4 ,
SPINNAKER_PixelFormatNamespace_CUSTOM_ID = 1000 }
```

*This enum represents the namespace in which the TL specific pixel format resides.*

- enum `spinImageStatus` {
 

```
SPINNAKER_IMAGE_STATUS_UNKNOWN_ERROR = -1 ,
SPINNAKER_IMAGE_STATUS_NO_ERROR = 0 ,
SPINNAKER_IMAGE_STATUS_CRC_CHECK_FAILED = 1 ,
SPINNAKER_IMAGE_STATUS_DATA_OVERFLOW = 2 ,
SPINNAKER_IMAGE_STATUS_MISSING_PACKETS ,
SPINNAKER_IMAGE_STATUS_LEADER_BUFFER_SIZE_INCONSISTENT ,
SPINNAKER_IMAGE_STATUS_TRAILER_BUFFER_SIZE_INCONSISTENT ,
SPINNAKER_IMAGE_STATUS_PACKETID_INCONSISTENT ,
SPINNAKER_IMAGE_STATUS_MISSING_LEADER = 7 ,
SPINNAKER_IMAGE_STATUS_MISSING_TRAILER = 8 ,
SPINNAKER_IMAGE_STATUS_DATA_INCOMPLETE = 9 ,
SPINNAKER_IMAGE_STATUS_INFO_INCONSISTENT = 10 ,
SPINNAKER_IMAGE_STATUS_CHUNK_DATA_INVALID = 11 ,
SPINNAKER_IMAGE_STATUS_NO_SYSTEM_RESOURCES = 12 }
```

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

- enum `spinnakerLogLevel` {
 

```
SPINNAKER_LOG_LEVEL_OFF = -1 ,
SPINNAKER_LOG_LEVEL_FATAL = 0 ,
SPINNAKER_LOG_LEVEL_ALERT = 100 ,
SPINNAKER_LOG_LEVEL_CRIT = 200 ,
SPINNAKER_LOG_LEVEL_ERROR = 300 ,
SPINNAKER_LOG_LEVEL_WARN = 400 ,
SPINNAKER_LOG_LEVEL_NOTICE = 500 ,
SPINNAKER_LOG_LEVEL_INFO = 600 ,
SPINNAKER_LOG_LEVEL_DEBUG = 700 ,
SPINNAKER_LOG_LEVEL_NOTSET = 800 }
```

*log levels*

- enum `spinTLPayloadType` {
 

```
SPINNAKER_TLPayloadType_UNKNOWN = 0 ,
SPINNAKER_TLPayloadType_IMAGE = 1 ,
SPINNAKER_TLPayloadType_RAW_DATA = 2 ,
SPINNAKER_TLPayloadType_FILE = 3 ,
SPINNAKER_TLPayloadType_CHUNK_DATA = 4 ,
SPINNAKER_TLPayloadType_JPEG = 5 ,
SPINNAKER_TLPayloadType_JPEG2000 = 6 ,
SPINNAKER_TLPayloadType_H264 = 7 ,
SPINNAKER_TLPayloadType_CHUNK_ONLY = 8 ,
SPINNAKER_TLPayloadType_DEVICE_SPECIFIC = 9 ,
SPINNAKER_TLPayloadType_MULTI_PART = 10 ,
SPINNAKER_TLPayloadType_CUSTOM_ID = 1000 ,
```

- ```

SPINNAKER_TLPAYLOAD_TYPE_LOSSLESS_COMPRESSED = SPINNAKER_TLPAYLOAD_TYPE_↵
CUSTOM_ID + 1 ,
SPINNAKER_TLPAYLOAD_TYPE_LOSSY_COMPRESSED    = SPINNAKER_TLPAYLOAD_TYPE_↵
CUSTOM_ID + 2 ,
SPINNAKER_TLPAYLOAD_TYPE_JPEG_LOSSLESS_COMPRESSED = SPINNAKER_TLPAYLOAD_↵
TYPE_CUSTOM_ID + 3 }

```
- enum `spinTIFFCompressionMethod` {

```

SPINNAKER_TIFF_COMPRESS_METHOD_NONE = 1 ,
SPINNAKER_TIFF_COMPRESS_METHOD_PACKBITS ,
SPINNAKER_TIFF_COMPRESS_METHOD_DEFLATE ,
SPINNAKER_TIFF_COMPRESS_METHOD_ADOBE_DEFLATE ,
SPINNAKER_TIFF_COMPRESS_METHOD_CCITTFAX3 ,
SPINNAKER_TIFF_COMPRESS_METHOD_CCITTFAX4 ,
SPINNAKER_TIFF_COMPRESS_METHOD_LZW ,
SPINNAKER_TIFF_COMPRESS_METHOD_JPG }

```

Compression method to use for encoding TIFF images.
 - enum `spinActionCommandStatus` {

```

SPINNAKER_ACTION_COMMAND_STATUS_OK = 0 ,
SPINNAKER_ACTION_COMMAND_STATUS_NO_REF_TIME = 0x8013 ,
SPINNAKER_ACTION_COMMAND_STATUS_OVERFLOW = 0x8015 ,
SPINNAKER_ACTION_COMMAND_STATUS_ACTION_LATE = 0x8016 ,
SPINNAKER_ACTION_COMMAND_STATUS_ERROR = 0x8FFF }

```

Possible Status Codes Returned from Action Command.

Variables

- static const `bool8_t False` = 0
- static const `bool8_t True` = 1

13.13.1 Typedef Documentation

13.13.1.1 bool8_t

```
typedef uint8_t bool8_t
```

13.13.1.2 spinArrivalEventFunction

```
typedef void(* spinArrivalEventFunction) (const spinCamera hCamera, void *pUserData)
```

13.13.1.3 spinCamera

```
typedef void* spinCamera
```

Handle for camera functionality.

Created by calling `spinCameraListGet()`, which requires a call to `spinCameraRelease()` to release.

13.13.1.4 spinCameraList

```
typedef void* spinCameraList
```

Handle for interface functionality.

Created by calling [spinSystemGetCameras\(\)](#) or [spinInterfaceGetCameras\(\)](#), which require a call to [spinCameraListClear\(\)](#) to clear, or [spinCameraListCreateEmpty\(\)](#), which requires a call to [spinCameraListDestroy\(\)](#) to destroy.

13.13.1.5 spinDeviceArrivalEventHandler

```
typedef void* spinDeviceArrivalEventHandler
```

Handle for arrival event handler functionality.

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

13.13.1.6 spinDeviceEventData

```
typedef void* spinDeviceEventData
```

Handle for device event data functionality.

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

13.13.1.7 spinDeviceEventFunction

```
typedef void(* spinDeviceEventFunction) (const spinDeviceEventData hEventData, const char *p↔  
EventName, void *pUserData)
```

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

13.13.1.8 spinDeviceEventHandler

```
typedef void* spinDeviceEventHandler
```

Handle for device event handler functionality.

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

13.13.1.9 spinDeviceRemovalEventHandler

```
typedef void* spinDeviceRemovalEventHandler
```

Handle for removal event handler functionality.

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

13.13.1.10 spinImage

```
typedef void* spinImage
```

Handle for image functionality.

Created by calling [spinCameraGetNextImage\(\)](#) or [spinCameraGetNextImageEx\(\)](#), which require a call to [spinImageRelease\(\)](#) to remove from buffer, or [spinImageCreateEmpty\(\)](#), [spinImageCreateEx\(\)](#), or [spinImageCreate\(\)](#), which require a call to [spinImageDestroy\(\)](#) to destroy.

13.13.1.11 spinImageEventFunction

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

13.13.1.12 spinImageEventHandler

```
typedef void* spinImageEventHandler
```

Handle for image event handler functionality.

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

13.13.1.13 spinImageList

```
typedef void* spinImageList
```

Handle for image list functionality.

Created by calling [spinCameraGetNextImageSync\(\)](#), which require a call to [spinImageRelease\(\)](#) to remove from buffer, or [spinImageCreateEmpty\(\)](#), [spinImageCreateEx\(\)](#), or [spinImageCreate\(\)](#), which require a call to [spinImageDestroy\(\)](#) to destroy.

13.13.1.14 spinImageListEventFunction

```
typedef void(* spinImageListEventFunction) (const spinImageList hImage, void *pUserData)
```

13.13.1.15 spinImageListEventHandler

```
typedef void* spinImageListEventHandler
```

Handle for image list event handler functionality.

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

13.13.1.16 spinImageProcessor

```
typedef void* spinImageProcessor
```

Handle for image processor functionality.

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

13.13.1.17 spinImageStatistics

```
typedef void* spinImageStatistics
```

Handle for image statistics functionality.

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

13.13.1.18 spinInterface

```
typedef void* spinInterface
```

Handle for interface functionality.

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

13.13.1.19 spinInterfaceEventHandler

```
typedef void* spinInterfaceEventHandler
```

Handle for interface event handler functionality.

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

13.13.1.20 spinInterfaceList

```
typedef void* spinInterfaceList
```

Handle for interface list functionality.

Created by calling [spinSystemGetInterfaces\(\)](#), which requires a call to [spinInterfaceListClear\(\)](#) to clear, or [spinInterfaceListCreateEmpty\(\)](#), which requires a call to [spinInterfaceListDestroy\(\)](#) to destroy.

13.13.1.21 spinLogEventData

```
typedef void* spinLogEventData
```

Handle for logging event data functionality.

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

13.13.1.22 spinLogEventFunction

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

13.13.1.23 spinLogEventHandler

```
typedef void* spinLogEventHandler
```

Handle for logging event handler functionality.

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

13.13.1.24 spinRemovalEventFunction

```
typedef void(* spinRemovalEventFunction) (const spinCamera hCamera, void *pUserData)
```

13.13.1.25 spinSystem

```
typedef void* spinSystem
```

Handle for system functionality.

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

13.13.1.26 spinVideo

```
typedef void* spinVideo
```

Handle for video recording functionality.

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

13.13.2 Enumeration Type Documentation

13.13.2.1 spinActionCommandStatus

```
enum spinActionCommandStatus
```

Possible Status Codes Returned from Action Command.

Enumerator

SPINNAKER_ACTION_COMMAND_STATUS_OK	The device acknowledged the command.
SPINNAKER_ACTION_COMMAND_STATUS_NO_REF_TIME	
SPINNAKER_ACTION_COMMAND_STATUS_OVERFLOW	
SPINNAKER_ACTION_COMMAND_STATUS_ACTION_LATE	
SPINNAKER_ACTION_COMMAND_STATUS_ERROR	

13.13.2.2 spinColorProcessingAlgorithm

enum `spinColorProcessingAlgorithm`

Color processing algorithms.

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

Enumerator

SPINNAKER_COLOR_PROCESSING_↔ ALGORITHM_NONE	No color processing.
SPINNAKER_COLOR_PROCESSING_↔ ALGORITHM_NEAREST_NEIGHBOR	Fastest but lowest quality. Equivalent to FLYCAPTURE_NEAREST_NEIGHBOR_FAST in FlyCapture.
SPINNAKER_COLOR_PROCESSING_↔ ALGORITHM_NEAREST_NEIGHBOR_AVG	Nearest Neighbor with averaged green pixels. Higher quality but slower compared to nearest neighbor without averaging.
SPINNAKER_COLOR_PROCESSING_↔ ALGORITHM_BILINEAR	Weighted average of surrounding 4 pixels in a 2x2 neighborhood.
SPINNAKER_COLOR_PROCESSING_↔ ALGORITHM_EDGE_SENSING	Weights surrounding pixels based on localized edge orientation.
SPINNAKER_COLOR_PROCESSING_↔ ALGORITHM_HQ_LINEAR	Well-balanced speed and quality.
SPINNAKER_COLOR_PROCESSING_↔ ALGORITHM_IPP	Multi-threaded with similar results to edge sensing.
SPINNAKER_COLOR_PROCESSING_↔ ALGORITHM_DIRECTIONAL_FILTER	Best quality but much faster than rigorous.
SPINNAKER_COLOR_PROCESSING_↔ ALGORITHM_RIGOROUS	Slowest but produces good results.
SPINNAKER_COLOR_PROCESSING_↔ ALGORITHM_WEIGHTED_DIRECTIONAL_FILTER	Weighted pixel average from different directions.

13.13.2.3 spinError

enum `spinError`

The error codes used in Spinnaker C.

These codes are returned from every function in Spinnaker C. The error codes in the range of -2000 to -2999 are reserved for GenICam related errors. The error codes in the range of -3000 to -3999 are reserved for image processing related errors.

Enumerator

SPINNAKER_ERR_SUCCESS	An error code of 0 means that the function has run without error.
SPINNAKER_ERR_ERROR	The error codes in the range of -1000 to -1999 are reserved for Spinnaker exceptions.
SPINNAKER_ERR_NOT_INITIALIZED	
SPINNAKER_ERR_NOT_IMPLEMENTED	
SPINNAKER_ERR_RESOURCE_IN_USE	
SPINNAKER_ERR_ACCESS_DENIED	
SPINNAKER_ERR_INVALID_HANDLE	
SPINNAKER_ERR_INVALID_ID	
SPINNAKER_ERR_NO_DATA	
SPINNAKER_ERR_INVALID_PARAMETER	
SPINNAKER_ERR_IO	
SPINNAKER_ERR_TIMEOUT	
SPINNAKER_ERR_ABORT	
SPINNAKER_ERR_INVALID_BUFFER	
SPINNAKER_ERR_NOT_AVAILABLE	
SPINNAKER_ERR_INVALID_ADDRESS	
SPINNAKER_ERR_BUFFER_TOO_SMALL	
SPINNAKER_ERR_INVALID_INDEX	
SPINNAKER_ERR_PARSING_CHUNK_DATA	
SPINNAKER_ERR_INVALID_VALUE	
SPINNAKER_ERR_RESOURCE_EXHAUSTED	
SPINNAKER_ERR_OUT_OF_MEMORY	
SPINNAKER_ERR_BUSY	
SPINNAKER_ERR_GENICAM_INVALID_ARGUMENT	The error codes in the range of -2000 to -2999 are reserved for Gen API related errors.
SPINNAKER_ERR_GENICAM_OUT_OF_RANGE	
SPINNAKER_ERR_GENICAM_PROPERTY	
SPINNAKER_ERR_GENICAM_RUN_TIME	
SPINNAKER_ERR_GENICAM_LOGICAL	
SPINNAKER_ERR_GENICAM_ACCESS	
SPINNAKER_ERR_GENICAM_TIMEOUT	
SPINNAKER_ERR_GENICAM_DYNAMIC_CAST	
SPINNAKER_ERR_GENICAM_GENERIC	
SPINNAKER_ERR_GENICAM_BAD_ALLOCATION	
SPINNAKER_ERR_IM_CONVERT	The error codes in the range of -3000 to -3999 are reserved for image processing related errors.
SPINNAKER_ERR_IM_COPY	
SPINNAKER_ERR_IM_MALLOC	
SPINNAKER_ERR_IM_NOT_SUPPORTED	
SPINNAKER_ERR_IM_HISTOGRAM_RANGE	
SPINNAKER_ERR_IM_HISTOGRAM_MEAN	
SPINNAKER_ERR_IM_MIN_MAX	
SPINNAKER_ERR_IM_COLOR_CONVERSION	

Enumerator

SPINNAKER_ERR_CUSTOM_ID	Error codes less than -10000 are reserved for user-defined custom errors.
-------------------------	---

13.13.2.4 spinImageFileFormat

enum [spinImageFileFormat](#)

File formats to be used for saving images to disk.

Enumerator

SPINNAKER_IMAGE_FILE_FORMAT_FROM_FILE_EXT	Determine file format from file extension.
SPINNAKER_IMAGE_FILE_FORMAT_PGM	Portable gray map.
SPINNAKER_IMAGE_FILE_FORMAT_PPM	Portable pixmap.
SPINNAKER_IMAGE_FILE_FORMAT_BMP	Bitmap.
SPINNAKER_IMAGE_FILE_FORMAT_JPEG	JPEG.
SPINNAKER_IMAGE_FILE_FORMAT_JPEG2000	JPEG 2000.
SPINNAKER_IMAGE_FILE_FORMAT_TIFF	Tagged image file format.
SPINNAKER_IMAGE_FILE_FORMAT_PNG	Portable network graphics.
SPINNAKER_IMAGE_FILE_FORMAT_RAW	Raw data.
SPINNAKER_IMAGE_FILE_FORMAT_FORCE_32BITS	

13.13.2.5 spinImageStatus

enum [spinImageStatus](#)

Status of images returned from [spinImageGetStatus\(\)](#) call.

Enumerator

SPINNAKER_IMAGE_STATUS_UNKNOWN_ERROR	Image has an unknown error.
SPINNAKER_IMAGE_STATUS_NO_ERROR	Image is returned from GetNextImage() call without any errors.
SPINNAKER_IMAGE_STATUS_CRC_CHECK_FAILED	Image failed CRC check.
SPINNAKER_IMAGE_STATUS_DATA_OVERFLOW	Received more data than the size of the image.
SPINNAKER_IMAGE_STATUS_MISSING_PACKETS	Image has missing packets. Potential fixes include enabling jumbo packets and adjusting packet size/delay. For more information see https://www.flir.com/support-center/iis/machine-vision/application
SPINNAKER_IMAGE_STATUS_LEADER_BUFFER_SIZE_INCONSISTENT	Image leader is incomplete. Could be caused by missing packet(s). See link above.

Enumerator

SPINNAKER_IMAGE_STATUS_TRAILER_↔ BUFFER_SIZE_INCONSISTENT	Image trailer is incomplete. Could be caused by missing packet(s). See link above.
SPINNAKER_IMAGE_STATUS_PACKETID_↔ INCONSISTENT	Image has an inconsistent packet id. Could be caused by missing packet(s). See link above.
SPINNAKER_IMAGE_STATUS_MISSING_LEADER	Image leader is missing. Could be caused by missing packet(s). See link above.
SPINNAKER_IMAGE_STATUS_MISSING_TRAILER	Image trailer is missing. Could be caused by missing packet(s). See link above.
SPINNAKER_IMAGE_STATUS_DATA_INCOMPLETE	Image data is incomplete. Could be caused by missing packet(s). See link above.
SPINNAKER_IMAGE_STATUS_INFO_↔ INCONSISTENT	Image info is corrupted. Could be caused by missing packet(s). See link above.
SPINNAKER_IMAGE_STATUS_CHUNK_DATA_↔ INVALID	Image chunk data is invalid.
SPINNAKER_IMAGE_STATUS_NO_SYSTEM_↔ RESOURCES	Image cannot be processed due to lack of system resources.

13.13.2.6 spinnakerLogLevel

enum `spinnakerLogLevel`

log levels

Enumerator

SPINNAKER_LOG_LEVEL_OFF	
SPINNAKER_LOG_LEVEL_FATAL	
SPINNAKER_LOG_LEVEL_ALERT	
SPINNAKER_LOG_LEVEL_CRIT	
SPINNAKER_LOG_LEVEL_ERROR	
SPINNAKER_LOG_LEVEL_WARN	
SPINNAKER_LOG_LEVEL_NOTICE	
SPINNAKER_LOG_LEVEL_INFO	
SPINNAKER_LOG_LEVEL_DEBUG	
SPINNAKER_LOG_LEVEL_NOTSET	

13.13.2.7 spinStatisticsChannel

enum `spinStatisticsChannel`

Channels that allow statistics to be calculated.

Enumerator

SPINNAKER_STATISTICS_CHANNEL_GREY	
SPINNAKER_STATISTICS_CHANNEL_RED	
SPINNAKER_STATISTICS_CHANNEL_GREEN	
SPINNAKER_STATISTICS_CHANNEL_BLUE	
SPINNAKER_STATISTICS_CHANNEL_HUE	
SPINNAKER_STATISTICS_CHANNEL_SATURATION	
SPINNAKER_STATISTICS_CHANNEL_LIGHTNESS	
SPINNAKER_STATISTICS_CHANNEL_NUM_CHANNELS	

13.13.2.8 spinTIFFCompressionMethod

enum `spinTIFFCompressionMethod`

Compression method to use for encoding TIFF images.

Enumerator

SPINNAKER_TIFF_COMPRESS_METHOD_NONE	
SPINNAKER_TIFF_COMPRESS_METHOD_PACKBITS	
SPINNAKER_TIFF_COMPRESS_METHOD_DEFLATE	
SPINNAKER_TIFF_COMPRESS_METHOD_ADOBE_DEFLATE	
SPINNAKER_TIFF_COMPRESS_METHOD_CCITTFAX3	
SPINNAKER_TIFF_COMPRESS_METHOD_CCITTFAX4	
SPINNAKER_TIFF_COMPRESS_METHOD_LZW	
SPINNAKER_TIFF_COMPRESS_METHOD_JPG	

13.13.2.9 spinTLPayloadType

enum `spinTLPayloadType`

Enumerator

SPINNAKER_TLPAYLOAD_TYPE_UNKNOWN	
SPINNAKER_TLPAYLOAD_TYPE_IMAGE	
SPINNAKER_TLPAYLOAD_TYPE_RAW_DATA	
SPINNAKER_TLPAYLOAD_TYPE_FILE	
SPINNAKER_TLPAYLOAD_TYPE_CHUNK_DATA	
SPINNAKER_TLPAYLOAD_TYPE_JPEG	
SPINNAKER_TLPAYLOAD_TYPE_JPEG2000	
SPINNAKER_TLPAYLOAD_TYPE_H264	
SPINNAKER_TLPAYLOAD_TYPE_CHUNK_ONLY	
SPINNAKER_TLPAYLOAD_TYPE_DEVICE_SPECIFIC	
SPINNAKER_TLPAYLOAD_TYPE_MULTI_PART	

Enumerator

SPINNAKER_TLPAYLOAD_TYPE_CUSTOM_ID	
SPINNAKER_TLPAYLOAD_TYPE_LOSSLESS_COMPRESSED	
SPINNAKER_TLPAYLOAD_TYPE_LOSSY_COMPRESSED	
SPINNAKER_TLPAYLOAD_TYPE_JPEG_LOSSLESS_COMPRESSED	

13.13.2.10 spinTLPixelFormatNamespace

enum [spinTLPixelFormatNamespace](#)

This enum represents the namespace in which the TL specific pixel format resides.

This enum is returned from a captured image when calling [spinImageGetTLPixelFormatNamespace\(\)](#). It can be used to interpret the raw pixel format returned from [spinImageGetTLPixelFormat\(\)](#).

See also

[spinImageGetTLPixelFormat\(\)](#)

[spinImageGetTLPixelFormatNamespace\(\)](#)

Enumerator

SPINNAKER_TLPIXELFORMAT_NAMESPACE_UNKNOWN	
SPINNAKER_TLPIXELFORMAT_NAMESPACE_GEV	
SPINNAKER_TLPIXELFORMAT_NAMESPACE_IIDC	
SPINNAKER_TLPIXELFORMAT_NAMESPACE_PFNC_16BIT	
SPINNAKER_TLPIXELFORMAT_NAMESPACE_PFNC_32BIT	
SPINNAKER_PIXELFORMAT_NAMESPACE_CUSTOM_ID	

13.13.3 Variable Documentation**13.13.3.1 False**

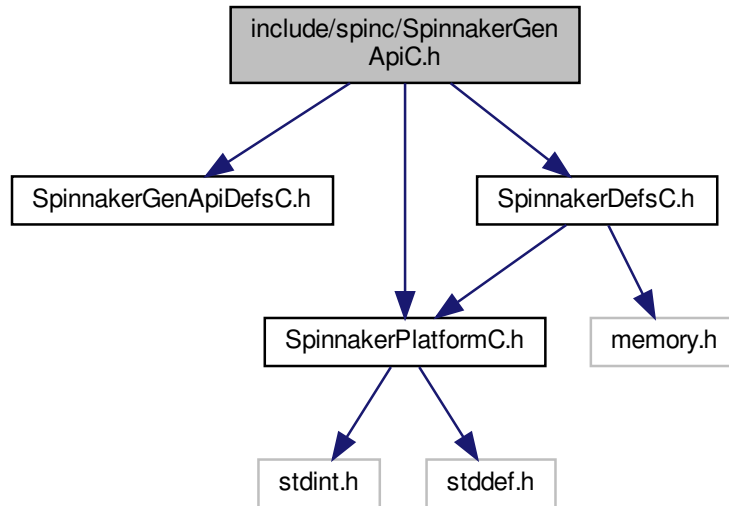
```
const bool8_t False = 0 [static]
```

13.13.3.2 True

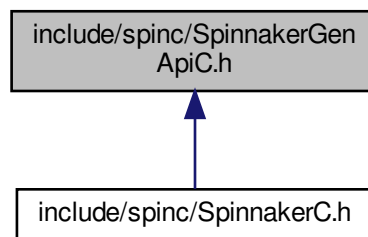
```
const bool8_t True = 1 [static]
```

13.14 include/spinc/SpinnakerGenApiC.h File Reference

Include dependency graph for SpinnakerGenApiC.h:



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



Functions

- [SPINNAKERC_API spinNodeMapGetNode](#) ([spinNodeMapHandle](#) hNodeMap, const char *pName, [spinNodeHandle](#) *phNode)
Retrieves a node from the nodemap by name.
- [SPINNAKERC_API spinNodeMapGetNumNodes](#) ([spinNodeMapHandle](#) hNodeMap, [size_t](#) *pValue)
Gets the number of nodes in the map.
- [SPINNAKERC_API spinNodeMapGetNodeByIndex](#) ([spinNodeMapHandle](#) hNodeMap, [size_t](#) index, [spinNodeHandle](#) *phNode)

- Retrieves a node from the nodemap by index.*

 - [SPINNAKERC_API spinNodeMapReleaseNode](#) ([spinNodeMapHandle](#) hNodeMap, [spinNodeHandle](#) hNode)

Releases the entry node handle.
- [SPINNAKERC_API spinNodeMapPoll](#) ([spinNodeMapHandle](#) hNodeMap, [int64_t](#) timestamp)

Fires nodes which have a polling time.
- [SPINNAKERC_API 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](#) *p↔BufLen)

Retrieves the value of any node type as a c-string; manually set whether to verify the node.
- [SPINNAKERC_API spinNodeFromString](#) ([spinNodeHandle](#) hNode, const char *pBuf)

Sets the value of any node type from a c-string; it is important to ensure that the value of the c-string is appropriate to the node type.
- [SPINNAKERC_API spinNodeFromStringEx](#) ([spinNodeHandle](#) hNode, [bool8_t](#) bVerify, const char *pBuf)

Sets the value of any node type from a c-string; manually set whether to verify the node; ensure the value of the c-string is appropriate to the node type.
- [SPINNAKERC_API spinStringSetValue](#) ([spinNodeHandle](#) hNode, const char *pBuf)

Sets the value of a string node.
- [SPINNAKERC_API spinStringSetValueEx](#) ([spinNodeHandle](#) hNode, [bool8_t](#) bVerify, const char *pBuf)

Sets the value of a string node; manually set whether to verify the node.
- [SPINNAKERC_API spinStringGetValue](#) ([spinNodeHandle](#) hNode, char *pBuf, [size_t](#) *pBufLen)

Retrieves the value of a string node as a c-string.
- [SPINNAKERC_API spinStringGetValueEx](#) ([spinNodeHandle](#) hNode, [bool8_t](#) bVerify, char *pBuf, [size_t](#) *p↔BufLen)

Retrieves the value of a string node as a cstring; manually set whether to verify the node.
- [SPINNAKERC_API spinStringGetMaxLength](#) ([spinNodeHandle](#) hNode, [int64_t](#) *pValue)

Retrieves the maximum length of the c-string to be returned.
- [SPINNAKERC_API spinIntegerSetValue](#) ([spinNodeHandle](#) hNode, [int64_t](#) value)

Sets the value of an integer node.
- [SPINNAKERC_API spinIntegerSetValueEx](#) ([spinNodeHandle](#) hNode, [bool8_t](#) bVerify, [int64_t](#) value)

Sets the value of an integer node; manually set whether to verify the node.
- [SPINNAKERC_API spinIntegerGetValue](#) ([spinNodeHandle](#) hNode, [int64_t](#) *pValue)

Retrieves the value of an integer node.
- [SPINNAKERC_API spinIntegerGetValueEx](#) ([spinNodeHandle](#) hNode, [bool8_t](#) bVerify, [int64_t](#) *pValue)

Retrieves the value of an integer node; manually set whether to verify the node.
- [SPINNAKERC_API spinIntegerGetMin](#) ([spinNodeHandle](#) hNode, [int64_t](#) *pValue)

Retrieves the minimum value of an integer node; all potential values must be greater than or equal to the minimum.
- [SPINNAKERC_API spinIntegerGetMax](#) ([spinNodeHandle](#) hNode, [int64_t](#) *pValue)

Retrieves the maximum value of an integer node; all potential values must be lesser than or equal to the maximum.
- [SPINNAKERC_API spinIntegerGetInc](#) ([spinNodeHandle](#) hNode, [int64_t](#) *pValue)

Retrieves the increment of an integer node; all possible values must be divisible by the increment.
- [SPINNAKERC_API spinIntegerGetRepresentation](#) ([spinNodeHandle](#) hNode, [spinRepresentation](#) *pValue)

Retrieves the numerical representation of the value of a node; i.e.
- [SPINNAKERC_API spinFloatSetValue](#) ([spinNodeHandle](#) hNode, double value)

Sets the value of a float node.
- [SPINNAKERC_API spinFloatSetValueEx](#) ([spinNodeHandle](#) hNode, [bool8_t](#) bVerify, double value)

Sets the value of a float node; manually set whether to verify the node.
- [SPINNAKERC_API spinFloatGetValue](#) ([spinNodeHandle](#) hNode, double *pValue)

Retrieves the value of a float node.
- [SPINNAKERC_API spinFloatGetValueEx](#) ([spinNodeHandle](#) hNode, [bool8_t](#) bVerify, double *pValue)

Retrieves the value of a float node; manually set whether to verify the node.
- [SPINNAKERC_API spinFloatGetMin](#) ([spinNodeHandle](#) hNode, double *pValue)

Retrieves the minimum value of a float node; all potential values must be greater than or equal to the minimum.
- [SPINNAKERC_API spinFloatGetMax](#) ([spinNodeHandle](#) hNode, double *pValue)

Retrieves the maximum value of a float node; all potential values must be lesser than or equal to the maximum.
- [SPINNAKERC_API spinFloatGetRepresentation](#) ([spinNodeHandle](#) hNode, [spinRepresentation](#) *pValue)

Retrieves the numerical representation of the value of a node; i.e.
- [SPINNAKERC_API spinFloatGetUnit](#) ([spinNodeHandle](#) hNode, char *pBuf, [size_t](#) *pBufLen)

- Retrieves the units of the float node value.*

 - [SPINNAKERC_API spinEnumerationGetNumEntries](#) ([spinNodeHandle](#) hEnumNode, [size_t](#) *pValue)

Retrieves the number of entries of an enum node.

 - [SPINNAKERC_API spinEnumerationGetEntryByIndex](#) ([spinNodeHandle](#) hEnumNode, [size_t](#) index, [spinNodeHandle](#) *phEntry)

Retrieves an entry node from an enum node using an index.

 - [SPINNAKERC_API spinEnumerationGetEntryByName](#) ([spinNodeHandle](#) hEnumNode, [const char](#) *pName, [spinNodeHandle](#) *phEntry)

Retrieves an entry node from an enum node using the entry's symbolic.

 - [SPINNAKERC_API spinEnumerationGetCurrentEntry](#) ([spinNodeHandle](#) hEnumNode, [spinNodeHandle](#) *phEntry)

Retrieves the currently selected entry node from an enum node.

 - [SPINNAKERC_API spinEnumerationReleaseNode](#) ([spinNodeHandle](#) hEnumNode, [spinNodeHandle](#) hEntry)

Releases the entry node from the enum node handle.

 - [SPINNAKERC_API spinEnumerationSetIntValue](#) ([spinNodeHandle](#) hEnumNode, [int64_t](#) value)

Sets a new entry using its integer value retrieved from a call to [spinEnumerationEntryGetIntValue\(\)](#); note that enumeration entry int and enum values are different - int values defined on camera, enum values found in [SpinnakerDefsC.h](#).

 - [SPINNAKERC_API spinEnumerationSetEnumValue](#) ([spinNodeHandle](#) hEnumNode, [size_t](#) value)

Sets a new entry using its enum; note that enumeration entry int and enum values are different - int values defined on camera, enum values found in [SpinnakerDefsC.h](#).

 - [SPINNAKERC_API spinEnumerationEntryGetIntValue](#) ([spinNodeHandle](#) hNode, [int64_t](#) *pValue)

Retrieves the integer value of an entry node; note that enumeration entry int and enum values are different - int values defined on camera, enum values found in [SpinnakerDefsC.h](#).

 - [SPINNAKERC_API spinEnumerationEntryGetEnumValue](#) ([spinNodeHandle](#) hNode, [size_t](#) *pValue)

Retrieves the enum value (as an integer) of an entry node; note that 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.

13.14.1 Function Documentation

13.14.1.1 spinBooleanGetValue()

```
SPINNAKERC_API spinBooleanGetValue (  
    spinNodeHandle hNode,  
    bool8\_t * pbValue )
```

Retrieves the value of a boolean node; boolean values are represented by 'True' (which equals '0') and 'False' (which equals '1')

See also

[spinError](#)

Parameters

<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

13.14.1.2 spinBooleanSetValue()

```
SPINNAKERC_API spinBooleanSetValue (  
    spinNodeHandle hNode,  
    bool8\_t value )
```

Sets the value of a boolean node; boolean values are represented by 'True' (which equals '0') and 'False' (which equals '1')

See also

[spinError](#)

Parameters

<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

13.14.1.3 spinCategoryGetFeatureByIndex()

```
SPINNAKERC_API spinCategoryGetFeatureByIndex (
    spinNodeHandle hCategoryNode,
    size_t index,
    spinNodeHandle * phFeature )
```

Retrieves a node from a category node using an index.

See also

[spinError](#)

Parameters

<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

13.14.1.4 spinCategoryGetNumFeatures()

```
SPINNAKERC_API spinCategoryGetNumFeatures (
    spinNodeHandle hCategoryNode,
    size_t * pValue )
```

Retrieves the number of a features (or child nodes) or a category node.

See also

[spinError](#)

Parameters

<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

13.14.1.5 spinCategoryReleaseNode()

```
SPINNAKERC_API spinCategoryReleaseNode (
    spinNodeHandle hCategoryNode,
    spinNodeHandle hFeature )
```

Releases the feature node from the category node.

Make sure node handle is cleaned up properly by setting it to NULL after the node is released. If this function is not explicitly called, the handle will be released upon the release of the camera handle.

See also

[spinCameraRelease](#)

[spinError](#)

Parameters

<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

13.14.1.6 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

13.14.1.7 spinCommandIsDone()

```
SPINNAKERC_API spinCommandIsDone (
    spinNodeHandle hNode,
    bool8_t * pbValue )
```

Retrieves whether or not the action of a command node has completed.

See also

[spinError](#)

Parameters

<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

13.14.1.8 spinEnumerationEntryGetEnumValue()

```
SPINNAKERC_API spinEnumerationEntryGetEnumValue (
    spinNodeHandle hNode,
    size_t * pValue )
```

Retrieves the enum value (as an integer) of an entry node; note that enumeraiton entry int and enum values are different - int values defined on camera, enum values found in [SpinnakerDefsC.h](#).

See also

[spinEnumerationSetEnumValue\(\)](#)

[spinError](#)

Parameters

<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

13.14.1.9 spinEnumerationEntryGetIntValue()

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

Retrieves the integer value of an entry node; note that enumeration entry int and enum values are different - int values defined on camera, enum values found in [SpinnakerDefsC.h](#).

See also

[spinEnumerationSetIntValue\(\)](#)
[spinError](#)

Parameters

<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

13.14.1.10 spinEnumerationEntryGetSymbolic()

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

Retrieves the symbolic of an entry node as a c-string.

See also

[spinError](#)

Parameters

<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

13.14.1.11 spinEnumerationGetCurrentEntry()

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

Retrieves the currently selected entry node from an enum node.

See also

[spinError](#)

Parameters

<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

13.14.1.12 spinEnumerationGetEntryByIndex()

```
SPINNAKERC_API spinEnumerationGetEntryByIndex (
    spinNodeHandle hEnumNode,
    size_t index,
    spinNodeHandle * phEntry )
```

Retrieves an entry node from an enum node using an index.

See also

[spinError](#)

Parameters

<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

13.14.1.13 spinEnumerationGetEntryByName()

```
SPINNAKERC_API spinEnumerationGetEntryByName (
    spinNodeHandle hEnumNode,
    const char * pName,
    spinNodeHandle * phEntry )
```

Retrieves an entry node from an enum node using the entry's symbolic.

See also

[spinError](#)

Parameters

<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

13.14.1.14 spinEnumerationGetNumEntries()

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

Retrieves the number of entries of an enum node.

See also

[spinError](#)

Parameters

<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

13.14.1.15 spinEnumerationReleaseNode()

```
SPINNAKERC_API spinEnumerationReleaseNode (
    spinNodeHandle hEnumNode,
    spinNodeHandle hEntry )
```

Releases the entry node from the enum node handle.

Make sure node handle is cleaned up properly by setting it to NULL after the node is released. If this function is not explicitly called, the handle will be released upon the release of the camera handle.

See also

[spinCameraRelease](#)

[spinError](#)

Parameters

<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

13.14.1.16 spinEnumerationSetEnumValue()

```
SPINNAKERC_API spinEnumerationSetEnumValue (
    spinNodeHandle hEnumNode,
    size_t value )
```

Sets a new entry using its enum; note that enumeration entry int and enum values are different - int values defined on camera, enum values found in [SpinnakerDefsC.h](#).

See also

[spinEnumerationEntryGetEnumValue\(\)](#)

[spinError](#)

Parameters

<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

13.14.1.17 spinEnumerationSetIntValue()

```
SPINNAKERC_API spinEnumerationSetIntValue (
    spinNodeHandle hEnumNode,
    int64_t value )
```

Sets a new entry using its integer value retrieved from a call to [spinEnumerationEntryGetIntValue\(\)](#); note that enumeration entry int and enum values are different - int values defined on camera, enum values found in [SpinnakerDefsC.h](#).

See also

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

Parameters

<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

13.14.1.18 spinFloatGetMax()

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

Retrieves the maximum value of a float node; all potential values must be lesser than or equal to the maximum.

See also

[spinError](#)

Parameters

<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

13.14.1.19 spinFloatGetMin()

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

Retrieves the minimum value of a float node; all potential values must be greater than or equal to the minimum.

See also

[spinError](#)

Parameters

<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

13.14.1.20 spinFloatGetRepresentation()

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

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

linear, logarithmic, 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

13.14.1.21 spinFloatGetUnit()

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

Retrieves the units of the float node value.

See also

[spinError](#)

Parameters

<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

13.14.1.22 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

13.14.1.23 spinFloatGetValueEx()

```
SPINNAKERC_API spinFloatGetValueEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    double * pValue )
```

Retrieves the value of a float node; manually set whether to verify the node.

See also

[spinError](#)

Parameters

<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

13.14.1.24 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

13.14.1.25 spinFloatSetValueEx()

```
SPINNAKERC_API spinFloatSetValueEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    double value )
```

Sets the value of a float node; manually set whether to verify the node.

See also

[spinError](#)

Parameters

<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

13.14.1.26 spinIntegerGetInc()

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

Retrieves the increment of an integer node; all possible values must be divisible by the increment.

See also

[spinError](#)

Parameters

<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

13.14.1.27 spinIntegerGetMax()

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

Retrieves the maximum value of an integer node; all potential values must be lesser than or equal to the maximum.

See also

[spinError](#)

Parameters

<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

13.14.1.28 spinIntegerGetMin()

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

Retrieves the minimum value of an integer node; all potential values must be greater than or equal to the minimum.

See also

[spinError](#)

Parameters

<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

13.14.1.29 spinIntegerGetRepresentation()

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

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

linear, logarithmic, 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

13.14.1.30 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

13.14.1.31 spinIntegerGetValueEx()

```
SPINNAKERC_API spinIntegerGetValueEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    int64_t * pValue )
```

Retrieves the value of an integer node; manually set whether to verify the node.

See also

[spinError](#)

Parameters

<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

13.14.1.32 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

13.14.1.33 spinIntegerSetValueEx()

```
SPINNAKERC_API spinIntegerSetValueEx (  
    spinNodeHandle hNode,  
    bool8_t bVerify,  
    int64_t value )
```

Sets the value of an integer node; manually set whether to verify the node.

See also

[spinError](#)

Parameters

<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

13.14.1.34 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

13.14.1.35 spinNodeFromString()

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

Sets the value of any node type from a c-string; it is important to ensure that the value of the c-string is appropriate to the node type.

See also

[spinError](#)

Parameters

<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

13.14.1.36 spinNodeFromStringEx()

```
SPINNAKERC_API spinNodeFromStringEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    const char * pBuf )
```

Sets the value of any node type from a c-string; manually set whether to verify the node; ensure the value of the c-string is appropriate to the node type.

See also

[spinError](#)

Parameters

<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

13.14.1.37 spinNodeGetAccessMode()

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

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

See also

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

Parameters

<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

13.14.1.38 spinNodeGetCachingMode()

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

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

See also

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

Parameters

<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

13.14.1.39 spinNodeGetDescription()

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

Retrieves a longer description of a node.

See also

[spinError](#)

Parameters

<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

13.14.1.40 spinNodeGetDisplayName()

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

Retrieves the display name of a node (whitespace possible)

See also

[spinError](#)

Parameters

<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

13.14.1.41 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

13.14.1.42 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

13.14.1.43 spinNodeGetName()

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

Retrieves the name of a node (no whitespace)

See also

[spinError](#)

Parameters

<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

13.14.1.44 spinNodeGetNameSpace()

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

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

See also

[spinError](#)

[spinNameSpace](#)

Parameters

<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

13.14.1.45 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

13.14.1.46 spinNodeGetToolTip()

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

Retrieves a short description of a node.

See also

[spinError](#)

Parameters

<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

13.14.1.47 spinNodeGetType()

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

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

See also

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

Parameters

<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

13.14.1.48 spinNodeGetVisibility()

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

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

See also

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

Parameters

<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

13.14.1.49 spinNodeInvalidateNode()

```
SPINNAKERC_API spinNodeInvalidateNode (  
    spinNodeHandle hNode )
```

Invalidates a node in case its values may have changed, rendering it no longer valid.

See also

[spinError](#)

Parameters

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

Returns

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

13.14.1.50 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

13.14.1.51 spinNodeIsEqual()

```
SPINNAKERC_API spinNodeIsEqual (
    spinNodeHandle hNodeFirst,
    spinNodeHandle hNodeSecond,
    bool8_t * pbResult )
```

Checks whether two nodes are equal.

See also

[spinError](#)

Parameters

<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

13.14.1.52 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

13.14.1.53 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

13.14.1.54 spinNodeIsWritable()

```
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

13.14.1.55 spinNodeMapGetNode()

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

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>pNode</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

13.14.1.56 spinNodeMapGetNodeByIndex()

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

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>pNode</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

13.14.1.57 spinNodeMapGetNumNodes()

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

Gets the number of nodes in the map.

See also

[spinError](#)

Parameters

<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

13.14.1.58 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

13.14.1.59 spinNodeMapReleaseNode()

```
SPINNAKERC_API spinNodeMapReleaseNode (
    spinNodeMapHandle hNodeMap,
    spinNodeHandle hNode )
```

Releases the entry node handle.

Make sure node handle is cleaned up properly by setting it to NULL after the node is released. If this function is not explicitly called, the handle will be released upon the release of the camera handle.

See also

[spinCameraRelease](#)

[spinError](#)

Parameters

<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

13.14.1.60 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 callback

Returns

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

13.14.1.61 spinNodeToString()

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

Retrieves the value of any node type as a c-string.

See also

[spinError](#)

Parameters

<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

13.14.1.62 spinNodeToStringEx()

```
SPINNAKERC_API spinNodeToStringEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the value of any node type as a c-string; manually set whether to verify the node.

See also

[spinError](#)

Parameters

<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

13.14.1.63 spinRegisterGet()

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

Retrieves the value of a register node.

See also

[spinError](#)

Parameters

<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

13.14.1.64 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

13.14.1.65 spinRegisterGetEx()

```
SPINNAKERC_API spinRegisterGetEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    bool8_t bIgnoreCache,
    uint8_t * pBuf,
    int64_t length )
```

Retrieves the value of a register node; manually set whether to verify the node and whether to ignore the cache.

See also

[spinError](#)

Parameters

<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

13.14.1.66 spinRegisterGetLength()

```
SPINNAKERC_API spinRegisterGetLength (
    spinNodeHandle hNode,
    int64_t * pLength )
```

Retrieves the length (in bytes) of the value of a register node.

See also

[spinError](#)

Parameters

<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

13.14.1.67 spinRegisterSet()

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

Sets the value of a register node.

See also

[spinError](#)

Parameters

<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

13.14.1.68 spinRegisterSetEx()

```
SPINNAKERC_API spinRegisterSetEx (  
    spinNodeHandle hNode,  
    bool8_t bVerify,  
    const uint8_t * pBuf,  
    int64_t length )
```

Sets the value of a register node; manually set whether to verify the node.

See also

[spinError](#)

Parameters

<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

13.14.1.69 spinRegisterSetReference()

```
SPINNAKERC_API spinRegisterSetReference (
    spinNodeHandle hNode,
    spinNodeHandle hRef )
```

Uses a second node as a reference for a register node.

See also

[spinError](#)

Parameters

<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

13.14.1.70 spinStringGetMaxLength()

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

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

See also

[spinError](#)

Parameters

<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

13.14.1.71 spinStringGetValue()

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

Retrieves the value of a string node as a c-string.

See also

[spinError](#)

Parameters

<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

13.14.1.72 spinStringGetValueEx()

```
SPINNAKERC_API spinStringGetValueEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the value of a string node as a cstring; manually set whether to verify the node.

See also

[spinError](#)

Parameters

<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

13.14.1.73 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

13.14.1.74 spinStringSetValueEx()

```
SPINNAKERC_API spinStringSetValueEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    const char * pBuf )
```

Sets the value of a string node; manually set whether to verify the node.

See also

[spinError](#)

Parameters

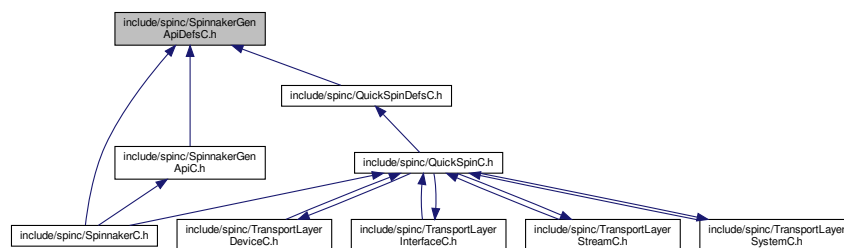
<i>hNode</i>	The string node having its value changed
<i>bVerify</i>	The boolean of whether to verify the node
<i>pBuf</i>	The c-string of the value to set

Returns

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

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 alternative.

- 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` ,
`intflPort` }
typedef for interface type
- enum `spinLinkType` {
`ctAllDependingNodes` ,
`ctAllTerminalNodes` ,
`ctInvalidators` ,
`ctReadingChildren` ,
`ctWritingChildren` ,
`ctDependingChildren` }
typedef for link type
- enum `spinIncMode` {
`noIncrement` ,
`fixedIncrement` ,
`listIncrement` }
typedef for increment mode
- enum `spinInputDirection` {
`idFrom` ,
`idTo` ,
`idNone` }
typedef for link type

13.15.1 Typedef Documentation

13.15.1.1 spinNodeCallbackFunction

```
typedef void(* spinNodeCallbackFunction) (spinNodeHandle hNode)
```

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

13.15.1.2 spinNodeCallbackHandle

```
typedef void* spinNodeCallbackHandle
```

Handle for callback functionality.

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

13.15.1.3 spinNodeHandle

```
typedef void* spinNodeHandle
```

Handle for node functionality.

Created by calling [spinNodeMapGetNode\(\)](#). No need to release, clear, or destroy.

13.15.1.4 spinNodeMapHandle

```
typedef void* spinNodeMapHandle
```

Handle for nodemap functionality.

Created by calling [spinCameraGetNodemap\(\)](#), [spinCameraGetTLDeviceNodeMap\(\)](#), [spinCameraGetTLStreamNodeMap\(\)](#) or [spinInterfaceGetTLNodeMap\(\)](#). No need to release, clear, or destroy.

13.15.2 Enumeration Type Documentation

13.15.2.1 spinAccessMode

```
enum spinAccessMode
```

Enumerator

NI	
NA	
WO	
RO	
RW	
_UndefinedAccesMode	
_CycleDetectAccesMode	

13.15.2.2 spinCachingMode

```
enum spinCachingMode
```

Enumerator

NoCache	
WriteThrough	
WriteAround	
_UndefinedCachingMode	

13.15.2.3 spinDisplayNotation

```
enum spinDisplayNotation
```

```
typedef for float notation
```

Enumerator

fnAutomatic	
fnFixed	the notation if either scientific or fixed depending on what is shorter
fnScientific	the notation is fixed, e.g. 123.4
_UndefinedEDisplayNotation	<p>the notation is scientific, e.g. 1.234e2</p> <p>Object is not yet initialized</p>

13.15.2.4 spinEndianness

enum `spinEndianness`

Endianness of a value in a register.

Enumerator

BigEndian	Register is big endian.
LittleEndian	Register is little endian.
_UndefinedEndian	Object is not yet initialized.

13.15.2.5 spinIncMode

enum `spinIncMode`

typedef for increment mode

Enumerator

noIncrement	
fixedIncrement	
listIncrement	

13.15.2.6 spinInputDirection

enum `spinInputDirection`

typedef for link type

Enumerator

idFrom	
idTo	Indicates a swiss knife that it is used as worker for a converter computing FROM
idNone	Indicates a swiss knife that it is used as worker for a converter computing TO SwissKnife is not used within a converter

13.15.2.7 spinInterfaceType

enum `spinInterfaceType`

typedef for interface type

Enumerator

intflValue	
intflBase	IValue interface
intflInteger	IBase interface
intflBoolean	IInteger interface
intflCommand	IBoolean interface
intflFloat	ICommand interface
intflString	IFloat interface
intflRegister	IString interface
intflCategory	IRegister interface
intflEnumeration	ICategory interface
intflEnumEntry	IEnumeration interface
intflPort	IEnumEntry interface IPort interface

13.15.2.8 spinLinkType

enum `spinLinkType`

typedef for link type

Enumerator

ctAllDependingNodes	
ctAllTerminalNodes	All nodes which will be invalidated if this node becomes invalid
ctInvalidators	All terminal nodes which may be written to by this node
ctReadingChildren	List of references to nodes which may invalidate this node
ctWritingChildren	All child nodes which influence this node's AccessMode
ctDependingChildren	All child nodes which may be written to All child nodes which will cause this node to be invalidated

13.15.2.9 spinNameSpace

enum `spinNameSpace`

Defines if a node name is standard or custom.

Enumerator

Custom	name resides in custom namespace
Standard	name resides in one of the standard namespaces
_UndefinedNameSpace	Object is not yet initialized.

13.15.2.10 spinNodeType

enum `spinNodeType`

Enumerator

ValueNode	
BaseNode	
IntegerNode	
BooleanNode	
FloatNode	
CommandNode	
StringNode	

Enumerator

RegisterNode	
EnumerationNode	
EnumEntryNode	
CategoryNode	
PortNode	
UnknownNode	

13.15.2.11 spinRepresentation

enum `spinRepresentation`

recommended representation of a node value

Enumerator

Linear	Slider with linear behavior.
Logarithmic	Slider with logarithmic behaviour.
Boolean	Check box.
PureNumber	Decimal number in an edit control.
HexNumber	Hex number in an edit control.
IPV4Address	IP-Address.
MACAddress	MAC-Address.
_UndefinedRepresentation	

13.15.2.12 spinSign

enum `spinSign`

Enumerator

Signed	
Unsigned	
_UndefinedSign	

13.15.2.13 spinSlope

enum `spinSlope`

typedef for fomula type

Enumerator

Increasing	
Decreasing	strictly monotonous increasing
Varying	strictly monotonous decreasing
Automatic	slope changes, e.g. at run-time
_UndefinedESlope	slope is determined automatically by probing the function Object is not yet initialized

13.15.2.14 spinStandardNameSpace

enum `spinStandardNameSpace`

Defines from which standard namespace a node name comes from.

Enumerator

None	name resides in custom namespace
GEV	name resides in GigE Vision namespace
IIDC	name resides in 1394 IIDC namespace
CL	name resides in camera link namespace
USB	name resides in USB namespace
_UndefinedStandardNameSpace	Object is not yet initialized.

13.15.2.15 spinVisibility

enum `spinVisibility`

Enumerator

Beginner	
Expert	
Guru	
Invisible	
_UndefinedVisibility	

13.15.2.16 spinXMLValidation

enum `spinXMLValidation`

typedef describing the different validity checks which can be performed on an XML file

The enum values for a bitfield of length `uint32_t`

Enumerator

<code>xvLoad</code>	
<code>xvCycles</code>	Creates a dummy node map
<code>xvSFNC</code>	checks for write and dependency cycles (implies <code>xvLoad</code>)
<code>xvDefault</code>	checks for conformance with the standard feature naming convention (SFNC)
<code>xvAll</code>	checks performed if nothing else is said
<code>_UndefinedEXMLValidation</code>	all possible checks Object is not yet initialized

13.15.2.17 spinYesNo

enum `spinYesNo`

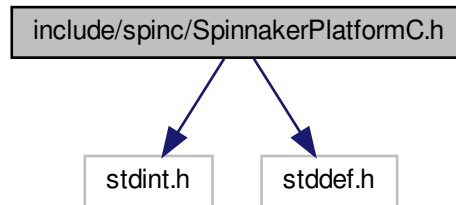
Defines the choices of a Yes/No alternative.

Enumerator

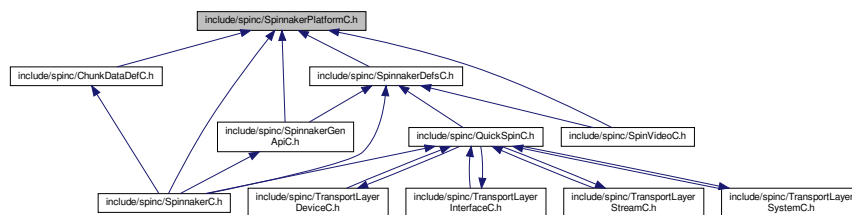
<code>Yes</code>	<code>yes</code>
<code>No</code>	<code>no</code>
<code>_UndefinedYesNo</code>	Object is not yet initialized.

13.16 include/spinc/SpinnakerPlatformC.h File Reference

Include dependency graph for SpinnakerPlatformC.h:



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



Macros

- #define `SPINNAKERC_API` `SPINC_IMPORT_EXPORT` `spinError` `SPINC_CALLTYPE`

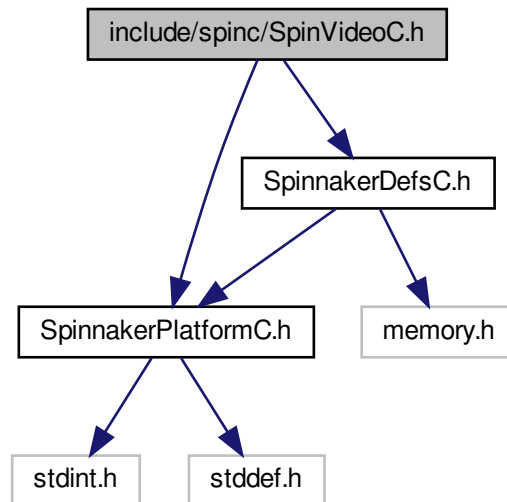
13.16.1 Macro Definition Documentation

13.16.1.1 SPINNAKERC_API

```
#define SPINNAKERC_API SPINC_IMPORT_EXPORT spinError SPINC_CALLTYPE
```

13.17 include/spinc/SpinVideoC.h File Reference

Include dependency graph for SpinVideoC.h:



Functions

- [SPINNAKERC_API spinVideoOpenUncompressed](#) ([spinVideo](#) *phSpinVideo, const char *pName, [spinAVIOption](#) option)
- [SPINNAKERC_API 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)

13.17.1 Function Documentation

13.17.1.1 spinVideoAppend()

```

SPINNAKERC_API spinVideoAppend (
    spinVideo hSpinVideo,
    spinImage hImage )

```

13.17.1.2 spinVideoClose()

```
SPINNAKERC_API spinVideoClose (
    spinVideo hSpinVideo )
```

13.17.1.3 spinVideoOpenH264()

```
SPINNAKERC_API spinVideoOpenH264 (
    spinVideo * phSpinVideo,
    const char * pName,
    spinH264Option option )
```

13.17.1.4 spinVideoOpenMJPEG()

```
SPINNAKERC_API spinVideoOpenMJPEG (
    spinVideo * phSpinVideo,
    const char * pName,
    spinMJPEGOption option )
```

13.17.1.5 spinVideoOpenUncompressed()

```
SPINNAKERC_API spinVideoOpenUncompressed (
    spinVideo * phSpinVideo,
    const char * pName,
    spinAVIOption option )
```

13.17.1.6 spinVideoSetMaximumFileSize()

```
SPINNAKERC_API spinVideoSetMaximumFileSize (
    spinVideo hSpinVideo,
    unsigned int size )
```

Set the maximum file size (in megabytes) of a AVI/MP4 file.

A new AVI/MP4 file is created automatically when file size limit is reached. Setting a maximum size of 0 indicates no limit on file size.

Parameters

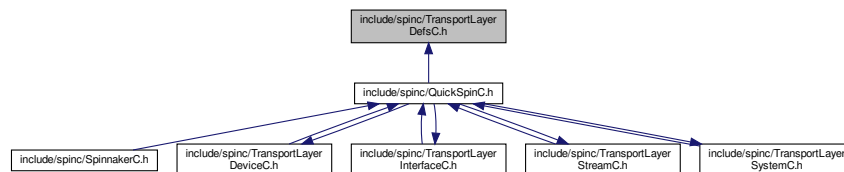
<i>hSpinVideo</i>	The spin video recorder to append the image to
<i>size</i>	The maximum video file size in MB.

Returns

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

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 [spinTLStreamModeEnums](#) {
[StreamMode_Socket](#) ,
[StreamMode_LWF](#) ,
[StreamMode_MVA](#) ,
[NUMSTREAMMODE](#) }
- enum [spinTLStreamBufferCountModeEnums](#) {
[StreamBufferCountMode_Manual](#) ,
[NUMSTREAMBUFFERCOUNTMODE](#) }
- enum [spinTLStreamBufferHandlingModeEnums](#) {
[StreamBufferHandlingMode_OldestFirst](#) ,
[StreamBufferHandlingMode_OldestFirstOverwrite](#) ,
[StreamBufferHandlingMode_NewestOnly](#) ,
[StreamBufferHandlingMode_NewestFirst](#) ,
[NUMSTREAMBUFFERHANDLINGMODE](#) }
- enum [spinTLDeviceTypeEnums](#) {
[DeviceType_GigEVision](#) ,
[DeviceType_CameraLink](#) ,
[DeviceType_CameraLinkHS](#) ,
[DeviceType_CoaXPress](#) ,
[DeviceType_USB3Vision](#) ,
[DeviceType_Custom](#) ,
[NUMDEVICETYPE](#) }

- enum [spinTLDeviceAccessStatusEnums](#) {
[DeviceAccessStatus_Unknown](#) ,
[DeviceAccessStatus_ReadWrite](#) ,
[DeviceAccessStatus_ReadOnly](#) ,
[DeviceAccessStatus_NoAccess](#) ,
[DeviceAccessStatus_Busy](#) ,
[DeviceAccessStatus_OpenReadWrite](#) ,
[DeviceAccessStatus_OpenReadOnly](#) ,
[NUMDEVICEACCESSSTATUS](#) }
- enum [spinTLGenICamXMLLocationEnums](#) {
[GenICamXMLLocation_Device](#) ,
[GenICamXMLLocation_Host](#) ,
[NUMGENICAMXMLLOCATION](#) }
- enum [spinTLGUIXMLLocationEnums](#) {
[GUIXMLLocation_Device](#) ,
[GUIXMLLocation_Host](#) ,
[NUMGUIXMLLOCATION](#) }
- enum [spinTLGevCCPEnums](#) {
[GevCCP_EnumEntry_GevCCP_OpenAccess](#) ,
[GevCCP_EnumEntry_GevCCP_ExclusiveAccess](#) ,
[GevCCP_EnumEntry_GevCCP_ControlAccess](#) ,
[NUMGEVCCP](#) }
- enum [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 }
```

13.18.1 Enumeration Type Documentation

13.18.1.1 spinTLDeviceAccessStatusEnums

```
enum spinTLDeviceAccessStatusEnums
```

< Gets the access status the transport layer Producer has on the device.

Enumerator

DeviceAccessStatus_Unknown	Not known to producer.
DeviceAccessStatus_ReadWrite	Full access
DeviceAccessStatus_ReadOnly	Read-only access
DeviceAccessStatus_NoAccess	Not available to connect
DeviceAccessStatus_Busy	The device is already opened by another entity
DeviceAccessStatus_OpenReadWrite	Open in Read/Write mode by this GenTL host
DeviceAccessStatus_OpenReadOnly	Open in Read access mode by this GenTL host
NUMDEVICEACCESSSTATUS	

13.18.1.2 spinTLDeviceCurrentSpeedEnums

```
enum spinTLDeviceCurrentSpeedEnums
```

< The USB Speed that the device is currently operating at.

Enumerator

DeviceCurrentSpeed_UnknownSpeed	Unknown-Speed.
DeviceCurrentSpeed_LowSpeed	Low-Speed.
DeviceCurrentSpeed_FullSpeed	Full-Speed.
DeviceCurrentSpeed_HighSpeed	High-Speed.
DeviceCurrentSpeed_SuperSpeed	Super-Speed.
NUMDEVICECURRENTSPEED	

13.18.1.3 spinTLDeviceEndianessMechanismEnums

```
enum spinTLDeviceEndianessMechanismEnums
```

< Identifies the endianness handling mode.

Enumerator

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	

13.18.1.4 spinTLDeviceTypeEnums

enum `spinTLDeviceTypeEnums`

< Transport layer type of the device.

Enumerator

DeviceType_GigEVision	GigE Vision
DeviceType_CameraLink	Camera Link
DeviceType_CameraLinkHS	Camera Link High Speed
DeviceType_CoaXPress	CoaXPress
DeviceType_USB3Vision	USB3 Vision
DeviceType_Custom	Custom transport layer
NUMDEVICETYPE	

13.18.1.5 spinTLFilterDriverStatusEnums

enum `spinTLFilterDriverStatusEnums`

< Reports whether FLIR Light Weight Filter Driver is enabled, disabled, or not installed.

Enumerator

FilterDriverStatus_NotSupported	Not Installed
FilterDriverStatus_Disabled	FLIR Light Weight Filter Driver is disabled across all interfaces
FilterDriverStatus_Enabled	FLIR Light Weight Filter Driver is enabled
NUMFILTERDRIVERSTATUS	

13.18.1.6 spinTLGenICamXMLLocationEnums

enum `spinTLGenICamXMLLocationEnums`

< Sets the location to load GenICam XML.

Enumerator

GenICamXMLLocation_Device	Load GenICam XML from device
GenICamXMLLocation_Host	Load GenICam XML from host
NUMGENICAMXMLLOCATION	

13.18.1.7 spinTLGevCCPEnums

enum [spinTLGevCCPEnums](#)

< Controls the device access privilege of an application.

Enumerator

GevCCP_EnumEntry_GevCCP_OpenAccess	Open access privilege.
GevCCP_EnumEntry_GevCCP_ExclusiveAccess	Exclusive access privilege.
GevCCP_EnumEntry_GevCCP_ControlAccess	Control access privilege.
NUMGEVCCP	

13.18.1.8 spinTLGUIXMLLocationEnums

enum [spinTLGUIXMLLocationEnums](#)

< Sets the location to load GUI XML.

Enumerator

GUIXMLLocation_Device	Load XML from device
GUIXMLLocation_Host	Load XML from host
NUMGUIXMLLOCATION	

13.18.1.9 spinTLInterfaceTypeEnums

enum [spinTLInterfaceTypeEnums](#)

< Transport layer type of the interface.

Enumerator

InterfaceType_GigEVision	GigE Vision
InterfaceType_CameraLink	Camera Link

Enumerator

InterfaceType_CameraLinkHS	Camera Link High Speed
InterfaceType_CoaXPress	CoaXPress
InterfaceType_USB3Vision	USB3 Vision
InterfaceType_Custom	Custom transport layer
NUMINTERFACETYPE	

13.18.1.10 spinTLPOEStatusEnums

enum [spinTLPOEStatusEnums](#)

< Reports and controls the interface's power over Ethernet status.

Enumerator

POEStatus_NotSupported	Not Supported
POEStatus_PowerOff	Power is Off
POEStatus_PowerOn	Power is On
NUMPOESTATUS	

13.18.1.11 spinTLStreamBufferCountModeEnums

enum [spinTLStreamBufferCountModeEnums](#)

< Controls access to setting the number of buffers used for the stream.

Enumerator

StreamBufferCountMode_Manual	The number of buffers used for the stream is set by the user.
NUMSTREAMBUFFERCOUNTMODE	

13.18.1.12 spinTLStreamBufferHandlingModeEnums

enum [spinTLStreamBufferHandlingModeEnums](#)

< Available buffer handling modes of this data stream:

Enumerator

StreamBufferHandlingMode_OldestFirst	The application always gets the buffer from the head of the output buffer queue (thus, the oldest available one). If the output buffer queue is empty, the application waits for a newly acquired buffer until the timeout expires.
StreamBufferHandlingMode_OldestFirstOverwrite	The application always gets the buffer from the head of the output buffer queue (thus, the oldest available one). If the output buffer queue is empty, the application waits for a newly acquired buffer until the timeout expires. If a new buffer arrives it will overwrite the existing buffer from the head of the queue (behaves like a circular buffer).
StreamBufferHandlingMode_NewestOnly	The application always gets the latest completed buffer (the newest one). If the Output Buffer Queue is empty, the application waits for a newly acquired buffer until the timeout expires. This buffer handling mode is typically used in a live display GUI where it is important that there is no lag between camera and display.
StreamBufferHandlingMode_NewestFirst	The application always gets the buffer from the tail of the output buffer queue (thus, the newest available one). If the output buffer queue is empty, the application waits for a newly acquired buffer until the timeout expires.
NUMSTREAMBUFFERHANDLINGMODE	

13.18.1.13 spinTLStreamModeEnums

enum [spinTLStreamModeEnums](#)

< Stream mode of the device.

Enumerator

StreamMode_Socket	Socket
StreamMode_LWF	Light Weight Filter Driver
StreamMode_MVA	Machine Vision Accelerator Driver
NUMSTREAMMODE	

13.18.1.14 spinTLStreamTypeEnums

enum [spinTLStreamTypeEnums](#)

The enumeration definitions for transport layer nodes.

< Stream type of the device.

Enumerator

StreamType_GigEVision	GigE Vision
StreamType_CameraLink	Camera Link
StreamType_CameraLinkHS	Camera Link High Speed
StreamType_CoaXPRESS	CoaXPRESS
StreamType_USB3Vision	USB3 Vision
StreamType_Custom	Custom transport layer
NUMSTREAMTYPE	

13.18.1.15 spinTLTLTypeEnums

enum `spinTLTLTypeEnums`

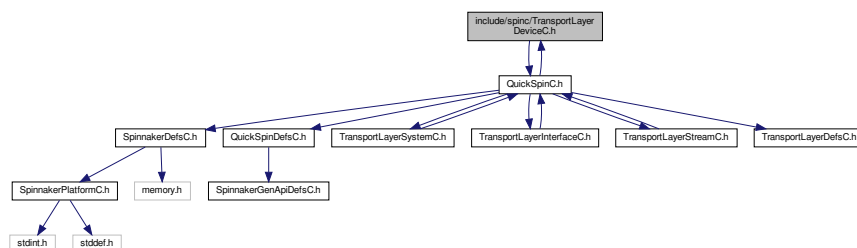
< Transport layer type of the GenTL Producer implementation.

Enumerator

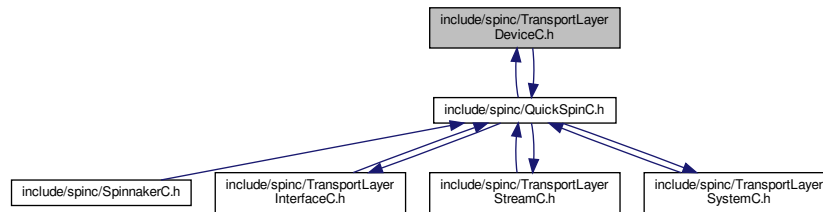
TLType_GigEVision	GigE Vision
TLType_CameraLink	Camera Link
TLType_CameraLinkHS	Camera Link High Speed
TLType_CoaXPRESS	CoaXPRESS
TLType_USB3Vision	USB3 Vision
TLType_Mixed	Different Interface modules of the GenTL Producer are of different types
TLType_Custom	Custom transport layer
NUMTLTYPE	

13.19 include/spinc/TransportLayerDeviceC.h File Reference

Include dependency graph for TransportLayerDeviceC.h:



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

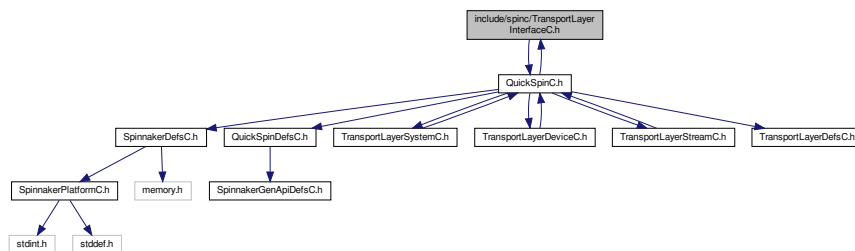


Data Structures

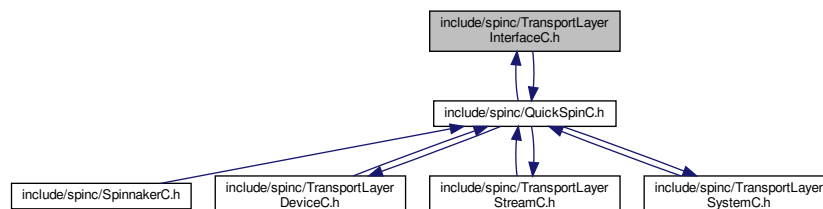
- struct [quickSpinTLDevice](#)

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:

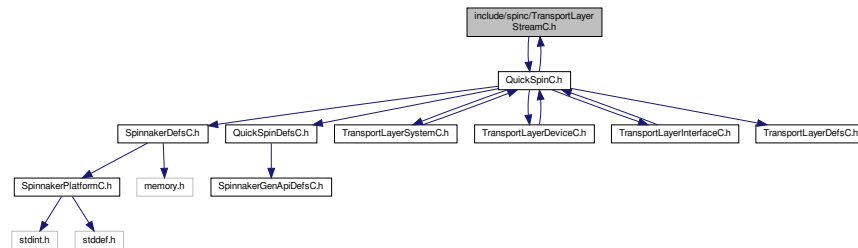


Data Structures

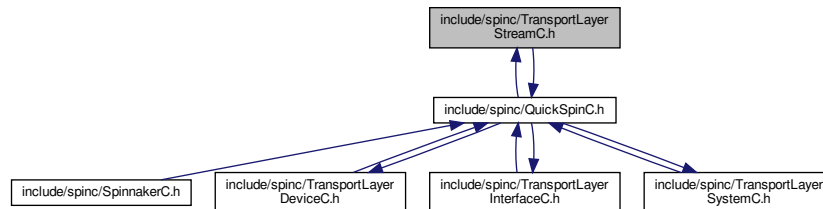
- struct [quickSpinTLInterface](#)

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:

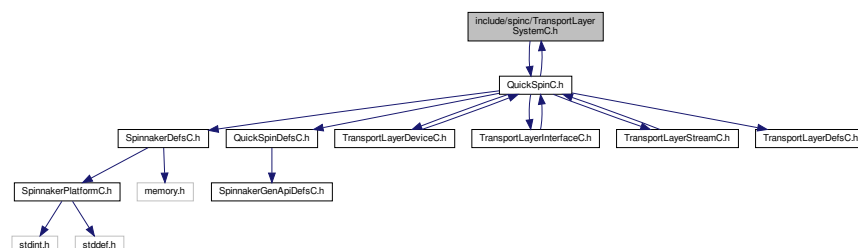


Data Structures

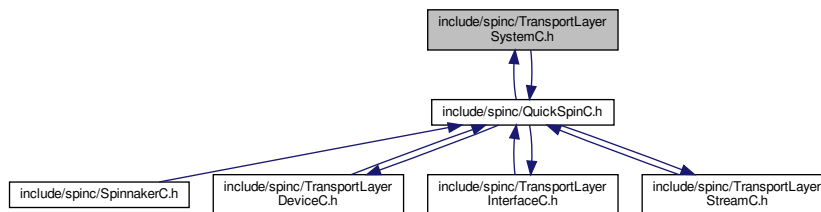
- struct [quickSpinTLStream](#)

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

- [_CycleDetectAccesMode](#)
SpinnakerGenApiDefsC.h, [464](#)
 - [_UndefinedAccesMode](#)
SpinnakerGenApiDefsC.h, [464](#)
 - [_UndefinedCachingMode](#)
SpinnakerGenApiDefsC.h, [464](#)
 - [_UndefinedEDisplayNotation](#)
SpinnakerGenApiDefsC.h, [464](#)
 - [_UndefinedESlope](#)
SpinnakerGenApiDefsC.h, [469](#)
 - [_UndefinedEXMLValidation](#)
SpinnakerGenApiDefsC.h, [470](#)
 - [_UndefinedEndian](#)
SpinnakerGenApiDefsC.h, [465](#)
 - [_UndefinedNameSpace](#)
SpinnakerGenApiDefsC.h, [467](#)
 - [_UndefinedRepresentation](#)
SpinnakerGenApiDefsC.h, [468](#)
 - [_UndefinedSign](#)
SpinnakerGenApiDefsC.h, [468](#)
 - [_UndefinedStandardNameSpace](#)
SpinnakerGenApiDefsC.h, [469](#)
 - [_UndefinedVisibility](#)
SpinnakerGenApiDefsC.h, [469](#)
 - [_UndefinedYesNo](#)
SpinnakerGenApiDefsC.h, [470](#)
- [AasRoiEnable](#)
quickSpin, [48](#)
- [AasRoiHeight](#)
quickSpin, [48](#)
- [AasRoiOffsetX](#)
quickSpin, [48](#)
- [AasRoiOffsetY](#)
quickSpin, [48](#)
- [AasRoiWidth](#)
quickSpin, [48](#)
- [AcquisitionAbort](#)
quickSpin, [49](#)
- [AcquisitionArm](#)
quickSpin, [49](#)
- [AcquisitionBurstFrameCount](#)
quickSpin, [49](#)
- [AcquisitionFrameCount](#)
quickSpin, [49](#)
- [AcquisitionFrameRate](#)
quickSpin, [49](#)
- [AcquisitionFrameRateEnable](#)
quickSpin, [49](#)
- [AcquisitionLineRate](#)
quickSpin, [49](#)
- [AcquisitionMode](#)
quickSpin, [49](#)
- [AcquisitionMode_Continuous](#)
CameraDefsC.h, [208](#)
- [AcquisitionMode_MultiFrame](#)
CameraDefsC.h, [208](#)
- [AcquisitionMode_SingleFrame](#)
CameraDefsC.h, [208](#)
- [AcquisitionResultingFrameRate](#)
quickSpin, [50](#)
- [AcquisitionStart](#)
quickSpin, [50](#)
- [AcquisitionStatus](#)
quickSpin, [50](#)
- [AcquisitionStatusSelector](#)
quickSpin, [50](#)
- [AcquisitionStatusSelector_AcquisitionActive](#)
CameraDefsC.h, [208](#)
- [AcquisitionStatusSelector_AcquisitionTransfer](#)
CameraDefsC.h, [208](#)
- [AcquisitionStatusSelector_AcquisitionTriggerWait](#)
CameraDefsC.h, [208](#)
- [AcquisitionStatusSelector_ExposureActive](#)
CameraDefsC.h, [208](#)
- [AcquisitionStatusSelector_FrameActive](#)
CameraDefsC.h, [208](#)
- [AcquisitionStatusSelector_FrameTriggerWait](#)
CameraDefsC.h, [208](#)
- [AcquisitionStop](#)
quickSpin, [50](#)
- [ActionCommand](#)
quickSpinTLInterface, [140](#)
- [actionCommandResult](#), [35](#)
DeviceAddress, [35](#)
Status, [35](#)
- [ActionDeviceKey](#)
quickSpin, [50](#)
- [ActionGroupKey](#)
quickSpin, [50](#)
- [ActionGroupMask](#)
quickSpin, [50](#)
- [ActionQueueSize](#)
quickSpin, [51](#)
- [ActionSelector](#)
quickSpin, [51](#)
- [ActionUnconditionalMode](#)
quickSpin, [51](#)
- [ActionUnconditionalMode_Off](#)

- CameraDefsC.h, [209](#)
- ActionUnconditionalMode_On
 - CameraDefsC.h, [209](#)
- AdaptiveCompressionEnable
 - quickSpin, [51](#)
- AdcBitDepth
 - quickSpin, [51](#)
- AdcBitDepth_Bit10
 - CameraDefsC.h, [209](#)
- AdcBitDepth_Bit12
 - CameraDefsC.h, [209](#)
- AdcBitDepth_Bit14
 - CameraDefsC.h, [209](#)
- AdcBitDepth_Bit8
 - CameraDefsC.h, [209](#)
- aPAUSEMACCtrlFramesReceived
 - quickSpin, [51](#)
- aPAUSEMACCtrlFramesTransmitted
 - quickSpin, [51](#)
- AutoAlgorithmSelector
 - quickSpin, [51](#)
- AutoAlgorithmSelector_Ae
 - CameraDefsC.h, [209](#)
- AutoAlgorithmSelector_Awb
 - CameraDefsC.h, [209](#)
- AutoExposureControlLoopDamping
 - quickSpin, [52](#)
- AutoExposureControlPriority
 - quickSpin, [52](#)
- AutoExposureControlPriority_ExposureTime
 - CameraDefsC.h, [210](#)
- AutoExposureControlPriority_Gain
 - CameraDefsC.h, [210](#)
- AutoExposureEVCompensation
 - quickSpin, [52](#)
- AutoExposureExposureTimeLowerLimit
 - quickSpin, [52](#)
- AutoExposureExposureTimeUpperLimit
 - quickSpin, [52](#)
- AutoExposureGainLowerLimit
 - quickSpin, [52](#)
- AutoExposureGainUpperLimit
 - quickSpin, [52](#)
- AutoExposureGreyValueLowerLimit
 - quickSpin, [52](#)
- AutoExposureGreyValueUpperLimit
 - quickSpin, [53](#)
- AutoExposureLightingMode
 - quickSpin, [53](#)
- AutoExposureLightingMode_AutoDetect
 - CameraDefsC.h, [210](#)
- AutoExposureLightingMode_Backlight
 - CameraDefsC.h, [210](#)
- AutoExposureLightingMode_Frontlight
 - CameraDefsC.h, [210](#)
- AutoExposureLightingMode_Normal
 - CameraDefsC.h, [210](#)
- AutoExposureMeteringMode
 - quickSpin, [53](#)
- AutoExposureMeteringMode_Average
 - CameraDefsC.h, [211](#)
- AutoExposureMeteringMode_CenterWeighted
 - CameraDefsC.h, [211](#)
- AutoExposureMeteringMode_HistogramPeak
 - CameraDefsC.h, [211](#)
- AutoExposureMeteringMode_Partial
 - CameraDefsC.h, [211](#)
- AutoExposureMeteringMode_Spot
 - CameraDefsC.h, [211](#)
- AutoExposureTargetGreyValue
 - quickSpin, [53](#)
- AutoExposureTargetGreyValueAuto
 - quickSpin, [53](#)
- AutoExposureTargetGreyValueAuto_Continuous
 - CameraDefsC.h, [211](#)
- AutoExposureTargetGreyValueAuto_Off
 - CameraDefsC.h, [211](#)
- Automatic
 - SpinnakerGenApiDefsC.h, [469](#)
- BalanceRatio
 - quickSpin, [53](#)
- BalanceRatioSelector
 - quickSpin, [53](#)
- BalanceRatioSelector_Blue
 - CameraDefsC.h, [211](#)
- BalanceRatioSelector_Red
 - CameraDefsC.h, [211](#)
- BalanceWhiteAuto
 - quickSpin, [53](#)
- BalanceWhiteAuto_Continuous
 - CameraDefsC.h, [212](#)
- BalanceWhiteAuto_Off
 - CameraDefsC.h, [212](#)
- BalanceWhiteAuto_Once
 - CameraDefsC.h, [212](#)
- BalanceWhiteAutoDamping
 - quickSpin, [54](#)
- BalanceWhiteAutoLowerLimit
 - quickSpin, [54](#)
- BalanceWhiteAutoProfile
 - quickSpin, [54](#)
- BalanceWhiteAutoProfile_Indoor
 - CameraDefsC.h, [212](#)
- BalanceWhiteAutoProfile_Outdoor
 - CameraDefsC.h, [212](#)
- BalanceWhiteAutoUpperLimit
 - quickSpin, [54](#)
- BaseNode
 - SpinnakerGenApiDefsC.h, [467](#)
- Beginner
 - SpinnakerGenApiDefsC.h, [469](#)
- BigEndian
 - SpinnakerGenApiDefsC.h, [465](#)
- binaryFile
 - spinPGMOption, [170](#)
 - spinPPMOption, [172](#)

- BinningHorizontal
 - quickSpin, [54](#)
- BinningHorizontalMode
 - quickSpin, [54](#)
- BinningHorizontalMode_Average
 - CameraDefsC.h, [212](#)
- BinningHorizontalMode_Sum
 - CameraDefsC.h, [212](#)
- BinningSelector
 - quickSpin, [54](#)
- BinningSelector_All
 - CameraDefsC.h, [213](#)
- BinningSelector_ISP
 - CameraDefsC.h, [213](#)
- BinningSelector_Sensor
 - CameraDefsC.h, [213](#)
- BinningVertical
 - quickSpin, [54](#)
- BinningVerticalMode
 - quickSpin, [55](#)
- BinningVerticalMode_Average
 - CameraDefsC.h, [213](#)
- BinningVerticalMode_Sum
 - CameraDefsC.h, [213](#)
- bitrate
 - spinH264Option, [164](#)
- BlackLevel
 - quickSpin, [55](#)
- BlackLevelAuto
 - quickSpin, [55](#)
- BlackLevelAuto_Continuous
 - CameraDefsC.h, [214](#)
- BlackLevelAuto_Off
 - CameraDefsC.h, [214](#)
- BlackLevelAuto_Once
 - CameraDefsC.h, [214](#)
- BlackLevelAutoBalance
 - quickSpin, [55](#)
- BlackLevelAutoBalance_Continuous
 - CameraDefsC.h, [213](#)
- BlackLevelAutoBalance_Off
 - CameraDefsC.h, [213](#)
- BlackLevelAutoBalance_Once
 - CameraDefsC.h, [213](#)
- BlackLevelClampingEnable
 - quickSpin, [55](#)
- BlackLevelRaw
 - quickSpin, [55](#)
- BlackLevelSelector
 - quickSpin, [55](#)
- BlackLevelSelector_All
 - CameraDefsC.h, [214](#)
- BlackLevelSelector_Analog
 - CameraDefsC.h, [214](#)
- BlackLevelSelector_Digital
 - CameraDefsC.h, [214](#)
- bool8_t
 - SpinnakerDefsC.h, [408](#)
- Boolean
 - SpinnakerGenApiDefsC.h, [468](#)
- BooleanNode
 - SpinnakerGenApiDefsC.h, [467](#)
- build
 - spinLibraryVersion, [168](#)
- Camera Access, [24](#)
- Camera Enumerations, [22](#)
- CameraDefsC.h
 - AcquisitionMode_Continuous, [208](#)
 - AcquisitionMode_MultiFrame, [208](#)
 - AcquisitionMode_SingleFrame, [208](#)
 - AcquisitionStatusSelector_AcquisitionActive, [208](#)
 - AcquisitionStatusSelector_AcquisitionTransfer, [208](#)
 - AcquisitionStatusSelector_AcquisitionTriggerWait, [208](#)
 - AcquisitionStatusSelector_ExposureActive, [208](#)
 - AcquisitionStatusSelector_FrameActive, [208](#)
 - AcquisitionStatusSelector_FrameTriggerWait, [208](#)
 - ActionUnconditionalMode_Off, [209](#)
 - ActionUnconditionalMode_On, [209](#)
 - AdcBitDepth_Bit10, [209](#)
 - AdcBitDepth_Bit12, [209](#)
 - AdcBitDepth_Bit14, [209](#)
 - AdcBitDepth_Bit8, [209](#)
 - AutoAlgorithmSelector_Ae, [209](#)
 - AutoAlgorithmSelector_Awb, [209](#)
 - AutoExposureControlPriority_ExposureTime, [210](#)
 - AutoExposureControlPriority_Gain, [210](#)
 - AutoExposureLightingMode_AutoDetect, [210](#)
 - AutoExposureLightingMode_Backlight, [210](#)
 - AutoExposureLightingMode_Frontlight, [210](#)
 - AutoExposureLightingMode_Normal, [210](#)
 - AutoExposureMeteringMode_Average, [211](#)
 - AutoExposureMeteringMode_CenterWeighted, [211](#)
 - AutoExposureMeteringMode_HistogramPeak, [211](#)
 - AutoExposureMeteringMode_Partial, [211](#)
 - AutoExposureMeteringMode_Spot, [211](#)
 - AutoExposureTargetGreyValueAuto_Continuous, [211](#)
 - AutoExposureTargetGreyValueAuto_Off, [211](#)
 - BalanceRatioSelector_Blue, [211](#)
 - BalanceRatioSelector_Red, [211](#)
 - BalanceWhiteAuto_Continuous, [212](#)
 - BalanceWhiteAuto_Off, [212](#)
 - BalanceWhiteAuto_Once, [212](#)
 - BalanceWhiteAutoProfile_Indoor, [212](#)
 - BalanceWhiteAutoProfile_Outdoor, [212](#)
 - BinningHorizontalMode_Average, [212](#)
 - BinningHorizontalMode_Sum, [212](#)
 - BinningSelector_All, [213](#)
 - BinningSelector_ISP, [213](#)
 - BinningSelector_Sensor, [213](#)
 - BinningVerticalMode_Average, [213](#)
 - BinningVerticalMode_Sum, [213](#)
 - BlackLevelAuto_Continuous, [214](#)
 - BlackLevelAuto_Off, [214](#)

- BlackLevelAuto_Once, [214](#)
- BlackLevelAutoBalance_Continuous, [213](#)
- BlackLevelAutoBalance_Off, [213](#)
- BlackLevelAutoBalance_Once, [213](#)
- BlackLevelSelector_All, [214](#)
- BlackLevelSelector_Analog, [214](#)
- BlackLevelSelector_Digital, [214](#)
- ChunkBlackLevelSelector_All, [214](#)
- ChunkCounterSelector_Counter0, [215](#)
- ChunkCounterSelector_Counter1, [215](#)
- ChunkCounterSelector_Counter2, [215](#)
- ChunkEncoderSelector_Encoder0, [215](#)
- ChunkEncoderSelector_Encoder1, [215](#)
- ChunkEncoderSelector_Encoder2, [215](#)
- ChunkEncoderStatus_EncoderDown, [215](#)
- ChunkEncoderStatus_EncoderIdle, [215](#)
- ChunkEncoderStatus_EncoderStatic, [215](#)
- ChunkEncoderStatus_EncoderUp, [215](#)
- ChunkExposureTimeSelector_Blue, [216](#)
- ChunkExposureTimeSelector_Common, [216](#)
- ChunkExposureTimeSelector_Cyan, [216](#)
- ChunkExposureTimeSelector_Green, [216](#)
- ChunkExposureTimeSelector_Infrared, [216](#)
- ChunkExposureTimeSelector_Magenta, [216](#)
- ChunkExposureTimeSelector_Red, [216](#)
- ChunkExposureTimeSelector_Stage1, [216](#)
- ChunkExposureTimeSelector_Stage2, [216](#)
- ChunkExposureTimeSelector_Ultraviolet, [216](#)
- ChunkExposureTimeSelector_Yellow, [216](#)
- ChunkGainSelector_All, [216](#)
- ChunkGainSelector_Blue, [216](#)
- ChunkGainSelector_Green, [216](#)
- ChunkGainSelector_Red, [216](#)
- ChunkImageComponent_Color, [216](#)
- ChunkImageComponent_Confidence, [217](#)
- ChunkImageComponent_Disparity, [216](#)
- ChunkImageComponent_Infrared, [216](#)
- ChunkImageComponent_Intensity, [216](#)
- ChunkImageComponent_Range, [216](#)
- ChunkImageComponent_Scatter, [217](#)
- ChunkImageComponent_Ultraviolet, [216](#)
- ChunkPixelFormat_BayerBG8, [217](#)
- ChunkPixelFormat_BayerGB8, [217](#)
- ChunkPixelFormat_BayerGR8, [217](#)
- ChunkPixelFormat_BayerRG8, [217](#)
- ChunkPixelFormat_Mono12Packed, [217](#)
- ChunkPixelFormat_Mono16, [217](#)
- ChunkPixelFormat_Mono8, [217](#)
- ChunkPixelFormat_RGB8Packed, [217](#)
- ChunkPixelFormat_YCbCr601_422_8_CbYCrY, [217](#)
- ChunkPixelFormat_YUV422Packed, [217](#)
- ChunkRegionID_Region0, [217](#)
- ChunkRegionID_Region1, [217](#)
- ChunkRegionID_Region2, [217](#)
- ChunkScan3dCoordinateReferenceSelector_RotationX, [218](#)
- ChunkScan3dCoordinateReferenceSelector_RotationY, [218](#)
- ChunkScan3dCoordinateReferenceSelector_RotationZ, [218](#)
- ChunkScan3dCoordinateReferenceSelector_TranslationX, [218](#)
- ChunkScan3dCoordinateReferenceSelector_TranslationY, [218](#)
- ChunkScan3dCoordinateReferenceSelector_TranslationZ, [218](#)
- ChunkScan3dCoordinateSelector_CoordinateA, [218](#)
- ChunkScan3dCoordinateSelector_CoordinateB, [218](#)
- ChunkScan3dCoordinateSelector_CoordinateC, [218](#)
- ChunkScan3dCoordinateSystem_Cartesian, [218](#)
- ChunkScan3dCoordinateSystem_Cylindrical, [218](#)
- ChunkScan3dCoordinateSystem_Spherical, [218](#)
- ChunkScan3dCoordinateSystemReference_Anchor, [219](#)
- ChunkScan3dCoordinateSystemReference_Transformed, [219](#)
- ChunkScan3dCoordinateTransformSelector_RotationX, [219](#)
- ChunkScan3dCoordinateTransformSelector_RotationY, [219](#)
- ChunkScan3dCoordinateTransformSelector_RotationZ, [219](#)
- ChunkScan3dCoordinateTransformSelector_TranslationX, [219](#)
- ChunkScan3dCoordinateTransformSelector_TranslationY, [219](#)
- ChunkScan3dCoordinateTransformSelector_TranslationZ, [219](#)
- ChunkScan3dDistanceUnit_Inch, [219](#)
- ChunkScan3dDistanceUnit_Millimeter, [219](#)
- ChunkScan3dOutputMode_CalibratedABC_Grid, [220](#)
- ChunkScan3dOutputMode_CalibratedABC_PointCloud, [220](#)
- ChunkScan3dOutputMode_CalibratedAC, [220](#)
- ChunkScan3dOutputMode_CalibratedAC_Linescan, [220](#)
- ChunkScan3dOutputMode_CalibratedC, [220](#)
- ChunkScan3dOutputMode_CalibratedC_Linescan, [220](#)
- ChunkScan3dOutputMode_DisparityC, [220](#)
- ChunkScan3dOutputMode_DisparityC_Linescan, [220](#)
- ChunkScan3dOutputMode_RectifiedC, [220](#)
- ChunkScan3dOutputMode_RectifiedC_Linescan, [220](#)
- ChunkScan3dOutputMode_UncalibratedC, [220](#)
- ChunkSelector_BlackLevel, [221](#)
- ChunkSelector_CRC, [221](#)
- ChunkSelector_ExposureEndLineStatusAll, [221](#)
- ChunkSelector_ExposureTime, [221](#)

ChunkSelector_FrameID, [221](#)
ChunkSelector_Gain, [221](#)
ChunkSelector_Height, [221](#)
ChunkSelector_Image, [221](#)
ChunkSelector_OffsetX, [221](#)
ChunkSelector_OffsetY, [221](#)
ChunkSelector_PixelFormat, [221](#)
ChunkSelector_SequencerSetActive, [221](#)
ChunkSelector_SerialData, [221](#)
ChunkSelector_Timestamp, [221](#)
ChunkSelector_Width, [221](#)
ChunkSourceID_Source0, [221](#)
ChunkSourceID_Source1, [221](#)
ChunkSourceID_Source2, [221](#)
ChunkTimerSelector_Timer0, [222](#)
ChunkTimerSelector_Timer1, [222](#)
ChunkTimerSelector_Timer2, [222](#)
ChunkTransferStreamID_Stream0, [222](#)
ChunkTransferStreamID_Stream1, [222](#)
ChunkTransferStreamID_Stream2, [222](#)
ChunkTransferStreamID_Stream3, [222](#)
CIConfiguration_Base, [222](#)
CIConfiguration_DualBase, [222](#)
CIConfiguration_EightyBit, [222](#)
CIConfiguration_Full, [222](#)
CIConfiguration_Medium, [222](#)
CITimeSlotsCount_One, [223](#)
CITimeSlotsCount_Three, [223](#)
CITimeSlotsCount_Two, [223](#)
ColorTransformationSelector_RGBtoRGB, [223](#)
ColorTransformationSelector_RGBtoYUV, [223](#)
ColorTransformationValueSelector_Gain00, [223](#)
ColorTransformationValueSelector_Gain01, [223](#)
ColorTransformationValueSelector_Gain02, [223](#)
ColorTransformationValueSelector_Gain10, [223](#)
ColorTransformationValueSelector_Gain11, [223](#)
ColorTransformationValueSelector_Gain12, [223](#)
ColorTransformationValueSelector_Gain20, [223](#)
ColorTransformationValueSelector_Gain21, [223](#)
ColorTransformationValueSelector_Gain22, [223](#)
ColorTransformationValueSelector_Offset0, [223](#)
ColorTransformationValueSelector_Offset1, [223](#)
ColorTransformationValueSelector_Offset2, [223](#)
CompressionSaturationPriority_DropFrame, [224](#)
CompressionSaturationPriority_ReduceFrameRate, [224](#)
CounterEventActivation_AnyEdge, [224](#)
CounterEventActivation_FallingEdge, [224](#)
CounterEventActivation_LevelHigh, [224](#)
CounterEventActivation_LevelLow, [224](#)
CounterEventActivation_RisingEdge, [224](#)
CounterEventSource_Counter0End, [225](#)
CounterEventSource_Counter0Start, [225](#)
CounterEventSource_Counter1End, [225](#)
CounterEventSource_Counter1Start, [225](#)
CounterEventSource_ExposureEnd, [225](#)
CounterEventSource_ExposureStart, [225](#)
CounterEventSource_FrameTriggerWait, [225](#)
CounterEventSource_Line0, [224](#)
CounterEventSource_Line1, [225](#)
CounterEventSource_Line2, [225](#)
CounterEventSource_Line3, [225](#)
CounterEventSource_LogicBlock0, [225](#)
CounterEventSource_LogicBlock1, [225](#)
CounterEventSource_MHzTick, [224](#)
CounterEventSource_Off, [224](#)
CounterEventSource_UserOutput0, [225](#)
CounterEventSource_UserOutput1, [225](#)
CounterEventSource_UserOutput2, [225](#)
CounterEventSource_UserOutput3, [225](#)
CounterResetActivation_AnyEdge, [225](#)
CounterResetActivation_FallingEdge, [225](#)
CounterResetActivation_LevelHigh, [225](#)
CounterResetActivation_LevelLow, [225](#)
CounterResetActivation_RisingEdge, [225](#)
CounterResetSource_Counter0End, [226](#)
CounterResetSource_Counter0Start, [226](#)
CounterResetSource_Counter1End, [226](#)
CounterResetSource_Counter1Start, [226](#)
CounterResetSource_ExposureEnd, [226](#)
CounterResetSource_ExposureStart, [226](#)
CounterResetSource_FrameTriggerWait, [226](#)
CounterResetSource_Line0, [226](#)
CounterResetSource_Line1, [226](#)
CounterResetSource_Line2, [226](#)
CounterResetSource_Line3, [226](#)
CounterResetSource_LogicBlock0, [226](#)
CounterResetSource_LogicBlock1, [226](#)
CounterResetSource_Off, [225](#)
CounterResetSource_UserOutput0, [226](#)
CounterResetSource_UserOutput1, [226](#)
CounterResetSource_UserOutput2, [226](#)
CounterResetSource_UserOutput3, [226](#)
CounterSelector_Counter0, [226](#)
CounterSelector_Counter1, [226](#)
CounterStatus_CounterActive, [226](#)
CounterStatus_CounterCompleted, [226](#)
CounterStatus_CounterIdle, [226](#)
CounterStatus_CounterOverflow, [226](#)
CounterStatus_CounterTriggerWait, [226](#)
CounterTriggerActivation_AnyEdge, [227](#)
CounterTriggerActivation_FallingEdge, [227](#)
CounterTriggerActivation_LevelHigh, [227](#)
CounterTriggerActivation_LevelLow, [227](#)
CounterTriggerActivation_RisingEdge, [227](#)
CounterTriggerSource_Counter0End, [227](#)
CounterTriggerSource_Counter0Start, [227](#)
CounterTriggerSource_Counter1End, [227](#)
CounterTriggerSource_Counter1Start, [227](#)
CounterTriggerSource_ExposureEnd, [227](#)
CounterTriggerSource_ExposureStart, [227](#)
CounterTriggerSource_FrameTriggerWait, [227](#)
CounterTriggerSource_Line0, [227](#)
CounterTriggerSource_Line1, [227](#)
CounterTriggerSource_Line2, [227](#)
CounterTriggerSource_Line3, [227](#)

CounterTriggerSource_LogicBlock0, [227](#)
 CounterTriggerSource_LogicBlock1, [227](#)
 CounterTriggerSource_Off, [227](#)
 CounterTriggerSource_UserOutput0, [227](#)
 CounterTriggerSource_UserOutput1, [227](#)
 CounterTriggerSource_UserOutput2, [227](#)
 CounterTriggerSource_UserOutput3, [227](#)
 CxpConnectionTestMode_Mode1, [228](#)
 CxpConnectionTestMode_Off, [228](#)
 CxpLinkConfiguration_Auto, [228](#)
 CxpLinkConfiguration_CXP1_X1, [228](#)
 CxpLinkConfiguration_CXP1_X2, [228](#)
 CxpLinkConfiguration_CXP1_X3, [228](#)
 CxpLinkConfiguration_CXP1_X4, [228](#)
 CxpLinkConfiguration_CXP1_X5, [229](#)
 CxpLinkConfiguration_CXP1_X6, [229](#)
 CxpLinkConfiguration_CXP2_X1, [228](#)
 CxpLinkConfiguration_CXP2_X2, [228](#)
 CxpLinkConfiguration_CXP2_X3, [228](#)
 CxpLinkConfiguration_CXP2_X4, [228](#)
 CxpLinkConfiguration_CXP2_X5, [229](#)
 CxpLinkConfiguration_CXP2_X6, [229](#)
 CxpLinkConfiguration_CXP3_X1, [228](#)
 CxpLinkConfiguration_CXP3_X2, [228](#)
 CxpLinkConfiguration_CXP3_X3, [228](#)
 CxpLinkConfiguration_CXP3_X4, [228](#)
 CxpLinkConfiguration_CXP3_X5, [229](#)
 CxpLinkConfiguration_CXP3_X6, [229](#)
 CxpLinkConfiguration_CXP5_X1, [228](#)
 CxpLinkConfiguration_CXP5_X2, [228](#)
 CxpLinkConfiguration_CXP5_X3, [228](#)
 CxpLinkConfiguration_CXP5_X4, [228](#)
 CxpLinkConfiguration_CXP5_X5, [229](#)
 CxpLinkConfiguration_CXP5_X6, [229](#)
 CxpLinkConfiguration_CXP6_X1, [228](#)
 CxpLinkConfiguration_CXP6_X2, [228](#)
 CxpLinkConfiguration_CXP6_X3, [228](#)
 CxpLinkConfiguration_CXP6_X4, [228](#)
 CxpLinkConfiguration_CXP6_X5, [229](#)
 CxpLinkConfiguration_CXP6_X6, [229](#)
 CxpLinkConfigurationPreferred_CXP1_X1, [229](#)
 CxpLinkConfigurationPreferred_CXP1_X2, [229](#)
 CxpLinkConfigurationPreferred_CXP1_X3, [229](#)
 CxpLinkConfigurationPreferred_CXP1_X4, [229](#)
 CxpLinkConfigurationPreferred_CXP1_X5, [229](#)
 CxpLinkConfigurationPreferred_CXP1_X6, [230](#)
 CxpLinkConfigurationPreferred_CXP2_X1, [229](#)
 CxpLinkConfigurationPreferred_CXP2_X2, [229](#)
 CxpLinkConfigurationPreferred_CXP2_X3, [229](#)
 CxpLinkConfigurationPreferred_CXP2_X4, [229](#)
 CxpLinkConfigurationPreferred_CXP2_X5, [229](#)
 CxpLinkConfigurationPreferred_CXP2_X6, [230](#)
 CxpLinkConfigurationPreferred_CXP3_X1, [229](#)
 CxpLinkConfigurationPreferred_CXP3_X2, [229](#)
 CxpLinkConfigurationPreferred_CXP3_X3, [229](#)
 CxpLinkConfigurationPreferred_CXP3_X4, [229](#)
 CxpLinkConfigurationPreferred_CXP3_X5, [229](#)
 CxpLinkConfigurationPreferred_CXP3_X6, [230](#)
 CxpLinkConfigurationPreferred_CXP5_X1, [229](#)
 CxpLinkConfigurationPreferred_CXP5_X2, [229](#)
 CxpLinkConfigurationPreferred_CXP5_X3, [229](#)
 CxpLinkConfigurationPreferred_CXP5_X4, [229](#)
 CxpLinkConfigurationPreferred_CXP5_X5, [229](#)
 CxpLinkConfigurationPreferred_CXP5_X6, [230](#)
 CxpLinkConfigurationPreferred_CXP6_X1, [229](#)
 CxpLinkConfigurationPreferred_CXP6_X2, [229](#)
 CxpLinkConfigurationPreferred_CXP6_X3, [229](#)
 CxpLinkConfigurationPreferred_CXP6_X4, [229](#)
 CxpLinkConfigurationPreferred_CXP6_X5, [230](#)
 CxpLinkConfigurationPreferred_CXP6_X6, [230](#)
 CxpLinkConfigurationStatus_CXP1_X1, [230](#)
 CxpLinkConfigurationStatus_CXP1_X2, [230](#)
 CxpLinkConfigurationStatus_CXP1_X3, [230](#)
 CxpLinkConfigurationStatus_CXP1_X4, [230](#)
 CxpLinkConfigurationStatus_CXP1_X5, [230](#)
 CxpLinkConfigurationStatus_CXP1_X6, [231](#)
 CxpLinkConfigurationStatus_CXP2_X1, [230](#)
 CxpLinkConfigurationStatus_CXP2_X2, [230](#)
 CxpLinkConfigurationStatus_CXP2_X3, [230](#)
 CxpLinkConfigurationStatus_CXP2_X4, [230](#)
 CxpLinkConfigurationStatus_CXP2_X5, [230](#)
 CxpLinkConfigurationStatus_CXP2_X6, [231](#)
 CxpLinkConfigurationStatus_CXP3_X1, [230](#)
 CxpLinkConfigurationStatus_CXP3_X2, [230](#)
 CxpLinkConfigurationStatus_CXP3_X3, [230](#)
 CxpLinkConfigurationStatus_CXP3_X4, [230](#)
 CxpLinkConfigurationStatus_CXP3_X5, [230](#)
 CxpLinkConfigurationStatus_CXP3_X6, [231](#)
 CxpLinkConfigurationStatus_CXP5_X1, [230](#)
 CxpLinkConfigurationStatus_CXP5_X2, [230](#)
 CxpLinkConfigurationStatus_CXP5_X3, [230](#)
 CxpLinkConfigurationStatus_CXP5_X4, [230](#)
 CxpLinkConfigurationStatus_CXP5_X5, [230](#)
 CxpLinkConfigurationStatus_CXP5_X6, [231](#)
 CxpLinkConfigurationStatus_CXP6_X1, [230](#)
 CxpLinkConfigurationStatus_CXP6_X2, [230](#)
 CxpLinkConfigurationStatus_CXP6_X3, [230](#)
 CxpLinkConfigurationStatus_CXP6_X4, [230](#)
 CxpLinkConfigurationStatus_CXP6_X5, [231](#)
 CxpLinkConfigurationStatus_CXP6_X6, [231](#)
 CxpLinkConfigurationStatus_None, [230](#)
 CxpLinkConfigurationStatus_Pending, [230](#)
 CxpPoCxpStatus_Auto, [231](#)
 CxpPoCxpStatus_Off, [231](#)
 CxpPoCxpStatus_Tripped, [231](#)
 DecimationHorizontalMode_Discard, [231](#)
 DecimationSelector_All, [232](#)
 DecimationSelector_Sensor, [232](#)
 DecimationVerticalMode_Discard, [232](#)
 DefectCorrectionMode_Average, [232](#)
 DefectCorrectionMode_Highlight, [232](#)
 DefectCorrectionMode_Zero, [232](#)
 Deinterlacing_LineDuplication, [233](#)
 Deinterlacing_Off, [233](#)
 Deinterlacing_Weave, [233](#)
 DeviceCharacterSet_ASCII, [233](#)

- DeviceCharacterSet_UTF8, [233](#)
- DeviceClockSelector_CameraLink, [233](#)
- DeviceClockSelector_Sensor, [233](#)
- DeviceClockSelector_SensorDigitization, [233](#)
- DeviceConnectionStatus_Active, [233](#)
- DeviceConnectionStatus_Inactive, [233](#)
- DeviceIndicatorMode_Active, [234](#)
- DeviceIndicatorMode_ErrorStatus, [234](#)
- DeviceIndicatorMode_Inactive, [234](#)
- DeviceLinkHeartbeatMode_Off, [234](#)
- DeviceLinkHeartbeatMode_On, [234](#)
- DeviceLinkThroughputLimitMode_Off, [234](#)
- DeviceLinkThroughputLimitMode_On, [234](#)
- DevicePowerSupplySelector_External, [235](#)
- DeviceRegistersEndianness_Big, [235](#)
- DeviceRegistersEndianness_Little, [235](#)
- DeviceScanType_Areascan, [235](#)
- DeviceSerialPortBaudRate_Baud115200, [236](#)
- DeviceSerialPortBaudRate_Baud19200, [236](#)
- DeviceSerialPortBaudRate_Baud230400, [236](#)
- DeviceSerialPortBaudRate_Baud38400, [236](#)
- DeviceSerialPortBaudRate_Baud460800, [236](#)
- DeviceSerialPortBaudRate_Baud57600, [236](#)
- DeviceSerialPortBaudRate_Baud921600, [236](#)
- DeviceSerialPortBaudRate_Baud9600, [236](#)
- DeviceSerialPortSelector_CameraLink, [236](#)
- DeviceStreamChannelEndianness_Big, [236](#)
- DeviceStreamChannelEndianness_Little, [236](#)
- DeviceStreamChannelType_Receiver, [237](#)
- DeviceStreamChannelType_Transmitter, [237](#)
- DeviceTapGeometry_Geometry_10X, [238](#)
- DeviceTapGeometry_Geometry_10X_1Y, [238](#)
- DeviceTapGeometry_Geometry_1X, [237](#)
- DeviceTapGeometry_Geometry_1X10, [238](#)
- DeviceTapGeometry_Geometry_1X10_1Y, [238](#)
- DeviceTapGeometry_Geometry_1X2, [237](#)
- DeviceTapGeometry_Geometry_1X2_1Y, [237](#)
- DeviceTapGeometry_Geometry_1X2_1Y2, [237](#)
- DeviceTapGeometry_Geometry_1X2_2YE, [237](#)
- DeviceTapGeometry_Geometry_1X3, [237](#)
- DeviceTapGeometry_Geometry_1X3_1Y, [237](#)
- DeviceTapGeometry_Geometry_1X4, [237](#)
- DeviceTapGeometry_Geometry_1X4_1Y, [237](#)
- DeviceTapGeometry_Geometry_1X8, [238](#)
- DeviceTapGeometry_Geometry_1X8_1Y, [238](#)
- DeviceTapGeometry_Geometry_1X_1Y, [237](#)
- DeviceTapGeometry_Geometry_1X_1Y2, [237](#)
- DeviceTapGeometry_Geometry_1X_2YE, [237](#)
- DeviceTapGeometry_Geometry_2X, [237](#)
- DeviceTapGeometry_Geometry_2X2, [237](#)
- DeviceTapGeometry_Geometry_2X2_1Y, [237](#)
- DeviceTapGeometry_Geometry_2X2E, [237](#)
- DeviceTapGeometry_Geometry_2X2E_1YGeometry_2X2M_1Y, [237](#)
- DeviceTapGeometry_Geometry_2X2E_2YE, [238](#)
- DeviceTapGeometry_Geometry_2X2M, [238](#)
- DeviceTapGeometry_Geometry_2X_1Y, [237](#)
- DeviceTapGeometry_Geometry_2X_1Y2Geometry_2XE_1Y, [237](#)
- DeviceTapGeometry_Geometry_2X_2YE, [237](#)
- DeviceTapGeometry_Geometry_2XE, [237](#)
- DeviceTapGeometry_Geometry_2XE_1Y2, [237](#)
- DeviceTapGeometry_Geometry_2XE_2YE, [237](#)
- DeviceTapGeometry_Geometry_2XM, [237](#)
- DeviceTapGeometry_Geometry_2XM_1Y, [237](#)
- DeviceTapGeometry_Geometry_2XM_1Y2, [237](#)
- DeviceTapGeometry_Geometry_2XM_2YE, [237](#)
- DeviceTapGeometry_Geometry_3X, [237](#)
- DeviceTapGeometry_Geometry_3X_1Y, [237](#)
- DeviceTapGeometry_Geometry_4X, [237](#)
- DeviceTapGeometry_Geometry_4X2, [238](#)
- DeviceTapGeometry_Geometry_4X2_1Y, [238](#)
- DeviceTapGeometry_Geometry_4X2E, [238](#)
- DeviceTapGeometry_Geometry_4X2E_1Y, [238](#)
- DeviceTapGeometry_Geometry_4X_1Y, [237](#)
- DeviceTapGeometry_Geometry_8X, [238](#)
- DeviceTapGeometry_Geometry_8X_1Y, [238](#)
- DeviceTemperatureSelector_Sensor, [238](#)
- DeviceTLType_CameraLink, [238](#)
- DeviceTLType_CameraLinkHS, [238](#)
- DeviceTLType_CoaXPress, [238](#)
- DeviceTLType_Custom, [238](#)
- DeviceTLType_GigEVision, [238](#)
- DeviceTLType_USB3Vision, [238](#)
- DeviceType_Peripheral, [239](#)
- DeviceType_Receiver, [239](#)
- DeviceType_Transceiver, [239](#)
- DeviceType_Transmitter, [239](#)
- EncoderMode_FourPhase, [239](#)
- EncoderMode_HighResolution, [239](#)
- EncoderOutputMode_DirectionDown, [240](#)
- EncoderOutputMode_DirectionUp, [240](#)
- EncoderOutputMode_Motion, [240](#)
- EncoderOutputMode_Off, [239](#)
- EncoderOutputMode_PositionDown, [240](#)
- EncoderOutputMode_PositionUp, [239](#)
- EncoderResetActivation_AnyEdge, [240](#)
- EncoderResetActivation_FallingEdge, [240](#)
- EncoderResetActivation_LevelHigh, [240](#)
- EncoderResetActivation_LevelLow, [240](#)
- EncoderResetActivation_RisingEdge, [240](#)
- EncoderResetSource_AcquisitionEnd, [240](#)
- EncoderResetSource_AcquisitionStart, [240](#)
- EncoderResetSource_AcquisitionTrigger, [240](#)
- EncoderResetSource_Action0, [241](#)
- EncoderResetSource_Action1, [241](#)
- EncoderResetSource_Action2, [241](#)
- EncoderResetSource_Counter0End, [241](#)
- EncoderResetSource_Counter0Start, [241](#)
- EncoderResetSource_Counter1End, [241](#)
- EncoderResetSource_Counter1Start, [241](#)
- EncoderResetSource_Counter2End, [241](#)
- EncoderResetSource_Counter2Start, [241](#)
- EncoderResetSource_ExposureEnd, [241](#)
- EncoderResetSource_ExposureStart, [240](#)

- EncoderResetSource_FrameEnd, 240
- EncoderResetSource_FrameStart, 240
- EncoderResetSource_FrameTrigger, 240
- EncoderResetSource_Line0, 241
- EncoderResetSource_Line1, 241
- EncoderResetSource_Line2, 241
- EncoderResetSource_LinkTrigger0, 241
- EncoderResetSource_LinkTrigger1, 241
- EncoderResetSource_LinkTrigger2, 241
- EncoderResetSource_Off, 240
- EncoderResetSource_SoftwareSignal0, 241
- EncoderResetSource_SoftwareSignal1, 241
- EncoderResetSource_SoftwareSignal2, 241
- EncoderResetSource_Timer0End, 241
- EncoderResetSource_Timer0Start, 241
- EncoderResetSource_Timer1End, 241
- EncoderResetSource_Timer1Start, 241
- EncoderResetSource_Timer2End, 241
- EncoderResetSource_Timer2Start, 241
- EncoderResetSource_UserOutput0, 241
- EncoderResetSource_UserOutput1, 241
- EncoderResetSource_UserOutput2, 241
- EncoderSelector_Encoder0, 241
- EncoderSelector_Encoder1, 241
- EncoderSelector_Encoder2, 241
- EncoderSourceA_Line0, 242
- EncoderSourceA_Line1, 242
- EncoderSourceA_Line2, 242
- EncoderSourceA_Off, 242
- EncoderSourceB_Line0, 242
- EncoderSourceB_Line1, 242
- EncoderSourceB_Line2, 242
- EncoderSourceB_Off, 242
- EncoderStatus_EncoderDown, 242
- EncoderStatus_EncoderIdle, 242
- EncoderStatus_EncoderStatic, 242
- EncoderStatus_EncoderUp, 242
- EventNotification_Off, 243
- EventNotification_On, 243
- EventSelector_Error, 243
- EventSelector_ExposureEnd, 243
- EventSelector_SerialPortReceive, 243
- ExposureActiveMode_AllPixels, 243
- ExposureActiveMode_AnyPixels, 243
- ExposureActiveMode_Line1, 243
- ExposureAuto_Continuous, 244
- ExposureAuto_Off, 244
- ExposureAuto_Once, 244
- ExposureMode_Timed, 244
- ExposureMode_TriggerWidth, 244
- ExposureTimeMode_Common, 244
- ExposureTimeMode_Individual, 244
- ExposureTimeSelector_Blue, 245
- ExposureTimeSelector_Common, 245
- ExposureTimeSelector_Cyan, 245
- ExposureTimeSelector_Green, 245
- ExposureTimeSelector_Infrared, 245
- ExposureTimeSelector_Magenta, 245
- ExposureTimeSelector_Red, 245
- ExposureTimeSelector_Stage1, 245
- ExposureTimeSelector_Stage2, 245
- ExposureTimeSelector_Ultraviolet, 245
- ExposureTimeSelector_Yellow, 245
- FileOpenMode_Read, 245
- FileOpenMode_ReadWrite, 245
- FileOpenMode_Write, 245
- FileOperationSelector_Close, 246
- FileOperationSelector_Delete, 246
- FileOperationSelector_Open, 246
- FileOperationSelector_Read, 246
- FileOperationSelector_Write, 246
- FileOperationStatus_Failure, 246
- FileOperationStatus_Overflow, 246
- FileOperationStatus_Success, 246
- FileSelector_SerialPort0, 246
- FileSelector_UserFile1, 246
- FileSelector_UserSet0, 246
- FileSelector_UserSet1, 246
- FileSelector_UserSetDefault, 246
- GainAuto_Continuous, 248
- GainAuto_Off, 248
- GainAuto_Once, 248
- GainAutoBalance_Continuous, 248
- GainAutoBalance_Off, 248
- GainAutoBalance_Once, 248
- GainSelector_All, 248
- GevCCP_ControlAccess, 249
- GevCCP_ExclusiveAccess, 249
- GevCCP_OpenAccess, 249
- GevCurrentPhysicalLinkConfiguration_DynamicLAG, 249
- GevCurrentPhysicalLinkConfiguration_MultiLink, 249
- GevCurrentPhysicalLinkConfiguration_SingleLink, 249
- GevCurrentPhysicalLinkConfiguration_StaticLAG, 249
- GevGVCPExtendedStatusCodesSelector_Version1_1, 249
- GevGVCPExtendedStatusCodesSelector_Version2_0, 249
- GevGVSPExtendedIDMode_Off, 250
- GevGVSPExtendedIDMode_On, 250
- GevIEEE1588ClockAccuracy_Unknown, 250
- GevIEEE1588Mode_Auto, 250
- GevIEEE1588Mode_SlaveOnly, 250
- GevIEEE1588Status_Disabled, 250
- GevIEEE1588Status_Faulty, 250
- GevIEEE1588Status_Initializing, 250
- GevIEEE1588Status_Listening, 251
- GevIEEE1588Status_Master, 251
- GevIEEE1588Status_Passive, 251
- GevIEEE1588Status_PreMaster, 251
- GevIEEE1588Status_Slave, 251
- GevIEEE1588Status_Uncalibrated, 251
- GevIPConfigurationStatus_DHCP, 251

- GevIPConfigurationStatus_ForceIP, [251](#)
- GevIPConfigurationStatus_LLA, [251](#)
- GevIPConfigurationStatus_None, [251](#)
- GevIPConfigurationStatus_PersistentIP, [251](#)
- GevPhysicalLinkConfiguration_DynamicLAG, [251](#)
- GevPhysicalLinkConfiguration_MultiLink, [251](#)
- GevPhysicalLinkConfiguration_SingleLink, [251](#)
- GevPhysicalLinkConfiguration_StaticLAG, [251](#)
- GevSupportedOptionSelector_Action, [252](#)
- GevSupportedOptionSelector_CCPApplicationSocket, [252](#)
- GevSupportedOptionSelector_CommandsConcatenation, [252](#)
- GevSupportedOptionSelector_DiscoveryAckDelay, [252](#)
- GevSupportedOptionSelector_DiscoveryAckDelayWritable, [252](#)
- GevSupportedOptionSelector_Event, [252](#)
- GevSupportedOptionSelector_EventData, [252](#)
- GevSupportedOptionSelector_ExtendedStatusCodes, [252](#)
- GevSupportedOptionSelector_HeartbeatDisable, [252](#)
- GevSupportedOptionSelector_IPConfigurationDHCP, [252](#)
- GevSupportedOptionSelector_IPConfigurationLLA, [252](#)
- GevSupportedOptionSelector_IPConfigurationPersistentIP, [252](#)
- GevSupportedOptionSelector_LinkSpeed, [252](#)
- GevSupportedOptionSelector_ManifestTable, [252](#)
- GevSupportedOptionSelector_MessageChannelSourceSocket, [252](#)
- GevSupportedOptionSelector_PacketResend, [252](#)
- GevSupportedOptionSelector_PendingAck, [252](#)
- GevSupportedOptionSelector_SerialNumber, [252](#)
- GevSupportedOptionSelector_StreamChannelSourceSocket, [252](#)
- GevSupportedOptionSelector_TestData, [252](#)
- GevSupportedOptionSelector_UserDefinedName, [252](#)
- GevSupportedOptionSelector_WriteMem, [252](#)
- ImageComponentSelector_Color, [252](#)
- ImageComponentSelector_Confidence, [253](#)
- ImageComponentSelector_Disparity, [253](#)
- ImageComponentSelector_Infrared, [252](#)
- ImageComponentSelector_Intensity, [252](#)
- ImageComponentSelector_Range, [252](#)
- ImageComponentSelector_Scatter, [253](#)
- ImageComponentSelector_Ultraviolet, [252](#)
- ImageCompressionJPEGFormatOption_BaselineOptimized, [253](#)
- ImageCompressionJPEGFormatOption_BaselineStandard, [253](#)
- ImageCompressionJPEGFormatOption_Lossless, [253](#)
- ImageCompressionJPEGFormatOption_Progressive, [253](#)
- ImageCompressionMode_Lossless, [254](#)
- ImageCompressionMode_Off, [254](#)
- ImageCompressionRateOption_FixBitrate, [254](#)
- ImageCompressionRateOption_FixQuality, [254](#)
- LineFormat_LVDS, [254](#)
- LineFormat_NoConnect, [254](#)
- LineFormat_OpenDrain, [254](#)
- LineFormat_OptoCoupled, [254](#)
- LineFormat_RS422, [254](#)
- LineFormat_TriState, [254](#)
- LineFormat_TTL, [254](#)
- LineInputFilterSelector_Debounce, [255](#)
- LineInputFilterSelector_Degitch, [255](#)
- LineMode_Input, [255](#)
- LineMode_Output, [255](#)
- LineSelector_Line0, [255](#)
- LineSelector_Line1, [255](#)
- LineSelector_Line2, [255](#)
- LineSelector_Line3, [255](#)
- LineSource_AllPixel, [256](#)
- LineSource_AnyPixel, [256](#)
- LineSource_Counter0Active, [256](#)
- LineSource_Counter1Active, [256](#)
- LineSource_ExposureActive, [256](#)
- LineSource_FrameTriggerWait, [256](#)
- LineSource_Line0, [256](#)
- LineSource_Line1, [256](#)
- LineSource_Line2, [256](#)
- LineSource_Line3, [256](#)
- LineSource_LogicBlock0, [256](#)
- LineSource_LogicBlock1, [256](#)
- LineSource_Off, [256](#)
- LineSource_PPSSignal, [256](#)
- LineSource_SerialPort0, [256](#)
- LineSource_UserOutput0, [256](#)
- LineSource_UserOutput1, [256](#)
- LineSource_UserOutput2, [256](#)
- LineSource_UserOutput3, [256](#)
- LogicBlockLUTInputActivation_AnyEdge, [256](#)
- LogicBlockLUTInputActivation_FallingEdge, [256](#)
- LogicBlockLUTInputActivation_LevelHigh, [256](#)
- LogicBlockLUTInputActivation_LevelLow, [256](#)
- LogicBlockLUTInputActivation_RisingEdge, [256](#)
- LogicBlockLUTInputSelector_Input0, [257](#)
- LogicBlockLUTInputSelector_Input1, [257](#)
- LogicBlockLUTInputSelector_Input2, [257](#)
- LogicBlockLUTInputSelector_Input3, [257](#)
- LogicBlockLUTInputSource_AcquisitionActive, [257](#)
- LogicBlockLUTInputSource_Counter0End, [257](#)
- LogicBlockLUTInputSource_Counter0Start, [257](#)
- LogicBlockLUTInputSource_Counter1End, [257](#)
- LogicBlockLUTInputSource_Counter1Start, [257](#)
- LogicBlockLUTInputSource_ExposureEnd, [257](#)
- LogicBlockLUTInputSource_ExposureStart, [257](#)
- LogicBlockLUTInputSource_FrameTriggerWait, [257](#)
- LogicBlockLUTInputSource_Line0, [257](#)
- LogicBlockLUTInputSource_Line1, [257](#)

- LogicBlockLUTInputSource_Line2, [257](#)
- LogicBlockLUTInputSource_Line3, [257](#)
- LogicBlockLUTInputSource_LogicBlock0, [257](#)
- LogicBlockLUTInputSource_LogicBlock1, [257](#)
- LogicBlockLUTInputSource_UserOutput0, [257](#)
- LogicBlockLUTInputSource_UserOutput1, [257](#)
- LogicBlockLUTInputSource_UserOutput2, [257](#)
- LogicBlockLUTInputSource_UserOutput3, [257](#)
- LogicBlockLUTInputSource_Zero, [257](#)
- LogicBlockLUTSelector_Enable, [258](#)
- LogicBlockLUTSelector_Value, [258](#)
- LogicBlockSelector_LogicBlock0, [258](#)
- LogicBlockSelector_LogicBlock1, [258](#)
- LUTSelector_LUT1, [258](#)
- NUM_ACQUISITIONMODE, [208](#)
- NUM_ACQUISITIONSTATUSSELECTOR, [208](#)
- NUM_ACTIONUNCONDITIONALMODE, [209](#)
- NUM_ADCBITDEPTH, [209](#)
- NUM_AUTOALGORITHMSELECTOR, [209](#)
- NUM_AUTOEXPOSURECONTROLPRIORITY, [210](#)
- NUM_AUTOEXPOSURELIGHTINGMODE, [210](#)
- NUM_AUTOEXPOSUREMETERINGMODE, [211](#)
- NUM_AUTOEXPOSURETARGETGREYVALUEAUTO, [211](#)
- NUM_BALANCERATIOSELECTOR, [211](#)
- NUM_BALANCEWHITEAUTO, [212](#)
- NUM_BALANCEWHITEAUTOPROFILE, [212](#)
- NUM_BINNINGHORIZONTALMODE, [212](#)
- NUM_BINNINGSELECTOR, [213](#)
- NUM_BINNINGVERTICALMODE, [213](#)
- NUM_BLACKLEVELAUTO, [214](#)
- NUM_BLACKLEVELAUTOBALANCE, [213](#)
- NUM_BLACKLEVELSELECTOR, [214](#)
- NUM_CHUNKBLACKLEVELSELECTOR, [214](#)
- NUM_CHUNKCOUNTERSELECTOR, [215](#)
- NUM_CHUNKENCODERSELECTOR, [215](#)
- NUM_CHUNKENCODERSTATUS, [215](#)
- NUM_CHUNKEXPOSURETIMESELECTOR, [216](#)
- NUM_CHUNKGAINSELECTOR, [216](#)
- NUM_CHUNKIMAGECOMPONENT, [217](#)
- NUM_CHUNKPIXELFORMAT, [217](#)
- NUM_CHUNKREGIONID, [217](#)
- NUM_CHUNKSCAN3DCOORDINATEREFERENCESELECTOR, [218](#)
- NUM_CHUNKSCAN3DCOORDINATESELECTOR, [218](#)
- NUM_CHUNKSCAN3DCOORDINATESYSTEM, [218](#)
- NUM_CHUNKSCAN3DCOORDINATESYSTEMREFERENCESELECTOR, [219](#)
- NUM_CHUNKSCAN3DCOORDINATETRANSFORMSELECTOR, [219](#)
- NUM_CHUNKSCAN3DDISTANCEUNIT, [219](#)
- NUM_CHUNKSCAN3DOUTPUTMODE, [220](#)
- NUM_CHUNKSELECTOR, [221](#)
- NUM_CHUNKSOURCEID, [221](#)
- NUM_CHUNKTIMERSELECTOR, [222](#)
- NUM_CHUNKTRANSFERSTREAMID, [222](#)
- NUM_CLCONFIGURATION, [222](#)
- NUM_CLTIMESLOTSCOUNT, [223](#)
- NUM_COLORTRANSFORMATIONSELECTOR, [223](#)
- NUM_COLORTRANSFORMATIONVALUESELECTOR, [223](#)
- NUM_COMPRESSIONSATURATIONPRIORITY, [224](#)
- NUM_COUNTEREVENTACTIVATION, [224](#)
- NUM_COUNTEREVENTSOURCE, [225](#)
- NUM_COUNTERRESETACTIVATION, [225](#)
- NUM_COUNTERRESETSOURCE, [226](#)
- NUM_COUNTERSELECTOR, [226](#)
- NUM_COUNTERSTATUS, [226](#)
- NUM_COUNTERTRIGGERACTIVATION, [227](#)
- NUM_COUNTERTRIGGERSOURCE, [227](#)
- NUM_CXPCONNECTIONTESTMODE, [228](#)
- NUM_CXPLINKCONFIGURATION, [229](#)
- NUM_CXPLINKCONFIGURATIONPREFERRED, [230](#)
- NUM_CXPLINKCONFIGURATIONSTATUS, [231](#)
- NUM_CXPPOCXPSTATUS, [231](#)
- NUM_DECIMATIONHORIZONTALMODE, [231](#)
- NUM_DECIMATIONSELECTOR, [232](#)
- NUM_DECIMATIONVERTICALMODE, [232](#)
- NUM_DEFECTCORRECTIONMODE, [232](#)
- NUM_DEINTERLACING, [233](#)
- NUM_DEVICECHARACTERSET, [233](#)
- NUM_DEVICECLOCKSELECTOR, [233](#)
- NUM_DEVICECONNECTIONSTATUS, [233](#)
- NUM_DEVICEINDICATORMODE, [234](#)
- NUM_DEVICELINKHEARTBEATMODE, [234](#)
- NUM_DEVICELINKTHROUGHPUTLIMITMODE, [234](#)
- NUM_DEVICEPOWERSUPPLYSELECTOR, [235](#)
- NUM_DEVICEREGISTERSENDIANNES, [235](#)
- NUM_DEVICESCANTYPE, [235](#)
- NUM_DEVICESERIALPORTBAUDRATE, [236](#)
- NUM_DEVICESERIALPORTSELECTOR, [236](#)
- NUM_DEVICESTREAMCHANNELENDIANNES, [236](#)
- NUM_DEVICESTREAMCHANNELTYPE, [237](#)
- NUM_DEVICETAPGEOMETRY, [238](#)
- NUM_DEVICETEMPERATURESELECTOR, [238](#)
- NUM_DEVICETLTYPE, [238](#)
- NUM_DEVICETYPE, [239](#)
- NUM_ENCODERMODE, [239](#)
- NUM_ENCODEROUTPUTMODE, [240](#)
- NUM_ENCODERRESETACTIVATION, [240](#)
- NUM_ENCODERRESETSOURCE, [241](#)
- NUM_ENCODERSELECTOR, [241](#)
- NUM_ENCODERSOURCEA, [242](#)
- NUM_ENCODERSOURCEB, [242](#)
- NUM_ENCODERSTATUS, [242](#)
- NUM_EVENTNOTIFICATION, [243](#)
- NUM_EVENTSELECTOR, [243](#)
- NUM_EXPOSUREACTIVEMODE, [243](#)

- NUM_EXPOSUREAUTO, [244](#)
- NUM_EXPOSUREMODE, [244](#)
- NUM_EXPOSURETIMEMODE, [244](#)
- NUM_EXPOSURETIMESELECTOR, [245](#)
- NUM_FILEOPENMODE, [245](#)
- NUM_FILEOPERATIONSELECTOR, [246](#)
- NUM_FILEOPERATIONSTATUS, [246](#)
- NUM_FILESELECTOR, [246](#)
- NUM_GAINAUTO, [248](#)
- NUM_GAINAUTOBALANCE, [248](#)
- NUM_GAINSELECTOR, [248](#)
- NUM_GEVCCP, [249](#)
- NUM_GEVCURRENTPHYSICALLINKCONFIGURATION, [249](#)
- NUM_GEVGVCPEXTENDEDSTATUSCODESSELECTOR, [249](#)
- NUM_GEVGVSPEXTENDEDIDMODE, [250](#)
- NUM_GEVEEEE1588CLOCKACCURACY, [250](#)
- NUM_GEVEEEE1588MODE, [250](#)
- NUM_GEVEEEE1588STATUS, [251](#)
- NUM_GEVIPCONFIGURATIONSTATUS, [251](#)
- NUM_GEVPHYSCALLINKCONFIGURATION, [251](#)
- NUM_GEVSUPPORTEDOPTIONSELECTOR, [252](#)
- NUM_IMAGECOMPONENTSELECTOR, [253](#)
- NUM_IMAGECOMPRESSIONJPEGFORMATOPTION, [253](#)
- NUM_IMAGECOMPRESSIONMODE, [254](#)
- NUM_IMAGECOMPRESSIONRATEOPTION, [254](#)
- NUM_LINEFORMAT, [254](#)
- NUM_LINEINPUTFILTERSELECTOR, [255](#)
- NUM_LINEMODE, [255](#)
- NUM_LINESELECTOR, [255](#)
- NUM_LINESOURCE, [256](#)
- NUM_LOGICBLOCKLUTINPUTACTIVATION, [256](#)
- NUM_LOGICBLOCKLUTINPUTSELECTOR, [257](#)
- NUM_LOGICBLOCKLUTINPUTSOURCE, [257](#)
- NUM_LOGICBLOCKLUTSELECTOR, [258](#)
- NUM_LOGICBLOCKSELECTOR, [258](#)
- NUM_LUTSELECTOR, [258](#)
- NUM_PIXELCOLORFILTER, [259](#)
- NUM_PIXELFORMAT, [264](#)
- NUM_PIXELFORMATINFOSELECTOR, [270](#)
- NUM_PIXELSIZE, [271](#)
- NUM_REGIONDESTINATION, [271](#)
- NUM_REGIONMODE, [271](#)
- NUM_REGIONSELECTOR, [272](#)
- NUM_RGBTRANSFORMLIGHTSOURCE, [272](#)
- NUM_SCAN3DCOORDINATEREFERENCESELECTOR, [273](#)
- NUM_SCAN3DCOORDINATESELECTOR, [273](#)
- NUM_SCAN3DCOORDINATESYSTEM, [273](#)
- NUM_SCAN3DCOORDINATESYSTEMREFERENCE, [274](#)
- NUM_SCAN3DCOORDINATETRANSFORMSELECTOR, [274](#)
- NUM_SCAN3DDISTANCEUNIT, [274](#)
- NUM_SCAN3DOUTPUTMODE, [276](#)
- NUM_SENSORDIGITIZATIONTAPS, [277](#)
- NUM_SENSORSHUTTERMODE, [277](#)
- NUM_SENSORTAPS, [277](#)
- NUM_SEQUENCERCONFIGURATIONMODE, [278](#)
- NUM_SEQUENCERCONFIGURATIONVALID, [278](#)
- NUM_SEQUENCERMODE, [278](#)
- NUM_SEQUENCERSETVALID, [279](#)
- NUM_SEQUENCERTRIGGERACTIVATION, [279](#)
- NUM_SEQUENCERTRIGGERSOURCE, [279](#)
- NUM_SERIALPORTBAUDRATE, [280](#)
- NUM_SERIALPORTPARITY, [280](#)
- NUM_SERIALPORTSELECTOR, [280](#)
- NUM_SERIALPORTSOURCE, [281](#)
- NUM_SERIALPORTSTOPBITS, [281](#)
- NUM_SOFTWARESIGNALSELECTOR, [281](#)
- NUM_SOURCESELECTOR, [282](#)
- NUM_TESTPATTERN, [282](#)
- NUM_TESTPATTERNGENERATORSELECTOR, [282](#)
- NUM_TIMERSELECTOR, [283](#)
- NUM_TIMERSTATUS, [283](#)
- NUM_TIMERTRIGGERACTIVATION, [283](#)
- NUM_TIMERTRIGGERSOURCE, [285](#)
- NUM_TRANSFERCOMPONENTSELECTOR, [285](#)
- NUM_TRANSFERCONTROLMODE, [286](#)
- NUM_TRANSFEROPERATIONMODE, [286](#)
- NUM_TRANSFERQUEUEMODE, [286](#)
- NUM_TRANSFERSELECTOR, [286](#)
- NUM_TRANSFERSTATUSSELECTOR, [287](#)
- NUM_TRANSFERTRIGGERACTIVATION, [287](#)
- NUM_TRANSFERTRIGGERMODE, [288](#)
- NUM_TRANSFERTRIGGERSELECTOR, [288](#)
- NUM_TRANSFERTRIGGERSOURCE, [289](#)
- NUM_TRIGGERACTIVATION, [289](#)
- NUM_TRIGGERMODE, [290](#)
- NUM_TRIGGEROVERLAP, [290](#)
- NUM_TRIGGERSELECTOR, [290](#)
- NUM_TRIGGERSOURCE, [291](#)
- NUM_USEROUTPUTSELECTOR, [291](#)
- NUM_USERSETDEFAULT, [292](#)
- NUM_USERSETSELECTOR, [292](#)
- NUM_WHITECLIPSELECTOR, [292](#)
- PixelColorFilter_BayerBG, [259](#)
- PixelColorFilter_BayerGB, [259](#)
- PixelColorFilter_BayerGR, [259](#)
- PixelColorFilter_BayerRG, [258](#)
- PixelColorFilter_None, [258](#)
- PixelFormat_B10, [261](#)
- PixelFormat_B12, [261](#)
- PixelFormat_B12_Jpeg, [264](#)
- PixelFormat_B16, [261](#)
- PixelFormat_B8, [261](#)
- PixelFormat_BayerBG10, [260](#)
- PixelFormat_BayerBG10p, [260](#)
- PixelFormat_BayerBG10Packed, [260](#)
- PixelFormat_BayerBG12, [260](#)
- PixelFormat_BayerBG12p, [259](#)

PixelFormat_BayerBG12Packed, 259
PixelFormat_BayerBG16, 259
PixelFormat_BayerBG8, 259
PixelFormat_BayerGB10, 260
PixelFormat_BayerGB10p, 260
PixelFormat_BayerGB10Packed, 260
PixelFormat_BayerGB12, 260
PixelFormat_BayerGB12p, 259
PixelFormat_BayerGB12Packed, 259
PixelFormat_BayerGB16, 259
PixelFormat_BayerGB8, 259
PixelFormat_BayerGR10, 260
PixelFormat_BayerGR10p, 260
PixelFormat_BayerGR10Packed, 259
PixelFormat_BayerGR12, 260
PixelFormat_BayerGR12p, 259
PixelFormat_BayerGR12Packed, 259
PixelFormat_BayerGR16, 259
PixelFormat_BayerGR8, 259
PixelFormat_BayerRG10, 260
PixelFormat_BayerRG10p, 260
PixelFormat_BayerRG10Packed, 259
PixelFormat_BayerRG12, 260
PixelFormat_BayerRG12p, 259
PixelFormat_BayerRG12Packed, 259
PixelFormat_BayerRG16, 259
PixelFormat_BayerRG8, 259
PixelFormat_BayerRGPolarized10p, 264
PixelFormat_BayerRGPolarized12p, 264
PixelFormat_BayerRGPolarized16, 264
PixelFormat_BayerRGPolarized8, 264
PixelFormat_BGR10, 261
PixelFormat_BGR10p, 261
PixelFormat_BGR12, 261
PixelFormat_BGR12p, 261
PixelFormat_BGR14, 261
PixelFormat_BGR16, 261
PixelFormat_BGR565p, 261
PixelFormat_BGR8, 259
PixelFormat_BGRa10, 261
PixelFormat_BGRa10p, 261
PixelFormat_BGRa12, 261
PixelFormat_BGRa12p, 261
PixelFormat_BGRa14, 261
PixelFormat_BGRa16, 261
PixelFormat_BGRa8, 259
PixelFormat_BiColorBGRG10, 262
PixelFormat_BiColorBGRG10p, 262
PixelFormat_BiColorBGRG12, 262
PixelFormat_BiColorBGRG12p, 262
PixelFormat_BiColorBGRG8, 262
PixelFormat_BiColorRGBG10, 262
PixelFormat_BiColorRGBG10p, 262
PixelFormat_BiColorRGBG12, 262
PixelFormat_BiColorRGBG12p, 262
PixelFormat_BiColorRGBG8, 262
PixelFormat_Confidence1, 262
PixelFormat_Confidence16, 262
PixelFormat_Confidence1p, 262
PixelFormat_Confidence32f, 262
PixelFormat_Confidence8, 262
PixelFormat_Coord3D_A10p, 262
PixelFormat_Coord3D_A12p, 262
PixelFormat_Coord3D_A16, 262
PixelFormat_Coord3D_A32f, 262
PixelFormat_Coord3D_A8, 262
PixelFormat_Coord3D_ABC10p, 261
PixelFormat_Coord3D_ABC10p_Planar, 261
PixelFormat_Coord3D_ABC12p, 261
PixelFormat_Coord3D_ABC12p_Planar, 261
PixelFormat_Coord3D_ABC16, 261
PixelFormat_Coord3D_ABC16_Planar, 261
PixelFormat_Coord3D_ABC32f, 261
PixelFormat_Coord3D_ABC32f_Planar, 261
PixelFormat_Coord3D_ABC8, 261
PixelFormat_Coord3D_ABC8_Planar, 261
PixelFormat_Coord3D_AC10p, 261
PixelFormat_Coord3D_AC10p_Planar, 261
PixelFormat_Coord3D_AC12p, 261
PixelFormat_Coord3D_AC12p_Planar, 261
PixelFormat_Coord3D_AC16, 261
PixelFormat_Coord3D_AC16_Planar, 261
PixelFormat_Coord3D_AC32f, 261
PixelFormat_Coord3D_AC32f_Planar, 261
PixelFormat_Coord3D_AC8, 261
PixelFormat_Coord3D_AC8_Planar, 261
PixelFormat_Coord3D_B10p, 262
PixelFormat_Coord3D_B12p, 262
PixelFormat_Coord3D_B16, 262
PixelFormat_Coord3D_B32f, 262
PixelFormat_Coord3D_B8, 262
PixelFormat_Coord3D_C10p, 262
PixelFormat_Coord3D_C12p, 262
PixelFormat_Coord3D_C16, 262
PixelFormat_Coord3D_C32f, 262
PixelFormat_Coord3D_C8, 262
PixelFormat_G10, 261
PixelFormat_G12, 261
PixelFormat_G16, 261
PixelFormat_G8, 261
PixelFormat_GB12, 264
PixelFormat_GB12_Jpeg, 264
PixelFormat_GR12, 264
PixelFormat_GR12_Jpeg, 264
PixelFormat_JPEGColor8, 264
PixelFormat_JPEGMono8, 264
PixelFormat_LLCBayerRG8, 264
PixelFormat_LLCMono8, 264
PixelFormat_Mono10, 260
PixelFormat_Mono10p, 260
PixelFormat_Mono10Packed, 259
PixelFormat_Mono12, 260
PixelFormat_Mono12p, 259
PixelFormat_Mono12Packed, 259
PixelFormat_Mono14, 260
PixelFormat_Mono16, 259

PixelFormat_Mono16s, [260](#)
PixelFormat_Mono1p, [260](#)
PixelFormat_Mono2p, [260](#)
PixelFormat_Mono32f, [260](#)
PixelFormat_Mono4p, [260](#)
PixelFormat_Mono8, [259](#)
PixelFormat_Mono8s, [260](#)
PixelFormat_Polarized10p, [264](#)
PixelFormat_Polarized12p, [264](#)
PixelFormat_Polarized16, [264](#)
PixelFormat_Polarized8, [264](#)
PixelFormat_R10, [261](#)
PixelFormat_R12, [261](#)
PixelFormat_R12_Jpeg, [264](#)
PixelFormat_R16, [261](#)
PixelFormat_R8, [261](#)
PixelFormat_Raw16, [264](#)
PixelFormat_Raw8, [264](#)
PixelFormat_RGB10, [260](#)
PixelFormat_RGB10_Planar, [260](#)
PixelFormat_RGB10p, [260](#)
PixelFormat_RGB10p32, [260](#)
PixelFormat_RGB12, [260](#)
PixelFormat_RGB12_Planar, [260](#)
PixelFormat_RGB12p, [260](#)
PixelFormat_RGB14, [260](#)
PixelFormat_RGB16, [260](#)
PixelFormat_RGB16_Planar, [260](#)
PixelFormat_RGB16s, [260](#)
PixelFormat_RGB32f, [260](#)
PixelFormat_RGB565p, [261](#)
PixelFormat_RGB8, [260](#)
PixelFormat_RGB8_Planar, [260](#)
PixelFormat_RGB8Packed, [259](#)
PixelFormat_RGBa10, [260](#)
PixelFormat_RGBa10p, [260](#)
PixelFormat_RGBa12, [260](#)
PixelFormat_RGBa12p, [260](#)
PixelFormat_RGBa14, [260](#)
PixelFormat_RGBa16, [260](#)
PixelFormat_RGBa32f, [261](#)
PixelFormat_RGBa8, [260](#)
PixelFormat_SCF1WBWG10, [262](#)
PixelFormat_SCF1WBWG10p, [262](#)
PixelFormat_SCF1WBWG12, [262](#)
PixelFormat_SCF1WBWG12p, [262](#)
PixelFormat_SCF1WBWG14, [262](#)
PixelFormat_SCF1WBWG16, [262](#)
PixelFormat_SCF1WBWG8, [262](#)
PixelFormat_SCF1WGWB10, [262](#)
PixelFormat_SCF1WGWB10p, [262](#)
PixelFormat_SCF1WGWB12, [262](#)
PixelFormat_SCF1WGWB12p, [262](#)
PixelFormat_SCF1WGWB14, [262](#)
PixelFormat_SCF1WGWB16, [262](#)
PixelFormat_SCF1WGWB8, [262](#)
PixelFormat_SCF1WGWR10, [262](#)
PixelFormat_SCF1WGWR10p, [262](#)
PixelFormat_SCF1WGWR12, [263](#)
PixelFormat_SCF1WGWR12p, [263](#)
PixelFormat_SCF1WGWR14, [263](#)
PixelFormat_SCF1WGWR16, [263](#)
PixelFormat_SCF1WGWR8, [262](#)
PixelFormat_SCF1WRWG10, [263](#)
PixelFormat_SCF1WRWG10p, [263](#)
PixelFormat_SCF1WRWG12, [263](#)
PixelFormat_SCF1WRWG12p, [263](#)
PixelFormat_SCF1WRWG14, [263](#)
PixelFormat_SCF1WRWG16, [263](#)
PixelFormat_SCF1WRWG8, [263](#)
PixelFormat_YCbCr10_CbYCr, [263](#)
PixelFormat_YCbCr10p_CbYCr, [263](#)
PixelFormat_YCbCr12_CbYCr, [263](#)
PixelFormat_YCbCr12p_CbYCr, [263](#)
PixelFormat_YCbCr411_8, [259](#)
PixelFormat_YCbCr411_8_CbYYCrYY, [263](#)
PixelFormat_YCbCr422_10, [263](#)
PixelFormat_YCbCr422_10_CbYCrY, [263](#)
PixelFormat_YCbCr422_10p, [263](#)
PixelFormat_YCbCr422_10p_CbYCrY, [263](#)
PixelFormat_YCbCr422_12, [263](#)
PixelFormat_YCbCr422_12_CbYCrY, [263](#)
PixelFormat_YCbCr422_12p, [263](#)
PixelFormat_YCbCr422_12p_CbYCrY, [263](#)
PixelFormat_YCbCr422_8, [259](#)
PixelFormat_YCbCr422_8_CbYCrY, [263](#)
PixelFormat_YCbCr601_10_CbYCr, [263](#)
PixelFormat_YCbCr601_10p_CbYCr, [263](#)
PixelFormat_YCbCr601_12_CbYCr, [263](#)
PixelFormat_YCbCr601_12p_CbYCr, [263](#)
PixelFormat_YCbCr601_411_8_CbYYCrYY, [263](#)
PixelFormat_YCbCr601_422_10, [263](#)
PixelFormat_YCbCr601_422_10_CbYCrY, [263](#)
PixelFormat_YCbCr601_422_10p, [263](#)
PixelFormat_YCbCr601_422_10p_CbYCrY, [263](#)
PixelFormat_YCbCr601_422_12, [263](#)
PixelFormat_YCbCr601_422_12_CbYCrY, [263](#)
PixelFormat_YCbCr601_422_12p, [263](#)
PixelFormat_YCbCr601_422_12p_CbYCrY, [263](#)
PixelFormat_YCbCr601_422_8, [263](#)
PixelFormat_YCbCr601_422_8_CbYCrY, [263](#)
PixelFormat_YCbCr601_8_CbYCr, [263](#)
PixelFormat_YCbCr709_10_CbYCr, [263](#)
PixelFormat_YCbCr709_10p_CbYCr, [263](#)
PixelFormat_YCbCr709_12_CbYCr, [263](#)
PixelFormat_YCbCr709_12p_CbYCr, [263](#)
PixelFormat_YCbCr709_411_8_CbYYCrYY, [264](#)
PixelFormat_YCbCr709_422_10, [264](#)
PixelFormat_YCbCr709_422_10_CbYCrY, [264](#)
PixelFormat_YCbCr709_422_10p, [264](#)
PixelFormat_YCbCr709_422_10p_CbYCrY, [264](#)
PixelFormat_YCbCr709_422_12, [264](#)
PixelFormat_YCbCr709_422_12_CbYCrY, [264](#)
PixelFormat_YCbCr709_422_12p, [264](#)
PixelFormat_YCbCr709_422_12p_CbYCrY, [264](#)
PixelFormat_YCbCr709_422_8, [264](#)

- PixelFormat_YCbCr709_422_8_CbYCrY, [264](#)
- PixelFormat_YCbCr709_8_CbYCr, [263](#)
- PixelFormat_YCbCr8, [259](#)
- PixelFormat_YCbCr8_CbYCr, [263](#)
- PixelFormat_YUV411_8_UYYVYY, [264](#)
- PixelFormat_YUV411Packed, [259](#)
- PixelFormat_YUV422_8, [264](#)
- PixelFormat_YUV422_8_UYVY, [264](#)
- PixelFormat_YUV422Packed, [259](#)
- PixelFormat_YUV444Packed, [259](#)
- PixelFormat_YUV8_UYV, [264](#)
- PixelFormatInfoSelector_B10, [266](#)
- PixelFormatInfoSelector_B12, [266](#)
- PixelFormatInfoSelector_B16, [266](#)
- PixelFormatInfoSelector_B8, [266](#)
- PixelFormatInfoSelector_BayerBG10, [265](#)
- PixelFormatInfoSelector_BayerBG10p, [265](#)
- PixelFormatInfoSelector_BayerBG12, [265](#)
- PixelFormatInfoSelector_BayerBG12p, [265](#)
- PixelFormatInfoSelector_BayerBG16, [265](#)
- PixelFormatInfoSelector_BayerBG8, [265](#)
- PixelFormatInfoSelector_BayerGB10, [265](#)
- PixelFormatInfoSelector_BayerGB10p, [265](#)
- PixelFormatInfoSelector_BayerGB12, [265](#)
- PixelFormatInfoSelector_BayerGB12p, [265](#)
- PixelFormatInfoSelector_BayerGB16, [265](#)
- PixelFormatInfoSelector_BayerGB8, [265](#)
- PixelFormatInfoSelector_BayerGR10, [265](#)
- PixelFormatInfoSelector_BayerGR10p, [265](#)
- PixelFormatInfoSelector_BayerGR12, [265](#)
- PixelFormatInfoSelector_BayerGR12p, [265](#)
- PixelFormatInfoSelector_BayerGR16, [265](#)
- PixelFormatInfoSelector_BayerGR8, [265](#)
- PixelFormatInfoSelector_BayerRG10, [265](#)
- PixelFormatInfoSelector_BayerRG10p, [265](#)
- PixelFormatInfoSelector_BayerRG12, [265](#)
- PixelFormatInfoSelector_BayerRG12p, [265](#)
- PixelFormatInfoSelector_BayerRG16, [265](#)
- PixelFormatInfoSelector_BayerRG8, [265](#)
- PixelFormatInfoSelector_BayerRGPolarized10p, [270](#)
- PixelFormatInfoSelector_BayerRGPolarized12p, [270](#)
- PixelFormatInfoSelector_BayerRGPolarized16, [270](#)
- PixelFormatInfoSelector_BayerRGPolarized8, [270](#)
- PixelFormatInfoSelector_BGR10, [266](#)
- PixelFormatInfoSelector_BGR10p, [266](#)
- PixelFormatInfoSelector_BGR12, [266](#)
- PixelFormatInfoSelector_BGR12p, [266](#)
- PixelFormatInfoSelector_BGR14, [266](#)
- PixelFormatInfoSelector_BGR16, [266](#)
- PixelFormatInfoSelector_BGR565p, [266](#)
- PixelFormatInfoSelector_BGR8, [266](#)
- PixelFormatInfoSelector_BGRa10, [266](#)
- PixelFormatInfoSelector_BGRa10p, [266](#)
- PixelFormatInfoSelector_BGRa12, [266](#)
- PixelFormatInfoSelector_BGRa12p, [266](#)
- PixelFormatInfoSelector_BGRa14, [266](#)
- PixelFormatInfoSelector_BGRa16, [266](#)
- PixelFormatInfoSelector_BGRa8, [266](#)
- PixelFormatInfoSelector_BiColorBGRG10, [267](#)
- PixelFormatInfoSelector_BiColorBGRG10p, [267](#)
- PixelFormatInfoSelector_BiColorBGRG12, [267](#)
- PixelFormatInfoSelector_BiColorBGRG12p, [267](#)
- PixelFormatInfoSelector_BiColorBGRG8, [267](#)
- PixelFormatInfoSelector_BiColorRGBG10, [267](#)
- PixelFormatInfoSelector_BiColorRGBG10p, [267](#)
- PixelFormatInfoSelector_BiColorRGBG12, [267](#)
- PixelFormatInfoSelector_BiColorRGBG12p, [267](#)
- PixelFormatInfoSelector_BiColorRGBG8, [267](#)
- PixelFormatInfoSelector_Confidence1, [267](#)
- PixelFormatInfoSelector_Confidence16, [267](#)
- PixelFormatInfoSelector_Confidence1p, [267](#)
- PixelFormatInfoSelector_Confidence32f, [267](#)
- PixelFormatInfoSelector_Confidence8, [267](#)
- PixelFormatInfoSelector_Coord3D_A10p, [267](#)
- PixelFormatInfoSelector_Coord3D_A12p, [267](#)
- PixelFormatInfoSelector_Coord3D_A16, [267](#)
- PixelFormatInfoSelector_Coord3D_A32f, [267](#)
- PixelFormatInfoSelector_Coord3D_A8, [267](#)
- PixelFormatInfoSelector_Coord3D_ABC10p, [266](#)
- PixelFormatInfoSelector_Coord3D_ABC10p_Planar, [266](#)
- PixelFormatInfoSelector_Coord3D_ABC12p, [266](#)
- PixelFormatInfoSelector_Coord3D_ABC12p_Planar, [266](#)
- PixelFormatInfoSelector_Coord3D_ABC16, [267](#)
- PixelFormatInfoSelector_Coord3D_ABC16_Planar, [267](#)
- PixelFormatInfoSelector_Coord3D_ABC32f, [267](#)
- PixelFormatInfoSelector_Coord3D_ABC32f_Planar, [267](#)
- PixelFormatInfoSelector_Coord3D_ABC8, [266](#)
- PixelFormatInfoSelector_Coord3D_ABC8_Planar, [266](#)
- PixelFormatInfoSelector_Coord3D_AC10p, [267](#)
- PixelFormatInfoSelector_Coord3D_AC10p_Planar, [267](#)
- PixelFormatInfoSelector_Coord3D_AC12p, [267](#)
- PixelFormatInfoSelector_Coord3D_AC12p_Planar, [267](#)
- PixelFormatInfoSelector_Coord3D_AC16, [267](#)
- PixelFormatInfoSelector_Coord3D_AC16_Planar, [267](#)
- PixelFormatInfoSelector_Coord3D_AC32f, [267](#)
- PixelFormatInfoSelector_Coord3D_AC32f_Planar, [267](#)
- PixelFormatInfoSelector_Coord3D_AC8, [267](#)
- PixelFormatInfoSelector_Coord3D_AC8_Planar, [267](#)
- PixelFormatInfoSelector_Coord3D_B10p, [267](#)
- PixelFormatInfoSelector_Coord3D_B12p, [267](#)
- PixelFormatInfoSelector_Coord3D_B16, [267](#)
- PixelFormatInfoSelector_Coord3D_B32f, [267](#)
- PixelFormatInfoSelector_Coord3D_B8, [267](#)

- PixelFormatInfoSelector_Coord3D_C10p, [267](#)
PixelFormatInfoSelector_Coord3D_C12p, [267](#)
PixelFormatInfoSelector_Coord3D_C16, [267](#)
PixelFormatInfoSelector_Coord3D_C32f, [267](#)
PixelFormatInfoSelector_Coord3D_C8, [267](#)
PixelFormatInfoSelector_G10, [266](#)
PixelFormatInfoSelector_G12, [266](#)
PixelFormatInfoSelector_G16, [266](#)
PixelFormatInfoSelector_G8, [266](#)
PixelFormatInfoSelector_JPEGColor8, [270](#)
PixelFormatInfoSelector_JPEGMono8, [270](#)
PixelFormatInfoSelector_LLCBayerRG8, [270](#)
PixelFormatInfoSelector_LLCMono8, [270](#)
PixelFormatInfoSelector_Mono10, [265](#)
PixelFormatInfoSelector_Mono10p, [265](#)
PixelFormatInfoSelector_Mono12, [265](#)
PixelFormatInfoSelector_Mono12p, [265](#)
PixelFormatInfoSelector_Mono14, [265](#)
PixelFormatInfoSelector_Mono16, [265](#)
PixelFormatInfoSelector_Mono16s, [265](#)
PixelFormatInfoSelector_Mono1p, [265](#)
PixelFormatInfoSelector_Mono2p, [265](#)
PixelFormatInfoSelector_Mono32f, [265](#)
PixelFormatInfoSelector_Mono4p, [265](#)
PixelFormatInfoSelector_Mono8, [265](#)
PixelFormatInfoSelector_Mono8s, [265](#)
PixelFormatInfoSelector_Polarized10p, [270](#)
PixelFormatInfoSelector_Polarized12p, [270](#)
PixelFormatInfoSelector_Polarized16, [270](#)
PixelFormatInfoSelector_Polarized8, [270](#)
PixelFormatInfoSelector_R10, [266](#)
PixelFormatInfoSelector_R12, [266](#)
PixelFormatInfoSelector_R16, [266](#)
PixelFormatInfoSelector_R8, [266](#)
PixelFormatInfoSelector_RGB10, [265](#)
PixelFormatInfoSelector_RGB10_Planar, [266](#)
PixelFormatInfoSelector_RGB10p, [266](#)
PixelFormatInfoSelector_RGB10p32, [266](#)
PixelFormatInfoSelector_RGB12, [266](#)
PixelFormatInfoSelector_RGB12_Planar, [266](#)
PixelFormatInfoSelector_RGB12p, [266](#)
PixelFormatInfoSelector_RGB14, [266](#)
PixelFormatInfoSelector_RGB16, [266](#)
PixelFormatInfoSelector_RGB16_Planar, [266](#)
PixelFormatInfoSelector_RGB16s, [266](#)
PixelFormatInfoSelector_RGB32f, [266](#)
PixelFormatInfoSelector_RGB565p, [266](#)
PixelFormatInfoSelector_RGB8, [265](#)
PixelFormatInfoSelector_RGB8_Planar, [265](#)
PixelFormatInfoSelector_RGBa10, [265](#)
PixelFormatInfoSelector_RGBa10p, [265](#)
PixelFormatInfoSelector_RGBa12, [265](#)
PixelFormatInfoSelector_RGBa12p, [265](#)
PixelFormatInfoSelector_RGBa14, [265](#)
PixelFormatInfoSelector_RGBa16, [265](#)
PixelFormatInfoSelector_RGBa32f, [266](#)
PixelFormatInfoSelector_RGBa8, [265](#)
PixelFormatInfoSelector_SCF1WBWG10, [267](#)
PixelFormatInfoSelector_SCF1WBWG10p, [268](#)
PixelFormatInfoSelector_SCF1WBWG12, [268](#)
PixelFormatInfoSelector_SCF1WBWG12p, [268](#)
PixelFormatInfoSelector_SCF1WBWG14, [268](#)
PixelFormatInfoSelector_SCF1WBWG16, [268](#)
PixelFormatInfoSelector_SCF1WBWG8, [267](#)
PixelFormatInfoSelector_SCF1WGWB10, [268](#)
PixelFormatInfoSelector_SCF1WGWB10p, [268](#)
PixelFormatInfoSelector_SCF1WGWB12, [268](#)
PixelFormatInfoSelector_SCF1WGWB12p, [268](#)
PixelFormatInfoSelector_SCF1WGWB14, [268](#)
PixelFormatInfoSelector_SCF1WGWB16, [268](#)
PixelFormatInfoSelector_SCF1WGWB8, [268](#)
PixelFormatInfoSelector_SCF1WGWR10, [268](#)
PixelFormatInfoSelector_SCF1WGWR10p, [268](#)
PixelFormatInfoSelector_SCF1WGWR12, [268](#)
PixelFormatInfoSelector_SCF1WGWR12p, [268](#)
PixelFormatInfoSelector_SCF1WGWR14, [268](#)
PixelFormatInfoSelector_SCF1WGWR16, [268](#)
PixelFormatInfoSelector_SCF1WGWR8, [268](#)
PixelFormatInfoSelector_SCF1WRWG10, [268](#)
PixelFormatInfoSelector_SCF1WRWG10p, [268](#)
PixelFormatInfoSelector_SCF1WRWG12, [268](#)
PixelFormatInfoSelector_SCF1WRWG12p, [268](#)
PixelFormatInfoSelector_SCF1WRWG14, [268](#)
PixelFormatInfoSelector_SCF1WRWG16, [268](#)
PixelFormatInfoSelector_SCF1WRWG8, [268](#)
PixelFormatInfoSelector_YCbCr10_CbYCr, [268](#)
PixelFormatInfoSelector_YCbCr10p_CbYCr, [268](#)
PixelFormatInfoSelector_YCbCr12_CbYCr, [269](#)
PixelFormatInfoSelector_YCbCr12p_CbYCr, [269](#)
PixelFormatInfoSelector_YCbCr411_8, [269](#)
PixelFormatInfoSelector_YCbCr411_8_CbYYCrYY, [269](#)
PixelFormatInfoSelector_YCbCr422_10, [269](#)
PixelFormatInfoSelector_YCbCr422_10_CbYCrY, [269](#)
PixelFormatInfoSelector_YCbCr422_10p, [269](#)
PixelFormatInfoSelector_YCbCr422_10p_CbYCrY, [269](#)
PixelFormatInfoSelector_YCbCr422_12, [269](#)
PixelFormatInfoSelector_YCbCr422_12_CbYCrY, [269](#)
PixelFormatInfoSelector_YCbCr422_12p, [269](#)
PixelFormatInfoSelector_YCbCr422_12p_CbYCrY, [269](#)
PixelFormatInfoSelector_YCbCr422_8, [269](#)
PixelFormatInfoSelector_YCbCr422_8_CbYCrY, [269](#)
PixelFormatInfoSelector_YCbCr601_10_CbYCr, [269](#)
PixelFormatInfoSelector_YCbCr601_10p_CbYCr, [269](#)
PixelFormatInfoSelector_YCbCr601_12_CbYCr, [269](#)
PixelFormatInfoSelector_YCbCr601_12p_CbYCr, [269](#)
PixelFormatInfoSelector_YCbCr601_411_8_CbYYCrYY, [269](#)

- [269](#)
- [PixelFormatInfoSelector_YCbCr601_422_10, 269](#)
- [PixelFormatInfoSelector_YCbCr601_422_10_CbYCrY, 269](#)
- [PixelFormatInfoSelector_YCbCr601_422_10p, 269](#)
- [PixelFormatInfoSelector_YCbCr601_422_10p_CbYCrY, 269](#)
- [PixelFormatInfoSelector_YCbCr601_422_12, 269](#)
- [PixelFormatInfoSelector_YCbCr601_422_12_CbYCrY, 269](#)
- [PixelFormatInfoSelector_YCbCr601_422_12p, 269](#)
- [PixelFormatInfoSelector_YCbCr601_422_12p_CbYCrY, 269](#)
- [PixelFormatInfoSelector_YCbCr601_422_8, 269](#)
- [PixelFormatInfoSelector_YCbCr601_422_8_CbYCrY, 269](#)
- [PixelFormatInfoSelector_YCbCr601_8_CbYCr, 269](#)
- [PixelFormatInfoSelector_YCbCr709_10_CbYCr, 269](#)
- [PixelFormatInfoSelector_YCbCr709_10p_CbYCr, 269](#)
- [PixelFormatInfoSelector_YCbCr709_12_CbYCr, 269](#)
- [PixelFormatInfoSelector_YCbCr709_12p_CbYCr, 269](#)
- [PixelFormatInfoSelector_YCbCr709_411_8_CbYYCrYY, 269](#)
- [PixelFormatInfoSelector_YCbCr709_422_10, 269](#)
- [PixelFormatInfoSelector_YCbCr709_422_10_CbYCrY, 269](#)
- [PixelFormatInfoSelector_YCbCr709_422_10p, 269](#)
- [PixelFormatInfoSelector_YCbCr709_422_10p_CbYCrY, 270](#)
- [PixelFormatInfoSelector_YCbCr709_422_12, 270](#)
- [PixelFormatInfoSelector_YCbCr709_422_12_CbYCrY, 270](#)
- [PixelFormatInfoSelector_YCbCr709_422_12p, 270](#)
- [PixelFormatInfoSelector_YCbCr709_422_12p_CbYCrY, 270](#)
- [PixelFormatInfoSelector_YCbCr709_422_8, 269](#)
- [PixelFormatInfoSelector_YCbCr709_422_8_CbYCrY, 269](#)
- [PixelFormatInfoSelector_YCbCr709_8_CbYCr, 269](#)
- [PixelFormatInfoSelector_YCbCr8, 268](#)
- [PixelFormatInfoSelector_YCbCr8_CbYCr, 268](#)
- [PixelFormatInfoSelector_YUV411_8_UYYVYY, 270](#)
- [PixelFormatInfoSelector_YUV422_8, 270](#)
- [PixelFormatInfoSelector_YUV422_8_UYVY, 270](#)
- [PixelFormatInfoSelector_YUV8_UYV, 270](#)
- [PixelSize_Bpp1, 270](#)
- [PixelSize_Bpp10, 270](#)
- [PixelSize_Bpp12, 270](#)
- [PixelSize_Bpp14, 270](#)
- [PixelSize_Bpp16, 270](#)
- [PixelSize_Bpp2, 270](#)
- [PixelSize_Bpp20, 270](#)
- [PixelSize_Bpp24, 270](#)
- [PixelSize_Bpp30, 270](#)
- [PixelSize_Bpp32, 270](#)
- [PixelSize_Bpp36, 271](#)
- [PixelSize_Bpp4, 270](#)
- [PixelSize_Bpp48, 271](#)
- [PixelSize_Bpp64, 271](#)
- [PixelSize_Bpp8, 270](#)
- [PixelSize_Bpp96, 271](#)
- [RegionDestination_Stream0, 271](#)
- [RegionDestination_Stream1, 271](#)
- [RegionDestination_Stream2, 271](#)
- [RegionMode_Off, 271](#)
- [RegionMode_On, 271](#)
- [RegionSelector_All, 272](#)
- [RegionSelector_Region0, 272](#)
- [RegionSelector_Region1, 272](#)
- [RegionSelector_Region2, 272](#)
- [RgbTransformLightSource_Cloudy6500K, 272](#)
- [RgbTransformLightSource_CoolFluorescent4000K, 272](#)
- [RgbTransformLightSource_Custom, 272](#)
- [RgbTransformLightSource_Daylight5000K, 272](#)
- [RgbTransformLightSource_General, 272](#)
- [RgbTransformLightSource_Shade8000K, 272](#)
- [RgbTransformLightSource_Tungsten2800K, 272](#)
- [RgbTransformLightSource_WarmFluorescent3000K, 272](#)
- [Scan3dCoordinateReferenceSelector_RotationX, 273](#)
- [Scan3dCoordinateReferenceSelector_RotationY, 273](#)
- [Scan3dCoordinateReferenceSelector_RotationZ, 273](#)
- [Scan3dCoordinateReferenceSelector_TranslationX, 273](#)
- [Scan3dCoordinateReferenceSelector_TranslationY, 273](#)
- [Scan3dCoordinateReferenceSelector_TranslationZ, 273](#)
- [Scan3dCoordinateSelector_CoordinateA, 273](#)
- [Scan3dCoordinateSelector_CoordinateB, 273](#)
- [Scan3dCoordinateSelector_CoordinateC, 273](#)
- [Scan3dCoordinateSystem_Cartesian, 273](#)
- [Scan3dCoordinateSystem_Cylindrical, 273](#)
- [Scan3dCoordinateSystem_Spherical, 273](#)
- [Scan3dCoordinateSystemReference_Anchor, 274](#)
- [Scan3dCoordinateSystemReference_Transformed, 274](#)
- [Scan3dCoordinateTransformSelector_RotationX, 274](#)
- [Scan3dCoordinateTransformSelector_RotationY, 274](#)
- [Scan3dCoordinateTransformSelector_RotationZ, 274](#)
- [Scan3dCoordinateTransformSelector_TranslationX, 274](#)

- Scan3dCoordinateTransformSelector_TranslationY, [274](#)
- Scan3dCoordinateTransformSelector_TranslationZ, [274](#)
- Scan3dDistanceUnit_Inch, [274](#)
- Scan3dDistanceUnit_Millimeter, [274](#)
- Scan3dOutputMode_CalibratedABC_Grid, [276](#)
- Scan3dOutputMode_CalibratedABC_PointCloud, [276](#)
- Scan3dOutputMode_CalibratedAC, [276](#)
- Scan3dOutputMode_CalibratedAC_Linescan, [276](#)
- Scan3dOutputMode_CalibratedC, [276](#)
- Scan3dOutputMode_CalibratedC_Linescan, [276](#)
- Scan3dOutputMode_DisparityC, [276](#)
- Scan3dOutputMode_DisparityC_Linescan, [276](#)
- Scan3dOutputMode_RectifiedC, [276](#)
- Scan3dOutputMode_RectifiedC_Linescan, [276](#)
- Scan3dOutputMode_UncalibratedC, [276](#)
- SensorDigitizationTaps_Eight, [277](#)
- SensorDigitizationTaps_Four, [277](#)
- SensorDigitizationTaps_One, [276](#)
- SensorDigitizationTaps_Ten, [277](#)
- SensorDigitizationTaps_Three, [276](#)
- SensorDigitizationTaps_Two, [276](#)
- SensorShutterMode_Global, [277](#)
- SensorShutterMode_GlobalReset, [277](#)
- SensorShutterMode_Rolling, [277](#)
- SensorTaps_Eight, [277](#)
- SensorTaps_Four, [277](#)
- SensorTaps_One, [277](#)
- SensorTaps_Ten, [277](#)
- SensorTaps_Three, [277](#)
- SensorTaps_Two, [277](#)
- SequencerConfigurationMode_Off, [278](#)
- SequencerConfigurationMode_On, [278](#)
- SequencerConfigurationValid_No, [278](#)
- SequencerConfigurationValid_Yes, [278](#)
- SequencerMode_Off, [278](#)
- SequencerMode_On, [278](#)
- SequencerSetValid_No, [279](#)
- SequencerSetValid_Yes, [279](#)
- SequencerTriggerActivation_AnyEdge, [279](#)
- SequencerTriggerActivation_FallingEdge, [279](#)
- SequencerTriggerActivation_LevelHigh, [279](#)
- SequencerTriggerActivation_LevelLow, [279](#)
- SequencerTriggerActivation_RisingEdge, [279](#)
- SequencerTriggerSource_FrameStart, [279](#)
- SequencerTriggerSource_Off, [279](#)
- SerialPortBaudRate_Baud115200, [280](#)
- SerialPortBaudRate_Baud1200, [280](#)
- SerialPortBaudRate_Baud14400, [280](#)
- SerialPortBaudRate_Baud19200, [280](#)
- SerialPortBaudRate_Baud230400, [280](#)
- SerialPortBaudRate_Baud2400, [280](#)
- SerialPortBaudRate_Baud300, [280](#)
- SerialPortBaudRate_Baud38400, [280](#)
- SerialPortBaudRate_Baud460800, [280](#)
- SerialPortBaudRate_Baud4800, [280](#)
- SerialPortBaudRate_Baud57600, [280](#)
- SerialPortBaudRate_Baud600, [280](#)
- SerialPortBaudRate_Baud921600, [280](#)
- SerialPortBaudRate_Baud9600, [280](#)
- SerialPortParity_Even, [280](#)
- SerialPortParity_Mark, [280](#)
- SerialPortParity_None, [280](#)
- SerialPortParity_Odd, [280](#)
- SerialPortParity_Space, [280](#)
- SerialPortSelector_SerialPort0, [280](#)
- SerialPortSource_Line0, [281](#)
- SerialPortSource_Line1, [281](#)
- SerialPortSource_Line2, [281](#)
- SerialPortSource_Line3, [281](#)
- SerialPortSource_Off, [281](#)
- SerialPortStopBits_Bits1, [281](#)
- SerialPortStopBits_Bits1AndAHalf, [281](#)
- SerialPortStopBits_Bits2, [281](#)
- SoftwareSignalSelector_SoftwareSignal0, [281](#)
- SoftwareSignalSelector_SoftwareSignal1, [281](#)
- SoftwareSignalSelector_SoftwareSignal2, [281](#)
- SourceSelector_All, [282](#)
- SourceSelector_Source0, [282](#)
- SourceSelector_Source1, [282](#)
- SourceSelector_Source2, [282](#)
- spinAcquisitionModeEnums, [208](#)
- spinAcquisitionStatusSelectorEnums, [208](#)
- spinActionUnconditionalModeEnums, [208](#)
- spinAdcBitDepthEnums, [209](#)
- spinAutoAlgorithmSelectorEnums, [209](#)
- spinAutoExposureControlPriorityEnums, [209](#)
- spinAutoExposureLightingModeEnums, [210](#)
- spinAutoExposureMeteringModeEnums, [210](#)
- spinAutoExposureTargetGreyValueAutoEnums, [211](#)
- spinBalanceRatioSelectorEnums, [211](#)
- spinBalanceWhiteAutoEnums, [211](#)
- spinBalanceWhiteAutoProfileEnums, [212](#)
- spinBinningHorizontalModeEnums, [212](#)
- spinBinningSelectorEnums, [212](#)
- spinBinningVerticalModeEnums, [213](#)
- spinBlackLevelAutoBalanceEnums, [213](#)
- spinBlackLevelAutoEnums, [213](#)
- spinBlackLevelSelectorEnums, [214](#)
- spinChunkBlackLevelSelectorEnums, [214](#)
- spinChunkCounterSelectorEnums, [214](#)
- spinChunkEncoderSelectorEnums, [215](#)
- spinChunkEncoderStatusEnums, [215](#)
- spinChunkExposureTimeSelectorEnums, [215](#)
- spinChunkGainSelectorEnums, [216](#)
- spinChunkImageComponentEnums, [216](#)
- spinChunkPixelFormatEnums, [217](#)
- spinChunkRegionIDEnums, [217](#)
- spinChunkScan3dCoordinateReferenceSelectorEnums, [217](#)
- spinChunkScan3dCoordinateSelectorEnums, [218](#)
- spinChunkScan3dCoordinateSystemEnums, [218](#)

- spinChunkScan3dCoordinateSystemReferenceEnums, 218
- spinChunkScan3dCoordinateTransformSelectorEnums, 219
- spinChunkScan3dDistanceUnitEnums, 219
- spinChunkScan3dOutputModeEnums, 219
- spinChunkSelectorEnums, 220
- spinChunkSourceIDEnums, 221
- spinChunkTimerSelectorEnums, 221
- spinChunkTransferStreamIDEnums, 222
- spinCIConfigurationEnums, 222
- spinCITimeSlotsCountEnums, 222
- spinColorTransformationSelectorEnums, 223
- spinColorTransformationValueSelectorEnums, 223
- spinCompressionSaturationPriorityEnums, 224
- spinCounterEventActivationEnums, 224
- spinCounterEventSourceEnums, 224
- spinCounterResetActivationEnums, 225
- spinCounterResetSourceEnums, 225
- spinCounterSelectorEnums, 226
- spinCounterStatusEnums, 226
- spinCounterTriggerActivationEnums, 227
- spinCounterTriggerSourceEnums, 227
- spinCxpConnectionTestModeEnums, 228
- spinCxpLinkConfigurationEnums, 228
- spinCxpLinkConfigurationPreferredEnums, 229
- spinCxpLinkConfigurationStatusEnums, 230
- spinCxpPoCxpStatusEnums, 231
- spinDecimationHorizontalModeEnums, 231
- spinDecimationSelectorEnums, 231
- spinDecimationVerticalModeEnums, 232
- spinDefectCorrectionModeEnums, 232
- spinDeinterlacingEnums, 232
- spinDeviceCharacterSetEnums, 233
- spinDeviceClockSelectorEnums, 233
- spinDeviceConnectionStatusEnums, 233
- spinDeviceIndicatorModeEnums, 234
- spinDeviceLinkHeartbeatModeEnums, 234
- spinDeviceLinkThroughputLimitModeEnums, 234
- spinDevicePowerSupplySelectorEnums, 234
- spinDeviceRegistersEndiannessEnums, 235
- spinDeviceScanTypeEnums, 235
- spinDeviceSerialPortBaudRateEnums, 235
- spinDeviceSerialPortSelectorEnums, 236
- spinDeviceStreamChannelEndiannessEnums, 236
- spinDeviceStreamChannelTypeEnums, 236
- spinDeviceTapGeometryEnums, 237
- spinDeviceTemperatureSelectorEnums, 238
- spinDeviceTLTypeEnums, 238
- spinDeviceTypeEnums, 239
- spinEncoderModeEnums, 239
- spinEncoderOutputModeEnums, 239
- spinEncoderResetActivationEnums, 240
- spinEncoderResetSourceEnums, 240
- spinEncoderSelectorEnums, 241
- spinEncoderSourceAEnums, 242
- spinEncoderSourceBEnums, 242
- spinEncoderStatusEnums, 242
- spinEventNotificationEnums, 243
- spinEventSelectorEnums, 243
- spinExposureActiveModeEnums, 243
- spinExposureAutoEnums, 243
- spinExposureModeEnums, 244
- spinExposureTimeModeEnums, 244
- spinExposureTimeSelectorEnums, 245
- spinFileOpenModeEnums, 245
- spinFileOperationSelectorEnums, 245
- spinFileOperationStatusEnums, 246
- spinFileSelectorEnums, 246
- spinGainAutoBalanceEnums, 246
- spinGainAutoEnums, 248
- spinGainSelectorEnums, 248
- spinGevCCPEnums, 248
- spinGevCurrentPhysicalLinkConfigurationEnums, 249
- spinGevGVCPEExtendedStatusCodesSelectorEnums, 249
- spinGevGVSPExtendedIDModeEnums, 249
- spinGevIEEE1588ClockAccuracyEnums, 250
- spinGevIEEE1588ModeEnums, 250
- spinGevIEEE1588StatusEnums, 250
- spinGevIPConfigurationStatusEnums, 251
- spinGevPhysicalLinkConfigurationEnums, 251
- spinGevSupportedOptionSelectorEnums, 251
- spinImageComponentSelectorEnums, 252
- spinImageCompressionJPEGFormatOptionEnums, 253
- spinImageCompressionModeEnums, 253
- spinImageCompressionRateOptionEnums, 254
- spinLineFormatEnums, 254
- spinLineInputFilterSelectorEnums, 254
- spinLineModeEnums, 255
- spinLineSelectorEnums, 255
- spinLineSourceEnums, 255
- spinLogicBlockLUTInputActivationEnums, 256
- spinLogicBlockLUTInputSelectorEnums, 256
- spinLogicBlockLUTInputSourceEnums, 257
- spinLogicBlockLUTSelectorEnums, 257
- spinLogicBlockSelectorEnums, 258
- spinLUTSelectorEnums, 258
- spinPixelColorFilterEnums, 258
- spinPixelFormatEnums, 259
- spinPixelFormatInfoSelectorEnums, 264
- spinPixelSizeEnums, 270
- spinRegionDestinationEnums, 271
- spinRegionModeEnums, 271
- spinRegionSelectorEnums, 271
- spinRgbTransformLightSourceEnums, 272
- spinScan3dCoordinateReferenceSelectorEnums, 272
- spinScan3dCoordinateSelectorEnums, 273
- spinScan3dCoordinateSystemEnums, 273
- spinScan3dCoordinateSystemReferenceEnums, 273
- spinScan3dCoordinateTransformSelectorEnums, 274

spinScan3dDistanceUnitEnums, 274
spinScan3dOutputModeEnums, 274
spinSensorDigitizationTapsEnums, 276
spinSensorShutterModeEnums, 277
spinSensorTapsEnums, 277
spinSequencerConfigurationModeEnums, 277
spinSequencerConfigurationValidEnums, 278
spinSequencerModeEnums, 278
spinSequencerSetValidEnums, 278
spinSequencerTriggerActivationEnums, 279
spinSequencerTriggerSourceEnums, 279
spinSerialPortBaudRateEnums, 279
spinSerialPortParityEnums, 280
spinSerialPortSelectorEnums, 280
spinSerialPortSourceEnums, 281
spinSerialPortStopBitsEnums, 281
spinSoftwareSignalSelectorEnums, 281
spinSourceSelectorEnums, 282
spinTestPatternEnums, 282
spinTestPatternGeneratorSelectorEnums, 282
spinTimerSelectorEnums, 283
spinTimerStatusEnums, 283
spinTimerTriggerActivationEnums, 283
spinTimerTriggerSourceEnums, 284
spinTransferComponentSelectorEnums, 285
spinTransferControlModeEnums, 285
spinTransferOperationModeEnums, 286
spinTransferQueueModeEnums, 286
spinTransferSelectorEnums, 286
spinTransferStatusSelectorEnums, 287
spinTransferTriggerActivationEnums, 287
spinTransferTriggerModeEnums, 287
spinTransferTriggerSelectorEnums, 288
spinTransferTriggerSourceEnums, 288
spinTriggerActivationEnums, 289
spinTriggerModeEnums, 290
spinTriggerOverlapEnums, 290
spinTriggerSelectorEnums, 290
spinTriggerSourceEnums, 290
spinUserOutputSelectorEnums, 291
spinUserSetDefaultEnums, 291
spinUserSetSelectorEnums, 292
spinWhiteClipSelectorEnums, 292
TestPattern_Increment, 282
TestPattern_Off, 282
TestPattern_SensorTestPattern, 282
TestPatternGeneratorSelector_PipelineStart, 282
TestPatternGeneratorSelector_Sensor, 282
TimerSelector_Timer0, 283
TimerSelector_Timer1, 283
TimerSelector_Timer2, 283
TimerStatus_TimerActive, 283
TimerStatus_TimerCompleted, 283
TimerStatus_TimerIdle, 283
TimerStatus_TimerTriggerWait, 283
TimerTriggerActivation_AnyEdge, 283
TimerTriggerActivation_FallingEdge, 283
TimerTriggerActivation_LevelHigh, 283
TimerTriggerActivation_LevelLow, 283
TimerTriggerActivation_RisingEdge, 283
TimerTriggerSource_AcquisitionEnd, 284
TimerTriggerSource_AcquisitionStart, 284
TimerTriggerSource_AcquisitionTrigger, 284
TimerTriggerSource_Action0, 285
TimerTriggerSource_Action1, 285
TimerTriggerSource_Action2, 285
TimerTriggerSource_Counter0End, 284
TimerTriggerSource_Counter0Start, 284
TimerTriggerSource_Counter1End, 284
TimerTriggerSource_Counter1Start, 284
TimerTriggerSource_Counter2End, 284
TimerTriggerSource_Counter2Start, 284
TimerTriggerSource_Encoder0, 285
TimerTriggerSource_Encoder1, 285
TimerTriggerSource_Encoder2, 285
TimerTriggerSource_ExposureEnd, 284
TimerTriggerSource_ExposureStart, 284
TimerTriggerSource_FrameBurstEnd, 284
TimerTriggerSource_FrameBurstStart, 284
TimerTriggerSource_FrameEnd, 284
TimerTriggerSource_FrameStart, 284
TimerTriggerSource_FrameTrigger, 284
TimerTriggerSource_Line0, 284
TimerTriggerSource_Line1, 284
TimerTriggerSource_Line2, 284
TimerTriggerSource_LineEnd, 284
TimerTriggerSource_LineStart, 284
TimerTriggerSource_LineTrigger, 284
TimerTriggerSource_LinkTrigger0, 285
TimerTriggerSource_LinkTrigger1, 285
TimerTriggerSource_LinkTrigger2, 285
TimerTriggerSource_Off, 284
TimerTriggerSource_SoftwareSignal0, 285
TimerTriggerSource_SoftwareSignal1, 285
TimerTriggerSource_SoftwareSignal2, 285
TimerTriggerSource_Timer0End, 284
TimerTriggerSource_Timer0Start, 284
TimerTriggerSource_Timer1End, 284
TimerTriggerSource_Timer1Start, 284
TimerTriggerSource_Timer2End, 284
TimerTriggerSource_Timer2Start, 284
TimerTriggerSource_UserOutput0, 284
TimerTriggerSource_UserOutput1, 284
TimerTriggerSource_UserOutput2, 284
TransferComponentSelector_All, 285
TransferComponentSelector_Blue, 285
TransferComponentSelector_Green, 285
TransferComponentSelector_Red, 285
TransferControlMode_Automatic, 286
TransferControlMode_Basic, 286
TransferControlMode_UserControlled, 286
TransferOperationMode_Continuous, 286
TransferOperationMode_MultiBlock, 286
TransferQueueMode_FirstInFirstOut, 286
TransferSelector_All, 286
TransferSelector_Stream0, 286

- TransferSelector_Stream1, [286](#)
- TransferSelector_Stream2, [286](#)
- TransferStatusSelector_Paused, [287](#)
- TransferStatusSelector_QueueOverflow, [287](#)
- TransferStatusSelector_Stopped, [287](#)
- TransferStatusSelector_Stopping, [287](#)
- TransferStatusSelector_Streaming, [287](#)
- TransferTriggerActivation_AnyEdge, [287](#)
- TransferTriggerActivation_FallingEdge, [287](#)
- TransferTriggerActivation_LevelHigh, [287](#)
- TransferTriggerActivation_LevelLow, [287](#)
- TransferTriggerActivation_RisingEdge, [287](#)
- TransferTriggerMode_Off, [288](#)
- TransferTriggerMode_On, [288](#)
- TransferTriggerSelector_TransferAbort, [288](#)
- TransferTriggerSelector_TransferActive, [288](#)
- TransferTriggerSelector_TransferBurstStart, [288](#)
- TransferTriggerSelector_TransferBurstStop, [288](#)
- TransferTriggerSelector_TransferPause, [288](#)
- TransferTriggerSelector_TransferResume, [288](#)
- TransferTriggerSelector_TransferStart, [288](#)
- TransferTriggerSelector_TransferStop, [288](#)
- TransferTriggerSource_Action0, [289](#)
- TransferTriggerSource_Action1, [289](#)
- TransferTriggerSource_Action2, [289](#)
- TransferTriggerSource_Counter0End, [289](#)
- TransferTriggerSource_Counter0Start, [288](#)
- TransferTriggerSource_Counter1End, [289](#)
- TransferTriggerSource_Counter1Start, [288](#)
- TransferTriggerSource_Counter2End, [289](#)
- TransferTriggerSource_Counter2Start, [289](#)
- TransferTriggerSource_Line0, [288](#)
- TransferTriggerSource_Line1, [288](#)
- TransferTriggerSource_Line2, [288](#)
- TransferTriggerSource_SoftwareSignal0, [289](#)
- TransferTriggerSource_SoftwareSignal1, [289](#)
- TransferTriggerSource_SoftwareSignal2, [289](#)
- TransferTriggerSource_Timer0End, [289](#)
- TransferTriggerSource_Timer0Start, [289](#)
- TransferTriggerSource_Timer1End, [289](#)
- TransferTriggerSource_Timer1Start, [289](#)
- TransferTriggerSource_Timer2End, [289](#)
- TransferTriggerSource_Timer2Start, [289](#)
- TriggerActivation_AnyEdge, [289](#)
- TriggerActivation_FallingEdge, [289](#)
- TriggerActivation_LevelHigh, [289](#)
- TriggerActivation_LevelLow, [289](#)
- TriggerActivation_RisingEdge, [289](#)
- TriggerMode_Off, [290](#)
- TriggerMode_On, [290](#)
- TriggerOverlap_Off, [290](#)
- TriggerOverlap_PreviousFrame, [290](#)
- TriggerOverlap_ReadOut, [290](#)
- TriggerSelector_AcquisitionStart, [290](#)
- TriggerSelector_FrameBurstStart, [290](#)
- TriggerSelector_FrameStart, [290](#)
- TriggerSource_Action0, [291](#)
- TriggerSource_Counter0End, [291](#)
- TriggerSource_Counter0Start, [291](#)
- TriggerSource_Counter1End, [291](#)
- TriggerSource_Counter1Start, [291](#)
- TriggerSource_Line0, [291](#)
- TriggerSource_Line1, [291](#)
- TriggerSource_Line2, [291](#)
- TriggerSource_Line3, [291](#)
- TriggerSource_LogicBlock0, [291](#)
- TriggerSource_LogicBlock1, [291](#)
- TriggerSource_Software, [291](#)
- TriggerSource_UserOutput0, [291](#)
- TriggerSource_UserOutput1, [291](#)
- TriggerSource_UserOutput2, [291](#)
- TriggerSource_UserOutput3, [291](#)
- UNKNOWN_PIXELFORMAT, [264](#)
- UserOutputSelector_UserOutput0, [291](#)
- UserOutputSelector_UserOutput1, [291](#)
- UserOutputSelector_UserOutput2, [291](#)
- UserOutputSelector_UserOutput3, [291](#)
- UserSetDefault_Default, [292](#)
- UserSetDefault_UserSet0, [292](#)
- UserSetDefault_UserSet1, [292](#)
- UserSetSelector_Default, [292](#)
- UserSetSelector_UserSet0, [292](#)
- UserSetSelector_UserSet1, [292](#)
- WhiteClipSelector_All, [292](#)
- WhiteClipSelector_Blue, [292](#)
- WhiteClipSelector_Green, [292](#)
- WhiteClipSelector_Red, [292](#)
- WhiteClipSelector_Tap1, [292](#)
- WhiteClipSelector_Tap2, [292](#)
- WhiteClipSelector_U, [292](#)
- WhiteClipSelector_V, [292](#)
- WhiteClipSelector_Y, [292](#)
- CameraList Access, [24](#)
- CategoryNode
 - SpinnakerGenApiDefsC.h, [468](#)
- Chunk data access, [28](#)
- Chunk Data Structures, [22](#)
- ChunkBlackLevel
 - quickSpin, [55](#)
- ChunkBlackLevelSelector
 - quickSpin, [56](#)
- ChunkBlackLevelSelector_All
 - CameraDefsC.h, [214](#)
- ChunkCompressionMode
 - quickSpin, [56](#)
- ChunkCompressionRatio
 - quickSpin, [56](#)
- ChunkCounterSelector
 - quickSpin, [56](#)
- ChunkCounterSelector_Counter0
 - CameraDefsC.h, [215](#)
- ChunkCounterSelector_Counter1
 - CameraDefsC.h, [215](#)
- ChunkCounterSelector_Counter2
 - CameraDefsC.h, [215](#)
- ChunkCounterValue

- quickSpin, [56](#)
- ChunkCRC
 - quickSpin, [56](#)
- ChunkEnable
 - quickSpin, [56](#)
- ChunkEncoderSelector
 - quickSpin, [56](#)
- ChunkEncoderSelector_Encoder0
 - CameraDefsC.h, [215](#)
- ChunkEncoderSelector_Encoder1
 - CameraDefsC.h, [215](#)
- ChunkEncoderSelector_Encoder2
 - CameraDefsC.h, [215](#)
- ChunkEncoderStatus
 - quickSpin, [57](#)
- ChunkEncoderStatus_EncoderDown
 - CameraDefsC.h, [215](#)
- ChunkEncoderStatus_EncoderIdle
 - CameraDefsC.h, [215](#)
- ChunkEncoderStatus_EncoderStatic
 - CameraDefsC.h, [215](#)
- ChunkEncoderStatus_EncoderUp
 - CameraDefsC.h, [215](#)
- ChunkEncoderValue
 - quickSpin, [57](#)
- ChunkExposureEndLineStatusAll
 - quickSpin, [57](#)
- ChunkExposureTime
 - quickSpin, [57](#)
- ChunkExposureTimeSelector
 - quickSpin, [57](#)
- ChunkExposureTimeSelector_Blue
 - CameraDefsC.h, [216](#)
- ChunkExposureTimeSelector_Common
 - CameraDefsC.h, [216](#)
- ChunkExposureTimeSelector_Cyan
 - CameraDefsC.h, [216](#)
- ChunkExposureTimeSelector_Green
 - CameraDefsC.h, [216](#)
- ChunkExposureTimeSelector_Infrared
 - CameraDefsC.h, [216](#)
- ChunkExposureTimeSelector_Magenta
 - CameraDefsC.h, [216](#)
- ChunkExposureTimeSelector_Red
 - CameraDefsC.h, [216](#)
- ChunkExposureTimeSelector_Stage1
 - CameraDefsC.h, [216](#)
- ChunkExposureTimeSelector_Stage2
 - CameraDefsC.h, [216](#)
- ChunkExposureTimeSelector_Ultraviolet
 - CameraDefsC.h, [216](#)
- ChunkExposureTimeSelector_Yellow
 - CameraDefsC.h, [216](#)
- ChunkFrameID
 - quickSpin, [57](#)
- ChunkGain
 - quickSpin, [57](#)
- ChunkGainSelector
 - quickSpin, [57](#)
- ChunkGainSelector_All
 - CameraDefsC.h, [216](#)
- ChunkGainSelector_Blue
 - CameraDefsC.h, [216](#)
- ChunkGainSelector_Green
 - CameraDefsC.h, [216](#)
- ChunkGainSelector_Red
 - CameraDefsC.h, [216](#)
- ChunkHeight
 - quickSpin, [58](#)
- ChunkImage
 - quickSpin, [58](#)
- ChunkImageComponent
 - quickSpin, [58](#)
- ChunkImageComponent_Color
 - CameraDefsC.h, [216](#)
- ChunkImageComponent_Confidence
 - CameraDefsC.h, [217](#)
- ChunkImageComponent_Disparity
 - CameraDefsC.h, [216](#)
- ChunkImageComponent_Infrared
 - CameraDefsC.h, [216](#)
- ChunkImageComponent_Intensity
 - CameraDefsC.h, [216](#)
- ChunkImageComponent_Range
 - CameraDefsC.h, [216](#)
- ChunkImageComponent_Scatter
 - CameraDefsC.h, [217](#)
- ChunkImageComponent_Ultraviolet
 - CameraDefsC.h, [216](#)
- ChunkInferenceBoundingBoxResult
 - quickSpin, [58](#)
- ChunkInferenceConfidence
 - quickSpin, [58](#)
- ChunkInferenceFrameID
 - quickSpin, [58](#)
- ChunkInferenceResult
 - quickSpin, [58](#)
- ChunkLinePitch
 - quickSpin, [58](#)
- ChunkLineStatusAll
 - quickSpin, [59](#)
- ChunkModeActive
 - quickSpin, [59](#)
- ChunkOffsetX
 - quickSpin, [59](#)
- ChunkOffsetY
 - quickSpin, [59](#)
- ChunkPartSelector
 - quickSpin, [59](#)
- ChunkPixelDynamicRangeMax
 - quickSpin, [59](#)
- ChunkPixelDynamicRangeMin
 - quickSpin, [59](#)
- ChunkPixelFormat
 - quickSpin, [59](#)
- ChunkPixelFormat_BayerBG8

- CameraDefsC.h, [217](#)
- ChunkPixelFormat_BayerGB8
 - CameraDefsC.h, [217](#)
- ChunkPixelFormat_BayerGR8
 - CameraDefsC.h, [217](#)
- ChunkPixelFormat_BayerRG8
 - CameraDefsC.h, [217](#)
- ChunkPixelFormat_Mono12Packed
 - CameraDefsC.h, [217](#)
- ChunkPixelFormat_Mono16
 - CameraDefsC.h, [217](#)
- ChunkPixelFormat_Mono8
 - CameraDefsC.h, [217](#)
- ChunkPixelFormat_RGB8Packed
 - CameraDefsC.h, [217](#)
- ChunkPixelFormat_YCbCr601_422_8_CbYCrY
 - CameraDefsC.h, [217](#)
- ChunkPixelFormat_YUV422Packed
 - CameraDefsC.h, [217](#)
- ChunkRegionID
 - quickSpin, [60](#)
- ChunkRegionID_Region0
 - CameraDefsC.h, [217](#)
- ChunkRegionID_Region1
 - CameraDefsC.h, [217](#)
- ChunkRegionID_Region2
 - CameraDefsC.h, [217](#)
- ChunkScan3dAxisMax
 - quickSpin, [60](#)
- ChunkScan3dAxisMin
 - quickSpin, [60](#)
- ChunkScan3dCoordinateOffset
 - quickSpin, [60](#)
- ChunkScan3dCoordinateReferenceSelector
 - quickSpin, [60](#)
- ChunkScan3dCoordinateReferenceSelector_RotationX
 - CameraDefsC.h, [218](#)
- ChunkScan3dCoordinateReferenceSelector_RotationY
 - CameraDefsC.h, [218](#)
- ChunkScan3dCoordinateReferenceSelector_RotationZ
 - CameraDefsC.h, [218](#)
- ChunkScan3dCoordinateReferenceSelector_TranslationX
 - CameraDefsC.h, [218](#)
- ChunkScan3dCoordinateReferenceSelector_TranslationY
 - CameraDefsC.h, [218](#)
- ChunkScan3dCoordinateReferenceSelector_TranslationZ
 - CameraDefsC.h, [218](#)
- ChunkScan3dCoordinateReferenceValue
 - quickSpin, [60](#)
- ChunkScan3dCoordinateScale
 - quickSpin, [60](#)
- ChunkScan3dCoordinateSelector
 - quickSpin, [60](#)
- ChunkScan3dCoordinateSelector_CoordinateA
 - CameraDefsC.h, [218](#)
- ChunkScan3dCoordinateSelector_CoordinateB
 - CameraDefsC.h, [218](#)
- ChunkScan3dCoordinateSelector_CoordinateC
 - CameraDefsC.h, [218](#)
- ChunkScan3dCoordinateSystem
 - quickSpin, [61](#)
- ChunkScan3dCoordinateSystem_Cartesian
 - CameraDefsC.h, [218](#)
- ChunkScan3dCoordinateSystem_Cylindrical
 - CameraDefsC.h, [218](#)
- ChunkScan3dCoordinateSystem_Spherical
 - CameraDefsC.h, [218](#)
- ChunkScan3dCoordinateSystemReference
 - quickSpin, [61](#)
- ChunkScan3dCoordinateSystemReference_Anchor
 - CameraDefsC.h, [219](#)
- ChunkScan3dCoordinateSystemReference_Transformed
 - CameraDefsC.h, [219](#)
- ChunkScan3dCoordinateTransformSelector
 - quickSpin, [61](#)
- ChunkScan3dCoordinateTransformSelector_RotationX
 - CameraDefsC.h, [219](#)
- ChunkScan3dCoordinateTransformSelector_RotationY
 - CameraDefsC.h, [219](#)
- ChunkScan3dCoordinateTransformSelector_RotationZ
 - CameraDefsC.h, [219](#)
- ChunkScan3dCoordinateTransformSelector_TranslationX
 - CameraDefsC.h, [219](#)
- ChunkScan3dCoordinateTransformSelector_TranslationY
 - CameraDefsC.h, [219](#)
- ChunkScan3dCoordinateTransformSelector_TranslationZ
 - CameraDefsC.h, [219](#)
- ChunkScan3dDistanceUnit
 - quickSpin, [61](#)
- ChunkScan3dDistanceUnit_Inch
 - CameraDefsC.h, [219](#)
- ChunkScan3dDistanceUnit_Millimeter
 - CameraDefsC.h, [219](#)
- ChunkScan3dInvalidDataFlag
 - quickSpin, [61](#)
- ChunkScan3dInvalidDataValue
 - quickSpin, [61](#)
- ChunkScan3dOutputMode
 - quickSpin, [61](#)
- ChunkScan3dOutputMode_CalibratedABC_Grid
 - CameraDefsC.h, [220](#)
- ChunkScan3dOutputMode_CalibratedABC_PointCloud
 - CameraDefsC.h, [220](#)
- ChunkScan3dOutputMode_CalibratedAC
 - CameraDefsC.h, [220](#)
- ChunkScan3dOutputMode_CalibratedAC_Linescan
 - CameraDefsC.h, [220](#)
- ChunkScan3dOutputMode_CalibratedC
 - CameraDefsC.h, [220](#)
- ChunkScan3dOutputMode_CalibratedC_Linescan
 - CameraDefsC.h, [220](#)
- ChunkScan3dOutputMode_DisparityC
 - CameraDefsC.h, [220](#)
- ChunkScan3dOutputMode_DisparityC_Linescan
 - CameraDefsC.h, [220](#)
- ChunkScan3dOutputMode_RectifiedC

- CameraDefsC.h, [220](#)
- ChunkScan3dOutputMode_RectifiedC_Linescan
 - CameraDefsC.h, [220](#)
- ChunkScan3dOutputMode_UncalibratedC
 - CameraDefsC.h, [220](#)
- ChunkScan3dTransformValue
 - quickSpin, [61](#)
- ChunkScanLineSelector
 - quickSpin, [62](#)
- ChunkSelector
 - quickSpin, [62](#)
- ChunkSelector_BlackLevel
 - CameraDefsC.h, [221](#)
- ChunkSelector_CRC
 - CameraDefsC.h, [221](#)
- ChunkSelector_ExposureEndLineStatusAll
 - CameraDefsC.h, [221](#)
- ChunkSelector_ExposureTime
 - CameraDefsC.h, [221](#)
- ChunkSelector_FrameID
 - CameraDefsC.h, [221](#)
- ChunkSelector_Gain
 - CameraDefsC.h, [221](#)
- ChunkSelector_Height
 - CameraDefsC.h, [221](#)
- ChunkSelector_Image
 - CameraDefsC.h, [221](#)
- ChunkSelector_OffsetX
 - CameraDefsC.h, [221](#)
- ChunkSelector_OffsetY
 - CameraDefsC.h, [221](#)
- ChunkSelector_PixelFormat
 - CameraDefsC.h, [221](#)
- ChunkSelector_SequencerSetActive
 - CameraDefsC.h, [221](#)
- ChunkSelector_SerialData
 - CameraDefsC.h, [221](#)
- ChunkSelector_Timestamp
 - CameraDefsC.h, [221](#)
- ChunkSelector_Width
 - CameraDefsC.h, [221](#)
- ChunkSequencerSetActive
 - quickSpin, [62](#)
- ChunkSerialData
 - quickSpin, [62](#)
- ChunkSerialDataLength
 - quickSpin, [62](#)
- ChunkSerialReceiveOverflow
 - quickSpin, [62](#)
- ChunkSourceID
 - quickSpin, [62](#)
- ChunkSourceID_Source0
 - CameraDefsC.h, [221](#)
- ChunkSourceID_Source1
 - CameraDefsC.h, [221](#)
- ChunkSourceID_Source2
 - CameraDefsC.h, [221](#)
- ChunkStreamChannelID
 - quickSpin, [62](#)
- ChunkTimerSelector
 - quickSpin, [63](#)
- ChunkTimerSelector_Timer0
 - CameraDefsC.h, [222](#)
- ChunkTimerSelector_Timer1
 - CameraDefsC.h, [222](#)
- ChunkTimerSelector_Timer2
 - CameraDefsC.h, [222](#)
- ChunkTimerValue
 - quickSpin, [63](#)
- ChunkTimestamp
 - quickSpin, [63](#)
- ChunkTimestampLatchValue
 - quickSpin, [63](#)
- ChunkTransferBlockID
 - quickSpin, [63](#)
- ChunkTransferQueueCurrentBlockCount
 - quickSpin, [63](#)
- ChunkTransferStreamID
 - quickSpin, [63](#)
- ChunkTransferStreamID_Stream0
 - CameraDefsC.h, [222](#)
- ChunkTransferStreamID_Stream1
 - CameraDefsC.h, [222](#)
- ChunkTransferStreamID_Stream2
 - CameraDefsC.h, [222](#)
- ChunkTransferStreamID_Stream3
 - CameraDefsC.h, [222](#)
- ChunkWidth
 - quickSpin, [63](#)
- CL
 - SpinnakerGenApiDefsC.h, [469](#)
- CIConfiguration
 - quickSpin, [64](#)
- CIConfiguration_Base
 - CameraDefsC.h, [222](#)
- CIConfiguration_DualBase
 - CameraDefsC.h, [222](#)
- CIConfiguration_EightyBit
 - CameraDefsC.h, [222](#)
- CIConfiguration_Full
 - CameraDefsC.h, [222](#)
- CIConfiguration_Medium
 - CameraDefsC.h, [222](#)
- CITimeSlotsCount
 - quickSpin, [64](#)
- CITimeSlotsCount_One
 - CameraDefsC.h, [223](#)
- CITimeSlotsCount_Three
 - CameraDefsC.h, [223](#)
- CITimeSlotsCount_Two
 - CameraDefsC.h, [223](#)
- ColorTransformationEnable
 - quickSpin, [64](#)
- ColorTransformationSelector
 - quickSpin, [64](#)
- ColorTransformationSelector_RGBtoRGB

- CameraDefsC.h, [223](#)
- ColorTransformationSelector_RGBtoYUV
 - CameraDefsC.h, [223](#)
- ColorTransformationValue
 - quickSpin, [64](#)
- ColorTransformationValueSelector
 - quickSpin, [64](#)
- ColorTransformationValueSelector_Gain00
 - CameraDefsC.h, [223](#)
- ColorTransformationValueSelector_Gain01
 - CameraDefsC.h, [223](#)
- ColorTransformationValueSelector_Gain02
 - CameraDefsC.h, [223](#)
- ColorTransformationValueSelector_Gain10
 - CameraDefsC.h, [223](#)
- ColorTransformationValueSelector_Gain11
 - CameraDefsC.h, [223](#)
- ColorTransformationValueSelector_Gain12
 - CameraDefsC.h, [223](#)
- ColorTransformationValueSelector_Gain20
 - CameraDefsC.h, [223](#)
- ColorTransformationValueSelector_Gain21
 - CameraDefsC.h, [223](#)
- ColorTransformationValueSelector_Gain22
 - CameraDefsC.h, [223](#)
- ColorTransformationValueSelector_Offset0
 - CameraDefsC.h, [223](#)
- ColorTransformationValueSelector_Offset1
 - CameraDefsC.h, [223](#)
- ColorTransformationValueSelector_Offset2
 - CameraDefsC.h, [223](#)
- CommandNode
 - SpinnakerGenApiDefsC.h, [467](#)
- compression
 - spinTIFFOption, [173](#)
- compressionLevel
 - spinPNGOption, [171](#)
- CompressionRatio
 - quickSpin, [64](#)
- CompressionSaturationPriority
 - quickSpin, [64](#)
- CompressionSaturationPriority_DropFrame
 - CameraDefsC.h, [224](#)
- CompressionSaturationPriority_ReduceFrameRate
 - CameraDefsC.h, [224](#)
- CounterDelay
 - quickSpin, [65](#)
- CounterDuration
 - quickSpin, [65](#)
- CounterEventActivation
 - quickSpin, [65](#)
- CounterEventActivation_AnyEdge
 - CameraDefsC.h, [224](#)
- CounterEventActivation_FallingEdge
 - CameraDefsC.h, [224](#)
- CounterEventActivation_LevelHigh
 - CameraDefsC.h, [224](#)
- CounterEventActivation_LevelLow
 - CameraDefsC.h, [224](#)
- CounterEventActivation_RisingEdge
 - CameraDefsC.h, [224](#)
- CounterEventSource
 - quickSpin, [65](#)
- CounterEventSource_Counter0End
 - CameraDefsC.h, [225](#)
- CounterEventSource_Counter0Start
 - CameraDefsC.h, [225](#)
- CounterEventSource_Counter1End
 - CameraDefsC.h, [225](#)
- CounterEventSource_Counter1Start
 - CameraDefsC.h, [225](#)
- CounterEventSource_ExposureEnd
 - CameraDefsC.h, [225](#)
- CounterEventSource_ExposureStart
 - CameraDefsC.h, [225](#)
- CounterEventSource_FrameTriggerWait
 - CameraDefsC.h, [225](#)
- CounterEventSource_Line0
 - CameraDefsC.h, [224](#)
- CounterEventSource_Line1
 - CameraDefsC.h, [225](#)
- CounterEventSource_Line2
 - CameraDefsC.h, [225](#)
- CounterEventSource_Line3
 - CameraDefsC.h, [225](#)
- CounterEventSource_LogicBlock0
 - CameraDefsC.h, [225](#)
- CounterEventSource_LogicBlock1
 - CameraDefsC.h, [225](#)
- CounterEventSource_MHzTick
 - CameraDefsC.h, [224](#)
- CounterEventSource_Off
 - CameraDefsC.h, [224](#)
- CounterEventSource_UserOutput0
 - CameraDefsC.h, [225](#)
- CounterEventSource_UserOutput1
 - CameraDefsC.h, [225](#)
- CounterEventSource_UserOutput2
 - CameraDefsC.h, [225](#)
- CounterEventSource_UserOutput3
 - CameraDefsC.h, [225](#)
- CounterReset
 - quickSpin, [65](#)
- CounterResetActivation
 - quickSpin, [65](#)
- CounterResetActivation_AnyEdge
 - CameraDefsC.h, [225](#)
- CounterResetActivation_FallingEdge
 - CameraDefsC.h, [225](#)
- CounterResetActivation_LevelHigh
 - CameraDefsC.h, [225](#)
- CounterResetActivation_LevelLow
 - CameraDefsC.h, [225](#)
- CounterResetActivation_RisingEdge
 - CameraDefsC.h, [225](#)
- CounterResetSource

- quickSpin, [65](#)
- CounterResetSource_Counter0End
 - CameraDefsC.h, [226](#)
- CounterResetSource_Counter0Start
 - CameraDefsC.h, [226](#)
- CounterResetSource_Counter1End
 - CameraDefsC.h, [226](#)
- CounterResetSource_Counter1Start
 - CameraDefsC.h, [226](#)
- CounterResetSource_ExposureEnd
 - CameraDefsC.h, [226](#)
- CounterResetSource_ExposureStart
 - CameraDefsC.h, [226](#)
- CounterResetSource_FrameTriggerWait
 - CameraDefsC.h, [226](#)
- CounterResetSource_Line0
 - CameraDefsC.h, [226](#)
- CounterResetSource_Line1
 - CameraDefsC.h, [226](#)
- CounterResetSource_Line2
 - CameraDefsC.h, [226](#)
- CounterResetSource_Line3
 - CameraDefsC.h, [226](#)
- CounterResetSource_LogicBlock0
 - CameraDefsC.h, [226](#)
- CounterResetSource_LogicBlock1
 - CameraDefsC.h, [226](#)
- CounterResetSource_Off
 - CameraDefsC.h, [225](#)
- CounterResetSource_UserOutput0
 - CameraDefsC.h, [226](#)
- CounterResetSource_UserOutput1
 - CameraDefsC.h, [226](#)
- CounterResetSource_UserOutput2
 - CameraDefsC.h, [226](#)
- CounterResetSource_UserOutput3
 - CameraDefsC.h, [226](#)
- CounterSelector
 - quickSpin, [65](#)
- CounterSelector_Counter0
 - CameraDefsC.h, [226](#)
- CounterSelector_Counter1
 - CameraDefsC.h, [226](#)
- CounterStatus
 - quickSpin, [66](#)
- CounterStatus_CounterActive
 - CameraDefsC.h, [226](#)
- CounterStatus_CounterCompleted
 - CameraDefsC.h, [226](#)
- CounterStatus_CounterIdle
 - CameraDefsC.h, [226](#)
- CounterStatus_CounterOverflow
 - CameraDefsC.h, [226](#)
- CounterStatus_CounterTriggerWait
 - CameraDefsC.h, [226](#)
- CounterTriggerActivation
 - quickSpin, [66](#)
- CounterTriggerActivation_AnyEdge
 - CameraDefsC.h, [227](#)
- CounterTriggerActivation_FallingEdge
 - CameraDefsC.h, [227](#)
- CounterTriggerActivation_LevelHigh
 - CameraDefsC.h, [227](#)
- CounterTriggerActivation_LevelLow
 - CameraDefsC.h, [227](#)
- CounterTriggerActivation_RisingEdge
 - CameraDefsC.h, [227](#)
- CounterTriggerSource
 - quickSpin, [66](#)
- CounterTriggerSource_Counter0End
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_Counter0Start
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_Counter1End
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_Counter1Start
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_ExposureEnd
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_ExposureStart
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_FrameTriggerWait
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_Line0
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_Line1
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_Line2
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_Line3
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_LogicBlock0
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_LogicBlock1
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_Off
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_UserOutput0
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_UserOutput1
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_UserOutput2
 - CameraDefsC.h, [227](#)
- CounterTriggerSource_UserOutput3
 - CameraDefsC.h, [227](#)
- CounterValue
 - quickSpin, [66](#)
- CounterValueAtReset
 - quickSpin, [66](#)
- ctAllDependingNodes
 - SpinnakerGenApiDefsC.h, [467](#)
- ctAllTerminalNodes
 - SpinnakerGenApiDefsC.h, [467](#)
- ctDependingChildren
 - SpinnakerGenApiDefsC.h, [467](#)
- ctInvalidators

- SpinnakerGenApiDefsC.h, [467](#)
- ctReadingChildren
 - SpinnakerGenApiDefsC.h, [467](#)
- ctWritingChildren
 - SpinnakerGenApiDefsC.h, [467](#)
- Custom
 - SpinnakerGenApiDefsC.h, [467](#)
- CxpConnectionSelector
 - quickSpin, [66](#)
- CxpConnectionTestErrorCount
 - quickSpin, [66](#)
- CxpConnectionTestMode
 - quickSpin, [66](#)
- CxpConnectionTestMode_Mode1
 - CameraDefsC.h, [228](#)
- CxpConnectionTestMode_Off
 - CameraDefsC.h, [228](#)
- CxpConnectionTestPacketCount
 - quickSpin, [67](#)
- CxpLinkConfiguration
 - quickSpin, [67](#)
- CxpLinkConfiguration_Auto
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP1_X1
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP1_X2
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP1_X3
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP1_X4
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP1_X5
 - CameraDefsC.h, [229](#)
- CxpLinkConfiguration_CXP1_X6
 - CameraDefsC.h, [229](#)
- CxpLinkConfiguration_CXP2_X1
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP2_X2
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP2_X3
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP2_X4
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP2_X5
 - CameraDefsC.h, [229](#)
- CxpLinkConfiguration_CXP2_X6
 - CameraDefsC.h, [229](#)
- CxpLinkConfiguration_CXP3_X1
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP3_X2
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP3_X3
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP3_X4
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP3_X5
 - CameraDefsC.h, [229](#)
- CxpLinkConfiguration_CXP3_X6
 - CameraDefsC.h, [229](#)
- CameraDefsC.h, [229](#)
- CxpLinkConfiguration_CXP5_X1
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP5_X2
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP5_X3
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP5_X4
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP5_X5
 - CameraDefsC.h, [229](#)
- CxpLinkConfiguration_CXP5_X6
 - CameraDefsC.h, [229](#)
- CxpLinkConfiguration_CXP6_X1
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP6_X2
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP6_X3
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP6_X4
 - CameraDefsC.h, [228](#)
- CxpLinkConfiguration_CXP6_X5
 - CameraDefsC.h, [229](#)
- CxpLinkConfiguration_CXP6_X6
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred
 - quickSpin, [67](#)
- CxpLinkConfigurationPreferred_CXP1_X1
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP1_X2
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP1_X3
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP1_X4
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP1_X5
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP1_X6
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationPreferred_CXP2_X1
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP2_X2
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP2_X3
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP2_X4
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP2_X5
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP2_X6
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationPreferred_CXP3_X1
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP3_X2
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP3_X3
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP3_X4

- CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP3_X5
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP3_X6
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationPreferred_CXP5_X1
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP5_X2
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP5_X3
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP5_X4
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP5_X5
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP5_X6
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationPreferred_CXP6_X1
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP6_X2
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP6_X3
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP6_X4
 - CameraDefsC.h, [229](#)
- CxpLinkConfigurationPreferred_CXP6_X5
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationPreferred_CXP6_X6
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus
 - quickSpin, [67](#)
- CxpLinkConfigurationStatus_CXP1_X1
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP1_X2
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP1_X3
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP1_X4
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP1_X5
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP1_X6
 - CameraDefsC.h, [231](#)
- CxpLinkConfigurationStatus_CXP2_X1
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP2_X2
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP2_X3
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP2_X4
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP2_X5
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP2_X6
 - CameraDefsC.h, [231](#)
- CxpLinkConfigurationStatus_CXP3_X1
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP3_X2
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP3_X3
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP3_X4
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP3_X5
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP3_X6
 - CameraDefsC.h, [231](#)
- CxpLinkConfigurationStatus_CXP5_X1
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP5_X2
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP5_X3
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP5_X4
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP5_X5
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP5_X6
 - CameraDefsC.h, [231](#)
- CxpLinkConfigurationStatus_CXP6_X1
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP6_X2
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP6_X3
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP6_X4
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_CXP6_X5
 - CameraDefsC.h, [231](#)
- CxpLinkConfigurationStatus_CXP6_X6
 - CameraDefsC.h, [231](#)
- CxpLinkConfigurationStatus_None
 - CameraDefsC.h, [230](#)
- CxpLinkConfigurationStatus_Pending
 - CameraDefsC.h, [230](#)
- CxpPoCxpAuto
 - quickSpin, [67](#)
- CxpPoCxpStatus
 - quickSpin, [67](#)
- CxpPoCxpStatus_Auto
 - CameraDefsC.h, [231](#)
- CxpPoCxpStatus_Off
 - CameraDefsC.h, [231](#)
- CxpPoCxpStatus_Tripped
 - CameraDefsC.h, [231](#)
- CxpPoCxpTripReset
 - quickSpin, [67](#)
- CxpPoCxpTurnOff
 - quickSpin, [67](#)
- DecimationHorizontal
 - quickSpin, [68](#)
- DecimationHorizontalMode
 - quickSpin, [68](#)
- DecimationHorizontalMode_Discard
 - CameraDefsC.h, [231](#)
- DecimationSelector

- quickSpin, [68](#)
- DecimationSelector_All
 - CameraDefsC.h, [232](#)
- DecimationSelector_Sensor
 - CameraDefsC.h, [232](#)
- DecimationVertical
 - quickSpin, [68](#)
- DecimationVerticalMode
 - quickSpin, [68](#)
- DecimationVerticalMode_Discard
 - CameraDefsC.h, [232](#)
- Decreasing
 - SpinnakerGenApiDefsC.h, [469](#)
- DefectCorrectionMode
 - quickSpin, [68](#)
- DefectCorrectionMode_Average
 - CameraDefsC.h, [232](#)
- DefectCorrectionMode_Highlight
 - CameraDefsC.h, [232](#)
- DefectCorrectionMode_Zero
 - CameraDefsC.h, [232](#)
- DefectCorrectStaticEnable
 - quickSpin, [68](#)
- DefectTableApply
 - quickSpin, [68](#)
- DefectTableCoordinateX
 - quickSpin, [69](#)
- DefectTableCoordinateY
 - quickSpin, [69](#)
- DefectTableFactoryRestore
 - quickSpin, [69](#)
- DefectTableIndex
 - quickSpin, [69](#)
- DefectTablePixelCount
 - quickSpin, [69](#)
- DefectTableSave
 - quickSpin, [69](#)
- Deinterlacing
 - quickSpin, [69](#)
- Deinterlacing_LineDuplication
 - CameraDefsC.h, [233](#)
- Deinterlacing_Off
 - CameraDefsC.h, [233](#)
- Deinterlacing_Weave
 - CameraDefsC.h, [233](#)
- Device Event Data Access, [28](#)
- DeviceAccessStatus
 - quickSpinTLDevice, [133](#)
 - quickSpinTLInterface, [140](#)
- DeviceAccessStatus_Busy
 - TransportLayerDefsC.h, [476](#)
- DeviceAccessStatus_NoAccess
 - TransportLayerDefsC.h, [476](#)
- DeviceAccessStatus_OpenReadOnly
 - TransportLayerDefsC.h, [476](#)
- DeviceAccessStatus_OpenReadWrite
 - TransportLayerDefsC.h, [476](#)
- DeviceAccessStatus_ReadOnly
 - TransportLayerDefsC.h, [476](#)
- DeviceAccessStatus_ReadWrite
 - TransportLayerDefsC.h, [476](#)
- DeviceAccessStatus_Unknown
 - TransportLayerDefsC.h, [476](#)
- DeviceAddress
 - actionCommandResult, [35](#)
- DeviceBootloaderVersion
 - quickSpinTLDevice, [134](#)
- DeviceCharacterSet
 - quickSpin, [69](#)
- DeviceCharacterSet_ASCII
 - CameraDefsC.h, [233](#)
- DeviceCharacterSet_UTF8
 - CameraDefsC.h, [233](#)
- DeviceClockFrequency
 - quickSpin, [70](#)
- DeviceClockSelector
 - quickSpin, [70](#)
- DeviceClockSelector_CameraLink
 - CameraDefsC.h, [233](#)
- DeviceClockSelector_Sensor
 - CameraDefsC.h, [233](#)
- DeviceClockSelector_SensorDigitization
 - CameraDefsC.h, [233](#)
- DeviceConnectionSelector
 - quickSpin, [70](#)
- DeviceConnectionSpeed
 - quickSpin, [70](#)
- DeviceConnectionStatus
 - quickSpin, [70](#)
- DeviceConnectionStatus_Active
 - CameraDefsC.h, [233](#)
- DeviceConnectionStatus_Inactive
 - CameraDefsC.h, [233](#)
- DeviceCount
 - quickSpinTLInterface, [140](#)
- DeviceCurrentSpeed
 - quickSpinTLDevice, [134](#)
- DeviceCurrentSpeed_FullSpeed
 - TransportLayerDefsC.h, [476](#)
- DeviceCurrentSpeed_HighSpeed
 - TransportLayerDefsC.h, [476](#)
- DeviceCurrentSpeed_LowSpeed
 - TransportLayerDefsC.h, [476](#)
- DeviceCurrentSpeed_SuperSpeed
 - TransportLayerDefsC.h, [476](#)
- DeviceCurrentSpeed_UnknownSpeed
 - TransportLayerDefsC.h, [476](#)
- DeviceDisplayName
 - quickSpinTLDevice, [134](#)
- DeviceDriverVersion
 - quickSpinTLDevice, [134](#)
- DeviceEndiannessMechanism
 - quickSpinTLDevice, [134](#)
- DeviceEndiannessMechanism_Legacy
 - TransportLayerDefsC.h, [478](#)
- DeviceEndiannessMechanism_Standard

- TransportLayerDefsC.h, [478](#)
- DeviceEventChannelCount
 - quickSpin, [70](#)
- DeviceFamilyName
 - quickSpin, [70](#)
- DeviceFeaturePersistenceEnd
 - quickSpin, [70](#)
- DeviceFeaturePersistenceStart
 - quickSpin, [71](#)
- DeviceFirmwareVersion
 - quickSpin, [71](#)
- DeviceGenCPVersionMajor
 - quickSpin, [71](#)
- DeviceGenCPVersionMinor
 - quickSpin, [71](#)
- DeviceID
 - quickSpin, [71](#)
 - quickSpinTLDevice, [134](#)
 - quickSpinTLInterface, [141](#)
- DeviceIndicatorMode
 - quickSpin, [71](#)
- DeviceIndicatorMode_Active
 - CameraDefsC.h, [234](#)
- DeviceIndicatorMode_ErrorStatus
 - CameraDefsC.h, [234](#)
- DeviceIndicatorMode_Inactive
 - CameraDefsC.h, [234](#)
- DeviceInstanceId
 - quickSpinTLDevice, [134](#)
- DeviceIsUpdater
 - quickSpinTLDevice, [134](#)
- DeviceLinkBandwidthReserve
 - quickSpin, [71](#)
- DeviceLinkCommandTimeout
 - quickSpin, [71](#)
- DeviceLinkConnectionCount
 - quickSpin, [72](#)
- DeviceLinkCurrentThroughput
 - quickSpin, [72](#)
- DeviceLinkHeartbeatMode
 - quickSpin, [72](#)
- DeviceLinkHeartbeatMode_Off
 - CameraDefsC.h, [234](#)
- DeviceLinkHeartbeatMode_On
 - CameraDefsC.h, [234](#)
- DeviceLinkHeartbeatTimeout
 - quickSpin, [72](#)
- DeviceLinkSelector
 - quickSpin, [72](#)
- DeviceLinkSpeed
 - quickSpin, [72](#)
 - quickSpinTLDevice, [135](#)
- DeviceLinkThroughputLimit
 - quickSpin, [72](#)
- DeviceLinkThroughputLimitMode
 - quickSpin, [72](#)
- DeviceLinkThroughputLimitMode_Off
 - CameraDefsC.h, [234](#)
- DeviceLinkThroughputLimitMode_On
 - CameraDefsC.h, [234](#)
- DeviceLocation
 - quickSpinTLDevice, [135](#)
- DeviceManifestEntrySelector
 - quickSpin, [73](#)
- DeviceManifestPrimaryURL
 - quickSpin, [73](#)
- DeviceManifestSchemaMajorVersion
 - quickSpin, [73](#)
- DeviceManifestSchemaMinorVersion
 - quickSpin, [73](#)
- DeviceManifestSecondaryURL
 - quickSpin, [73](#)
- DeviceManifestXMLMajorVersion
 - quickSpin, [73](#)
- DeviceManifestXMLMinorVersion
 - quickSpin, [73](#)
- DeviceManifestXMLSubMinorVersion
 - quickSpin, [73](#)
- DeviceManufacturerInfo
 - quickSpin, [74](#)
- DeviceMaxThroughput
 - quickSpin, [74](#)
- DeviceModelName
 - quickSpin, [74](#)
 - quickSpinTLDevice, [135](#)
 - quickSpinTLInterface, [141](#)
- DeviceMulticastMonitorMode
 - quickSpinTLDevice, [135](#)
- DevicePortId
 - quickSpinTLDevice, [135](#)
- DevicePowerSupplySelector
 - quickSpin, [74](#)
- DevicePowerSupplySelector_External
 - CameraDefsC.h, [235](#)
- DeviceRegistersCheck
 - quickSpin, [74](#)
- DeviceRegistersEndianness
 - quickSpin, [74](#)
- DeviceRegistersEndianness_Big
 - CameraDefsC.h, [235](#)
- DeviceRegistersEndianness_Little
 - CameraDefsC.h, [235](#)
- DeviceRegistersStreamingEnd
 - quickSpin, [74](#)
- DeviceRegistersStreamingStart
 - quickSpin, [74](#)
- DeviceRegistersValid
 - quickSpin, [75](#)
- DeviceReset
 - quickSpin, [75](#)
 - quickSpinTLDevice, [135](#)
- DeviceScanType
 - quickSpin, [75](#)
- DeviceScanType_Areascan
 - CameraDefsC.h, [235](#)
- DeviceSelector

- quickSpinTLInterface, [141](#)
- DeviceSerialNumber
 - quickSpin, [75](#)
 - quickSpinTLDevice, [135](#)
 - quickSpinTLInterface, [141](#)
- DeviceSerialPortBaudRate
 - quickSpin, [75](#)
- DeviceSerialPortBaudRate_Baud115200
 - CameraDefsC.h, [236](#)
- DeviceSerialPortBaudRate_Baud19200
 - CameraDefsC.h, [236](#)
- DeviceSerialPortBaudRate_Baud230400
 - CameraDefsC.h, [236](#)
- DeviceSerialPortBaudRate_Baud38400
 - CameraDefsC.h, [236](#)
- DeviceSerialPortBaudRate_Baud460800
 - CameraDefsC.h, [236](#)
- DeviceSerialPortBaudRate_Baud57600
 - CameraDefsC.h, [236](#)
- DeviceSerialPortBaudRate_Baud921600
 - CameraDefsC.h, [236](#)
- DeviceSerialPortBaudRate_Baud9600
 - CameraDefsC.h, [236](#)
- DeviceSerialPortSelector
 - quickSpin, [75](#)
- DeviceSerialPortSelector_CameraLink
 - CameraDefsC.h, [236](#)
- DeviceSFNCVersionMajor
 - quickSpin, [75](#)
- DeviceSFNCVersionMinor
 - quickSpin, [75](#)
- DeviceSFNCVersionSubMinor
 - quickSpin, [76](#)
- DeviceStreamChannelCount
 - quickSpin, [76](#)
- DeviceStreamChannelEndianness
 - quickSpin, [76](#)
- DeviceStreamChannelEndianness_Big
 - CameraDefsC.h, [236](#)
- DeviceStreamChannelEndianness_Little
 - CameraDefsC.h, [236](#)
- DeviceStreamChannelLink
 - quickSpin, [76](#)
- DeviceStreamChannelPacketSize
 - quickSpin, [76](#)
- DeviceStreamChannelSelector
 - quickSpin, [76](#)
- DeviceStreamChannelType
 - quickSpin, [76](#)
- DeviceStreamChannelType_Receiver
 - CameraDefsC.h, [237](#)
- DeviceStreamChannelType_Transmitter
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry
 - quickSpin, [76](#)
- DeviceTapGeometry_Geometry_10X
 - CameraDefsC.h, [238](#)
- DeviceTapGeometry_Geometry_10X_1Y
 - CameraDefsC.h, [238](#)
- DeviceTapGeometry_Geometry_1X
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_1X10
 - CameraDefsC.h, [238](#)
- DeviceTapGeometry_Geometry_1X10_1Y
 - CameraDefsC.h, [238](#)
- DeviceTapGeometry_Geometry_1X2
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_1X2_1Y
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_1X2_1Y2
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_1X2_2YE
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_1X3
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_1X3_1Y
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_1X4
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_1X4_1Y
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_1X8
 - CameraDefsC.h, [238](#)
- DeviceTapGeometry_Geometry_1X8_1Y
 - CameraDefsC.h, [238](#)
- DeviceTapGeometry_Geometry_1X_1Y
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_1X_1Y2
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_1X_2YE
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2X
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2X2
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2X2_1Y
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2X2E
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2X2E_1YGeometry_2X2M_1Y
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2X2E_2YE
 - CameraDefsC.h, [238](#)
- DeviceTapGeometry_Geometry_2X2M
 - CameraDefsC.h, [238](#)
- DeviceTapGeometry_Geometry_2X_1Y
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2X_1Y2Geometry_2XE_1Y
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2X_2YE
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2XE
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2XE_1Y2
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2XE_2YE
 - CameraDefsC.h, [237](#)

- CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2XM
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2XM_1Y
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2XM_1Y2
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_2XM_2YE
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_3X
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_3X_1Y
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_4X
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_4X2
 - CameraDefsC.h, [238](#)
- DeviceTapGeometry_Geometry_4X2_1Y
 - CameraDefsC.h, [238](#)
- DeviceTapGeometry_Geometry_4X2E
 - CameraDefsC.h, [238](#)
- DeviceTapGeometry_Geometry_4X2E_1Y
 - CameraDefsC.h, [238](#)
- DeviceTapGeometry_Geometry_4X_1Y
 - CameraDefsC.h, [237](#)
- DeviceTapGeometry_Geometry_8X
 - CameraDefsC.h, [238](#)
- DeviceTapGeometry_Geometry_8X_1Y
 - CameraDefsC.h, [238](#)
- DeviceTemperature
 - quickSpin, [77](#)
- DeviceTemperatureSelector
 - quickSpin, [77](#)
- DeviceTemperatureSelector_Sensor
 - CameraDefsC.h, [238](#)
- DeviceTLType
 - quickSpin, [77](#)
- DeviceTLType_CameraLink
 - CameraDefsC.h, [238](#)
- DeviceTLType_CameraLinkHS
 - CameraDefsC.h, [238](#)
- DeviceTLType_CoaXPress
 - CameraDefsC.h, [238](#)
- DeviceTLType_Custom
 - CameraDefsC.h, [238](#)
- DeviceTLType_GigEVision
 - CameraDefsC.h, [238](#)
- DeviceTLType_USB3Vision
 - CameraDefsC.h, [238](#)
- DeviceTLVersionMajor
 - quickSpin, [77](#)
- DeviceTLVersionMinor
 - quickSpin, [77](#)
- DeviceTLVersionSubMinor
 - quickSpin, [77](#)
- DeviceType
 - quickSpin, [77](#)
 - quickSpinTLDevice, [135](#)
- DeviceType_CameraLink
 - TransportLayerDefsC.h, [478](#)
- DeviceType_CameraLinkHS
 - TransportLayerDefsC.h, [478](#)
- DeviceType_CoaXPress
 - TransportLayerDefsC.h, [478](#)
- DeviceType_Custom
 - TransportLayerDefsC.h, [478](#)
- DeviceType_GigEVision
 - TransportLayerDefsC.h, [478](#)
- DeviceType_Peripheral
 - CameraDefsC.h, [239](#)
- DeviceType_Receiver
 - CameraDefsC.h, [239](#)
- DeviceType_Transceiver
 - CameraDefsC.h, [239](#)
- DeviceType_Transmitter
 - CameraDefsC.h, [239](#)
- DeviceType_USB3Vision
 - TransportLayerDefsC.h, [478](#)
- DeviceU3VProtocol
 - quickSpinTLDevice, [136](#)
- DeviceUnlock
 - quickSpinTLInterface, [141](#)
- DeviceUpdateList
 - quickSpinTLInterface, [141](#)
- DeviceUptime
 - quickSpin, [77](#)
- DeviceUserID
 - quickSpin, [78](#)
 - quickSpinTLDevice, [136](#)
- DeviceVendorName
 - quickSpin, [78](#)
 - quickSpinTLDevice, [136](#)
 - quickSpinTLInterface, [141](#)
- DeviceVersion
 - quickSpin, [78](#)
 - quickSpinTLDevice, [136](#)
- doc/spindocs/C/GettingStarted.dox, [175](#)
- doc/spindocs/C/ProgrammerGuide.dox, [175](#)
- doc/spindocs/shared/Benefits.dox, [175](#)
- doc/spindocs/shared/FlyCapture2Comparison.dox, [175](#)
- doc/spindocs/shared/GenICamGenTL.dox, [175](#)
- doc/spindocs/shared/Licensing.dox, [175](#)
- doc/spindocs/shared/Maintenance.dox, [175](#)
- EncoderDivider
 - quickSpin, [78](#)
- EncoderMode
 - quickSpin, [78](#)
- EncoderMode_FourPhase
 - CameraDefsC.h, [239](#)
- EncoderMode_HighResolution
 - CameraDefsC.h, [239](#)
- EncoderOutputMode
 - quickSpin, [78](#)
- EncoderOutputMode_DirectionDown
 - CameraDefsC.h, [240](#)
- EncoderOutputMode_DirectionUp

- CameraDefsC.h, [240](#)
- EncoderOutputMode_Motion
 - CameraDefsC.h, [240](#)
- EncoderOutputMode_Off
 - CameraDefsC.h, [239](#)
- EncoderOutputMode_PositionDown
 - CameraDefsC.h, [240](#)
- EncoderOutputMode_PositionUp
 - CameraDefsC.h, [239](#)
- EncoderReset
 - quickSpin, [78](#)
- EncoderResetActivation
 - quickSpin, [78](#)
- EncoderResetActivation_AnyEdge
 - CameraDefsC.h, [240](#)
- EncoderResetActivation_FallingEdge
 - CameraDefsC.h, [240](#)
- EncoderResetActivation_LevelHigh
 - CameraDefsC.h, [240](#)
- EncoderResetActivation_LevelLow
 - CameraDefsC.h, [240](#)
- EncoderResetActivation_RisingEdge
 - CameraDefsC.h, [240](#)
- EncoderResetSource
 - quickSpin, [79](#)
- EncoderResetSource_AcquisitionEnd
 - CameraDefsC.h, [240](#)
- EncoderResetSource_AcquisitionStart
 - CameraDefsC.h, [240](#)
- EncoderResetSource_AcquisitionTrigger
 - CameraDefsC.h, [240](#)
- EncoderResetSource_Action0
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Action1
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Action2
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Counter0End
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Counter0Start
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Counter1End
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Counter1Start
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Counter2End
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Counter2Start
 - CameraDefsC.h, [241](#)
- EncoderResetSource_ExposureEnd
 - CameraDefsC.h, [241](#)
- EncoderResetSource_ExposureStart
 - CameraDefsC.h, [240](#)
- EncoderResetSource_FrameEnd
 - CameraDefsC.h, [240](#)
- EncoderResetSource_FrameStart
 - CameraDefsC.h, [240](#)
- EncoderResetSource_FrameTrigger
 - CameraDefsC.h, [240](#)
- EncoderResetSource_Line0
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Line1
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Line2
 - CameraDefsC.h, [241](#)
- EncoderResetSource_LinkTrigger0
 - CameraDefsC.h, [241](#)
- EncoderResetSource_LinkTrigger1
 - CameraDefsC.h, [241](#)
- EncoderResetSource_LinkTrigger2
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Off
 - CameraDefsC.h, [240](#)
- EncoderResetSource_SoftwareSignal0
 - CameraDefsC.h, [241](#)
- EncoderResetSource_SoftwareSignal1
 - CameraDefsC.h, [241](#)
- EncoderResetSource_SoftwareSignal2
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Timer0End
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Timer0Start
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Timer1End
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Timer1Start
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Timer2End
 - CameraDefsC.h, [241](#)
- EncoderResetSource_Timer2Start
 - CameraDefsC.h, [241](#)
- EncoderResetSource_UserOutput0
 - CameraDefsC.h, [241](#)
- EncoderResetSource_UserOutput1
 - CameraDefsC.h, [241](#)
- EncoderResetSource_UserOutput2
 - CameraDefsC.h, [241](#)
- EncoderSelector
 - quickSpin, [79](#)
- EncoderSelector_Encoder0
 - CameraDefsC.h, [241](#)
- EncoderSelector_Encoder1
 - CameraDefsC.h, [241](#)
- EncoderSelector_Encoder2
 - CameraDefsC.h, [241](#)
- EncoderSourceA
 - quickSpin, [79](#)
- EncoderSourceA_Line0
 - CameraDefsC.h, [242](#)
- EncoderSourceA_Line1
 - CameraDefsC.h, [242](#)
- EncoderSourceA_Line2
 - CameraDefsC.h, [242](#)
- EncoderSourceA_Off
 - CameraDefsC.h, [242](#)
- EncoderSourceB

- quickSpin, [79](#)
- EncoderSourceB_Line0
 - CameraDefsC.h, [242](#)
- EncoderSourceB_Line1
 - CameraDefsC.h, [242](#)
- EncoderSourceB_Line2
 - CameraDefsC.h, [242](#)
- EncoderSourceB_Off
 - CameraDefsC.h, [242](#)
- EncoderStatus
 - quickSpin, [79](#)
- EncoderStatus_EncoderDown
 - CameraDefsC.h, [242](#)
- EncoderStatus_EncoderIdle
 - CameraDefsC.h, [242](#)
- EncoderStatus_EncoderStatic
 - CameraDefsC.h, [242](#)
- EncoderStatus_EncoderUp
 - CameraDefsC.h, [242](#)
- EncoderTimeout
 - quickSpin, [79](#)
- EncoderValue
 - quickSpin, [79](#)
- EncoderValueAtReset
 - quickSpin, [79](#)
- EnumEntryNode
 - SpinnakerGenApiDefsC.h, [468](#)
- EnumerateGen2Cameras
 - quickSpinTLSystem, [152](#)
- EnumerateGEVInterfaces
 - quickSpinTLSystem, [152](#)
- EnumerateUSBInterfaces
 - quickSpinTLSystem, [152](#)
- EnumerationCount
 - quickSpin, [80](#)
- EnumerationNode
 - SpinnakerGenApiDefsC.h, [468](#)
- Error Handling, [23](#)
- Event Access, [28](#)
- EventAcquisitionEnd
 - quickSpin, [80](#)
- EventAcquisitionEndFrameID
 - quickSpin, [80](#)
- EventAcquisitionEndTimestamp
 - quickSpin, [80](#)
- EventAcquisitionError
 - quickSpin, [80](#)
- EventAcquisitionErrorFrameID
 - quickSpin, [80](#)
- EventAcquisitionErrorTimestamp
 - quickSpin, [80](#)
- EventAcquisitionStart
 - quickSpin, [80](#)
- EventAcquisitionStartFrameID
 - quickSpin, [81](#)
- EventAcquisitionStartTimestamp
 - quickSpin, [81](#)
- EventAcquisitionTransferEnd
 - quickSpin, [81](#)
- EventAcquisitionTransferEndFrameID
 - quickSpin, [81](#)
- EventAcquisitionTransferEndTimestamp
 - quickSpin, [81](#)
- EventAcquisitionTransferStart
 - quickSpin, [81](#)
- EventAcquisitionTransferStartFrameID
 - quickSpin, [81](#)
- EventAcquisitionTransferStartTimestamp
 - quickSpin, [81](#)
- EventAcquisitionTrigger
 - quickSpin, [82](#)
- EventAcquisitionTriggerFrameID
 - quickSpin, [82](#)
- EventAcquisitionTriggerTimestamp
 - quickSpin, [82](#)
- EventActionLate
 - quickSpin, [82](#)
- EventActionLateFrameID
 - quickSpin, [82](#)
- EventActionLateTimestamp
 - quickSpin, [82](#)
- EventCounter0End
 - quickSpin, [82](#)
- EventCounter0EndFrameID
 - quickSpin, [82](#)
- EventCounter0EndTimestamp
 - quickSpin, [83](#)
- EventCounter0Start
 - quickSpin, [83](#)
- EventCounter0StartFrameID
 - quickSpin, [83](#)
- EventCounter0StartTimestamp
 - quickSpin, [83](#)
- EventCounter1End
 - quickSpin, [83](#)
- EventCounter1EndFrameID
 - quickSpin, [83](#)
- EventCounter1EndTimestamp
 - quickSpin, [83](#)
- EventCounter1Start
 - quickSpin, [83](#)
- EventCounter1StartFrameID
 - quickSpin, [84](#)
- EventCounter1StartTimestamp
 - quickSpin, [84](#)
- EventEncoder0Restarted
 - quickSpin, [84](#)
- EventEncoder0RestartedFrameID
 - quickSpin, [84](#)
- EventEncoder0RestartedTimestamp
 - quickSpin, [84](#)
- EventEncoder0Stopped
 - quickSpin, [84](#)
- EventEncoder0StoppedFrameID
 - quickSpin, [84](#)
- EventEncoder0StoppedTimestamp

- quickSpin, [84](#)
- EventEncoder1 Restarted
 - quickSpin, [85](#)
- EventEncoder1 RestartedFrameID
 - quickSpin, [85](#)
- EventEncoder1 RestartedTimestamp
 - quickSpin, [85](#)
- EventEncoder1 Stopped
 - quickSpin, [85](#)
- EventEncoder1 StoppedFrameID
 - quickSpin, [85](#)
- EventEncoder1 StoppedTimestamp
 - quickSpin, [85](#)
- EventError
 - quickSpin, [85](#)
- EventErrorCode
 - quickSpin, [85](#)
- EventErrorFrameID
 - quickSpin, [86](#)
- EventErrorTimestamp
 - quickSpin, [86](#)
- EventExposureEnd
 - quickSpin, [86](#)
- EventExposureEndFrameID
 - quickSpin, [86](#)
- EventExposureEndTimestamp
 - quickSpin, [86](#)
- EventExposureStart
 - quickSpin, [86](#)
- EventExposureStartFrameID
 - quickSpin, [86](#)
- EventExposureStartTimestamp
 - quickSpin, [86](#)
- EventFrameBurstEnd
 - quickSpin, [87](#)
- EventFrameBurstEndFrameID
 - quickSpin, [87](#)
- EventFrameBurstEndTimestamp
 - quickSpin, [87](#)
- EventFrameBurstStart
 - quickSpin, [87](#)
- EventFrameBurstStartFrameID
 - quickSpin, [87](#)
- EventFrameBurstStartTimestamp
 - quickSpin, [87](#)
- EventFrameEnd
 - quickSpin, [87](#)
- EventFrameEndFrameID
 - quickSpin, [87](#)
- EventFrameEndTimestamp
 - quickSpin, [88](#)
- EventFrameStart
 - quickSpin, [88](#)
- EventFrameStartFrameID
 - quickSpin, [88](#)
- EventFrameStartTimestamp
 - quickSpin, [88](#)
- EventFrameTransferEnd
 - quickSpin, [88](#)
- EventFrameTransferEndFrameID
 - quickSpin, [88](#)
- EventFrameTransferEndTimestamp
 - quickSpin, [88](#)
- EventFrameTransferStart
 - quickSpin, [88](#)
- EventFrameTransferStartFrameID
 - quickSpin, [89](#)
- EventFrameTransferStartTimestamp
 - quickSpin, [89](#)
- EventFrameTrigger
 - quickSpin, [89](#)
- EventFrameTriggerFrameID
 - quickSpin, [89](#)
- EventFrameTriggerTimestamp
 - quickSpin, [89](#)
- EventLine0AnyEdge
 - quickSpin, [89](#)
- EventLine0AnyEdgeFrameID
 - quickSpin, [89](#)
- EventLine0AnyEdgeTimestamp
 - quickSpin, [89](#)
- EventLine0FallingEdge
 - quickSpin, [90](#)
- EventLine0FallingEdgeFrameID
 - quickSpin, [90](#)
- EventLine0FallingEdgeTimestamp
 - quickSpin, [90](#)
- EventLine0RisingEdge
 - quickSpin, [90](#)
- EventLine0RisingEdgeFrameID
 - quickSpin, [90](#)
- EventLine0RisingEdgeTimestamp
 - quickSpin, [90](#)
- EventLine1AnyEdge
 - quickSpin, [90](#)
- EventLine1AnyEdgeFrameID
 - quickSpin, [90](#)
- EventLine1AnyEdgeTimestamp
 - quickSpin, [91](#)
- EventLine1FallingEdge
 - quickSpin, [91](#)
- EventLine1FallingEdgeFrameID
 - quickSpin, [91](#)
- EventLine1FallingEdgeTimestamp
 - quickSpin, [91](#)
- EventLine1RisingEdge
 - quickSpin, [91](#)
- EventLine1RisingEdgeFrameID
 - quickSpin, [91](#)
- EventLine1RisingEdgeTimestamp
 - quickSpin, [91](#)
- EventLinkSpeedChange
 - quickSpin, [91](#)
- EventLinkSpeedChangeFrameID
 - quickSpin, [92](#)
- EventLinkSpeedChangeTimestamp

- quickSpin, [92](#)
- EventLinkTrigger0
 - quickSpin, [92](#)
- EventLinkTrigger0FrameID
 - quickSpin, [92](#)
- EventLinkTrigger0Timestamp
 - quickSpin, [92](#)
- EventLinkTrigger1
 - quickSpin, [92](#)
- EventLinkTrigger1FrameID
 - quickSpin, [92](#)
- EventLinkTrigger1Timestamp
 - quickSpin, [92](#)
- EventNotification
 - quickSpin, [93](#)
- EventNotification_Off
 - CameraDefsC.h, [243](#)
- EventNotification_On
 - CameraDefsC.h, [243](#)
- EventSelector
 - quickSpin, [93](#)
- EventSelector_Error
 - CameraDefsC.h, [243](#)
- EventSelector_ExposureEnd
 - CameraDefsC.h, [243](#)
- EventSelector_SerialPortReceive
 - CameraDefsC.h, [243](#)
- EventSequencerSetChange
 - quickSpin, [93](#)
- EventSequencerSetChangeFrameID
 - quickSpin, [93](#)
- EventSequencerSetChangeTimestamp
 - quickSpin, [93](#)
- EventSerialData
 - quickSpin, [93](#)
- EventSerialDataLength
 - quickSpin, [93](#)
- EventSerialPortReceive
 - quickSpin, [93](#)
- EventSerialPortReceiveTimestamp
 - quickSpin, [94](#)
- EventSerialReceiveOverflow
 - quickSpin, [94](#)
- EventStream0TransferBlockEnd
 - quickSpin, [94](#)
- EventStream0TransferBlockEndFrameID
 - quickSpin, [94](#)
- EventStream0TransferBlockEndTimestamp
 - quickSpin, [94](#)
- EventStream0TransferBlockStart
 - quickSpin, [94](#)
- EventStream0TransferBlockStartFrameID
 - quickSpin, [94](#)
- EventStream0TransferBlockStartTimestamp
 - quickSpin, [94](#)
- EventStream0TransferBlockTrigger
 - quickSpin, [95](#)
- EventStream0TransferBlockTriggerFrameID
 - quickSpin, [95](#)
- EventStream0TransferBlockTriggerTimestamp
 - quickSpin, [95](#)
- EventStream0TransferBurstEnd
 - quickSpin, [95](#)
- EventStream0TransferBurstEndFrameID
 - quickSpin, [95](#)
- EventStream0TransferBurstEndTimestamp
 - quickSpin, [95](#)
- EventStream0TransferBurstStart
 - quickSpin, [95](#)
- EventStream0TransferBurstStartFrameID
 - quickSpin, [95](#)
- EventStream0TransferBurstStartTimestamp
 - quickSpin, [96](#)
- EventStream0TransferEnd
 - quickSpin, [96](#)
- EventStream0TransferEndFrameID
 - quickSpin, [96](#)
- EventStream0TransferEndTimestamp
 - quickSpin, [96](#)
- EventStream0TransferOverflow
 - quickSpin, [96](#)
- EventStream0TransferOverflowFrameID
 - quickSpin, [96](#)
- EventStream0TransferOverflowTimestamp
 - quickSpin, [96](#)
- EventStream0TransferPause
 - quickSpin, [96](#)
- EventStream0TransferPauseFrameID
 - quickSpin, [97](#)
- EventStream0TransferPauseTimestamp
 - quickSpin, [97](#)
- EventStream0TransferResume
 - quickSpin, [97](#)
- EventStream0TransferResumeFrameID
 - quickSpin, [97](#)
- EventStream0TransferResumeTimestamp
 - quickSpin, [97](#)
- EventStream0TransferStart
 - quickSpin, [97](#)
- EventStream0TransferStartFrameID
 - quickSpin, [97](#)
- EventStream0TransferStartTimestamp
 - quickSpin, [97](#)
- EventTest
 - quickSpin, [98](#)
- EventTestTimestamp
 - quickSpin, [98](#)
- EventTimer0End
 - quickSpin, [98](#)
- EventTimer0EndFrameID
 - quickSpin, [98](#)
- EventTimer0EndTimestamp
 - quickSpin, [98](#)
- EventTimer0Start
 - quickSpin, [98](#)
- EventTimer0StartFrameID
 - quickSpin, [98](#)

- quickSpin, [98](#)
- EventTimer0StartTimestamp
 - quickSpin, [98](#)
- EventTimer1End
 - quickSpin, [99](#)
- EventTimer1EndFrameID
 - quickSpin, [99](#)
- EventTimer1EndTimestamp
 - quickSpin, [99](#)
- EventTimer1Start
 - quickSpin, [99](#)
- EventTimer1StartFrameID
 - quickSpin, [99](#)
- EventTimer1StartTimestamp
 - quickSpin, [99](#)
- Expert
 - SpinnakerGenApiDefsC.h, [469](#)
- ExposureActiveMode
 - quickSpin, [99](#)
- ExposureActiveMode_AllPixels
 - CameraDefsC.h, [243](#)
- ExposureActiveMode_AnyPixels
 - CameraDefsC.h, [243](#)
- ExposureActiveMode_Line1
 - CameraDefsC.h, [243](#)
- ExposureAuto
 - quickSpin, [99](#)
- ExposureAuto_Continuous
 - CameraDefsC.h, [244](#)
- ExposureAuto_Off
 - CameraDefsC.h, [244](#)
- ExposureAuto_Once
 - CameraDefsC.h, [244](#)
- ExposureMode
 - quickSpin, [100](#)
- ExposureMode_Timed
 - CameraDefsC.h, [244](#)
- ExposureMode_TriggerWidth
 - CameraDefsC.h, [244](#)
- ExposureTime
 - quickSpin, [100](#)
- ExposureTimeMode
 - quickSpin, [100](#)
- ExposureTimeMode_Common
 - CameraDefsC.h, [244](#)
- ExposureTimeMode_Individual
 - CameraDefsC.h, [244](#)
- ExposureTimeSelector
 - quickSpin, [100](#)
- ExposureTimeSelector_Blue
 - CameraDefsC.h, [245](#)
- ExposureTimeSelector_Common
 - CameraDefsC.h, [245](#)
- ExposureTimeSelector_Cyan
 - CameraDefsC.h, [245](#)
- ExposureTimeSelector_Green
 - CameraDefsC.h, [245](#)
- ExposureTimeSelector_Infrared
 - CameraDefsC.h, [245](#)
- ExposureTimeSelector_Magenta
 - CameraDefsC.h, [245](#)
- ExposureTimeSelector_Red
 - CameraDefsC.h, [245](#)
- ExposureTimeSelector_Stage1
 - CameraDefsC.h, [245](#)
- ExposureTimeSelector_Stage2
 - CameraDefsC.h, [245](#)
- ExposureTimeSelector_Ultraviolet
 - CameraDefsC.h, [245](#)
- ExposureTimeSelector_Yellow
 - CameraDefsC.h, [245](#)
- FactoryReset
 - quickSpin, [100](#)
- False
 - SpinnakerDefsC.h, [418](#)
- FileAccessBuffer
 - quickSpin, [100](#)
- FileAccessLength
 - quickSpin, [100](#)
- FileAccessOffset
 - quickSpin, [100](#)
- FileOpenMode
 - quickSpin, [101](#)
- FileOpenMode_Read
 - CameraDefsC.h, [245](#)
- FileOpenMode_ReadWrite
 - CameraDefsC.h, [245](#)
- FileOpenMode_Write
 - CameraDefsC.h, [245](#)
- FileOperationExecute
 - quickSpin, [101](#)
- FileOperationResult
 - quickSpin, [101](#)
- FileOperationSelector
 - quickSpin, [101](#)
- FileOperationSelector_Close
 - CameraDefsC.h, [246](#)
- FileOperationSelector_Delete
 - CameraDefsC.h, [246](#)
- FileOperationSelector_Open
 - CameraDefsC.h, [246](#)
- FileOperationSelector_Read
 - CameraDefsC.h, [246](#)
- FileOperationSelector_Write
 - CameraDefsC.h, [246](#)
- FileOperationStatus
 - quickSpin, [101](#)
- FileOperationStatus_Failure
 - CameraDefsC.h, [246](#)
- FileOperationStatus_Overflow
 - CameraDefsC.h, [246](#)
- FileOperationStatus_Success
 - CameraDefsC.h, [246](#)
- FileSelector
 - quickSpin, [101](#)
- FileSelector_SerialPort0

- CameraDefsC.h, [246](#)
- FileSelector_UserFile1
 - CameraDefsC.h, [246](#)
- FileSelector_UserSet0
 - CameraDefsC.h, [246](#)
- FileSelector_UserSet1
 - CameraDefsC.h, [246](#)
- FileSelector_UserSetDefault
 - CameraDefsC.h, [246](#)
- FileSize
 - quickSpin, [101](#)
- FilterDriverStatus
 - quickSpinTLInterface, [141](#)
- FilterDriverStatus_Disabled
 - TransportLayerDefsC.h, [478](#)
- FilterDriverStatus_Enabled
 - TransportLayerDefsC.h, [478](#)
- FilterDriverStatus_NotSupported
 - TransportLayerDefsC.h, [478](#)
- fixedIncrement
 - SpinnakerGenApiDefsC.h, [465](#)
- FloatNode
 - SpinnakerGenApiDefsC.h, [467](#)
- fnAutomatic
 - SpinnakerGenApiDefsC.h, [464](#)
- fnFixed
 - SpinnakerGenApiDefsC.h, [464](#)
- fnScientific
 - SpinnakerGenApiDefsC.h, [464](#)
- frameRate
 - spinAVIOption, [156](#)
 - spinH264Option, [164](#)
 - spinMJPGOption, [169](#)
- Gain
 - quickSpin, [101](#)
- GainAuto
 - quickSpin, [102](#)
- GainAuto_Continuous
 - CameraDefsC.h, [248](#)
- GainAuto_Off
 - CameraDefsC.h, [248](#)
- GainAuto_Once
 - CameraDefsC.h, [248](#)
- GainAutoBalance
 - quickSpin, [102](#)
- GainAutoBalance_Continuous
 - CameraDefsC.h, [248](#)
- GainAutoBalance_Off
 - CameraDefsC.h, [248](#)
- GainAutoBalance_Once
 - CameraDefsC.h, [248](#)
- GainSelector
 - quickSpin, [102](#)
- GainSelector_All
 - CameraDefsC.h, [248](#)
- Gamma
 - quickSpin, [102](#)
- GammaEnable
 - quickSpin, [102](#)
- GenICamXMLLocation
 - quickSpinTLDevice, [136](#)
- GenICamXMLLocation_Device
 - TransportLayerDefsC.h, [479](#)
- GenICamXMLLocation_Host
 - TransportLayerDefsC.h, [479](#)
- GenICamXMLPath
 - quickSpinTLDevice, [136](#)
- GenTLFNCVersionMajor
 - quickSpinTLSystem, [153](#)
- GenTLFNCVersionMinor
 - quickSpinTLSystem, [153](#)
- GenTLFNCVersionSubMinor
 - quickSpinTLSystem, [153](#)
- GenTLVersionMajor
 - quickSpinTLSystem, [153](#)
- GenTLVersionMinor
 - quickSpinTLSystem, [153](#)
- GEV
 - SpinnakerGenApiDefsC.h, [469](#)
- GevActionDeviceKey
 - quickSpinTLInterface, [142](#)
- GevActionGroupKey
 - quickSpinTLInterface, [142](#)
- GevActionGroupMask
 - quickSpinTLInterface, [142](#)
- GevActionTime
 - quickSpinTLInterface, [142](#)
- GevActiveLinkCount
 - quickSpin, [102](#)
- GevCCP
 - quickSpin, [102](#)
 - quickSpinTLDevice, [136](#)
- GevCCP_ControlAccess
 - CameraDefsC.h, [249](#)
- GevCCP_EnumEntry_GevCCP_ControlAccess
 - TransportLayerDefsC.h, [479](#)
- GevCCP_EnumEntry_GevCCP_ExclusiveAccess
 - TransportLayerDefsC.h, [479](#)
- GevCCP_EnumEntry_GevCCP_OpenAccess
 - TransportLayerDefsC.h, [479](#)
- GevCCP_ExclusiveAccess
 - CameraDefsC.h, [249](#)
- GevCCP_OpenAccess
 - CameraDefsC.h, [249](#)
- GevCurrentDefaultGateway
 - quickSpin, [102](#)
- GevCurrentIPAddress
 - quickSpin, [103](#)
- GevCurrentIPConfigurationDHCP
 - quickSpin, [103](#)
- GevCurrentIPConfigurationLLA
 - quickSpin, [103](#)
- GevCurrentIPConfigurationPersistentIP
 - quickSpin, [103](#)
- GevCurrentPhysicalLinkConfiguration
 - quickSpin, [103](#)

- GevCurrentPhysicalLinkConfiguration_DynamicLAG
 - CameraDefsC.h, [249](#)
- GevCurrentPhysicalLinkConfiguration_MultiLink
 - CameraDefsC.h, [249](#)
- GevCurrentPhysicalLinkConfiguration_SingleLink
 - CameraDefsC.h, [249](#)
- GevCurrentPhysicalLinkConfiguration_StaticLAG
 - CameraDefsC.h, [249](#)
- GevCurrentSubnetMask
 - quickSpin, [103](#)
- GevDeviceAutoForceIP
 - quickSpinTLDevice, [136](#)
 - quickSpinTLInterface, [142](#)
- GevDeviceDiscoverMaximumPacketSize
 - quickSpinTLDevice, [137](#)
- GevDeviceForceGateway
 - quickSpinTLDevice, [137](#)
 - quickSpinTLInterface, [142](#)
- GevDeviceForceIP
 - quickSpinTLDevice, [137](#)
 - quickSpinTLInterface, [142](#)
- GevDeviceForceIPAddress
 - quickSpinTLDevice, [137](#)
 - quickSpinTLInterface, [142](#)
- GevDeviceForceSubnetMask
 - quickSpinTLDevice, [137](#)
 - quickSpinTLInterface, [143](#)
- GevDeviceGateway
 - quickSpinTLDevice, [137](#)
 - quickSpinTLInterface, [143](#)
- GevDeviceIPAddress
 - quickSpinTLDevice, [137](#)
 - quickSpinTLInterface, [143](#)
- GevDeviceIsWrongSubnet
 - quickSpinTLDevice, [137](#)
- GevDeviceMACAddress
 - quickSpinTLDevice, [138](#)
 - quickSpinTLInterface, [143](#)
- GevDeviceMaximumPacketSize
 - quickSpinTLDevice, [138](#)
- GevDeviceMaximumRetryCount
 - quickSpinTLDevice, [138](#)
- GevDeviceModelsBigEndian
 - quickSpinTLDevice, [138](#)
- GevDevicePort
 - quickSpinTLDevice, [138](#)
- GevDeviceReadAndWriteTimeout
 - quickSpinTLDevice, [138](#)
- GevDeviceSubnetMask
 - quickSpinTLDevice, [138](#)
 - quickSpinTLInterface, [143](#)
- GevDiscoveryAckDelay
 - quickSpin, [103](#)
- GevFirstURL
 - quickSpin, [103](#)
- GevGVCPExtendedStatusCodes
 - quickSpin, [104](#)
- GevGVCPExtendedStatusCodesSelector
 - quickSpin, [104](#)
- GevGVCPExtendedStatusCodesSelector_Version1_1
 - CameraDefsC.h, [249](#)
- GevGVCPExtendedStatusCodesSelector_Version2_0
 - CameraDefsC.h, [249](#)
- GevGVCPHeartbeatDisable
 - quickSpin, [104](#)
- GevGVCPPendingAck
 - quickSpin, [104](#)
- GevGVCPPendingTimeout
 - quickSpin, [104](#)
- GevGVSPExtendedIDMode
 - quickSpin, [104](#)
- GevGVSPExtendedIDMode_Off
 - CameraDefsC.h, [250](#)
- GevGVSPExtendedIDMode_On
 - CameraDefsC.h, [250](#)
- GevHeartbeatTimeout
 - quickSpin, [104](#)
- GevIEEE1588
 - quickSpin, [104](#)
- GevIEEE1588ClockAccuracy
 - quickSpin, [105](#)
- GevIEEE1588ClockAccuracy_Unknown
 - CameraDefsC.h, [250](#)
- GevIEEE1588Mode
 - quickSpin, [105](#)
- GevIEEE1588Mode_Auto
 - CameraDefsC.h, [250](#)
- GevIEEE1588Mode_SlaveOnly
 - CameraDefsC.h, [250](#)
- GevIEEE1588Status
 - quickSpin, [105](#)
- GevIEEE1588Status_Disabled
 - CameraDefsC.h, [250](#)
- GevIEEE1588Status_Faulty
 - CameraDefsC.h, [250](#)
- GevIEEE1588Status_Initializing
 - CameraDefsC.h, [250](#)
- GevIEEE1588Status_Listening
 - CameraDefsC.h, [251](#)
- GevIEEE1588Status_Master
 - CameraDefsC.h, [251](#)
- GevIEEE1588Status_Passive
 - CameraDefsC.h, [251](#)
- GevIEEE1588Status_PreMaster
 - CameraDefsC.h, [251](#)
- GevIEEE1588Status_Slave
 - CameraDefsC.h, [251](#)
- GevIEEE1588Status_Uncalibrated
 - CameraDefsC.h, [251](#)
- GevInterfaceDefaultGateway
 - quickSpinTLSystem, [153](#)
- GevInterfaceDefaultIPAddress
 - quickSpinTLSystem, [153](#)
- GevInterfaceDefaultSubnetMask
 - quickSpinTLSystem, [153](#)
- GevInterfaceGateway

- quickSpinTLInterface, [143](#)
- GevInterfaceGatewaySelector
 - quickSpinTLInterface, [143](#)
- GevInterfaceMACAddress
 - quickSpinTLInterface, [143](#)
 - quickSpinTLSystem, [154](#)
- GevInterfaceMTU
 - quickSpinTLInterface, [144](#)
- GevInterfaceReceiveLinkSpeed
 - quickSpinTLInterface, [144](#)
- GevInterfaceSelector
 - quickSpin, [105](#)
- GevInterfaceSubnetIPAddress
 - quickSpinTLInterface, [144](#)
- GevInterfaceSubnetMask
 - quickSpinTLInterface, [144](#)
- GevInterfaceSubnetSelector
 - quickSpinTLInterface, [144](#)
- GevInterfaceTransmitLinkSpeed
 - quickSpinTLInterface, [144](#)
- GevIPConfigurationStatus
 - quickSpin, [105](#)
- GevIPConfigurationStatus_DHCP
 - CameraDefsC.h, [251](#)
- GevIPConfigurationStatus_ForceIP
 - CameraDefsC.h, [251](#)
- GevIPConfigurationStatus_LLA
 - CameraDefsC.h, [251](#)
- GevIPConfigurationStatus_None
 - CameraDefsC.h, [251](#)
- GevIPConfigurationStatus_PersistentIP
 - CameraDefsC.h, [251](#)
- GevMACAddress
 - quickSpin, [105](#)
- GevMCDA
 - quickSpin, [105](#)
- GevMCPHostPort
 - quickSpin, [105](#)
- GevMCRC
 - quickSpin, [106](#)
- GevMCSP
 - quickSpin, [106](#)
- GevMCTT
 - quickSpin, [106](#)
- GevNumberOfInterfaces
 - quickSpin, [106](#)
- GevPAUSEFrameReception
 - quickSpin, [106](#)
- GevPAUSEFrameTransmission
 - quickSpin, [106](#)
- GevPersistentDefaultGateway
 - quickSpin, [106](#)
- GevPersistentIPAddress
 - quickSpin, [106](#)
- GevPersistentSubnetMask
 - quickSpin, [107](#)
- GevPhysicalLinkConfiguration
 - quickSpin, [107](#)
- GevPhysicalLinkConfiguration_DynamicLAG
 - CameraDefsC.h, [251](#)
- GevPhysicalLinkConfiguration_MultiLink
 - CameraDefsC.h, [251](#)
- GevPhysicalLinkConfiguration_SingleLink
 - CameraDefsC.h, [251](#)
- GevPhysicalLinkConfiguration_StaticLAG
 - CameraDefsC.h, [251](#)
- GevPrimaryApplicationIPAddress
 - quickSpin, [107](#)
- GevPrimaryApplicationSocket
 - quickSpin, [107](#)
- GevPrimaryApplicationSwitchoverKey
 - quickSpin, [107](#)
- GevSCCFGAllInTransmission
 - quickSpin, [107](#)
- GevSCCFGExtendedChunkData
 - quickSpin, [107](#)
- GevSCCFGPacketResendDestination
 - quickSpin, [107](#)
- GevSCCFGUnconditionalStreaming
 - quickSpin, [108](#)
- GevSCDA
 - quickSpin, [108](#)
- GevSCPD
 - quickSpin, [108](#)
- GevSCPDDirection
 - quickSpin, [108](#)
- GevSCPHostPort
 - quickSpin, [108](#)
- GevSCPIInterfaceIndex
 - quickSpin, [108](#)
- GevSCPSBigEndian
 - quickSpin, [108](#)
- GevSCPSDoNotFragment
 - quickSpin, [108](#)
- GevSCPSFireTestPacket
 - quickSpin, [109](#)
- GevSCPSPacketSize
 - quickSpin, [109](#)
- GevSCSP
 - quickSpin, [109](#)
- GevSCZoneConfigurationLock
 - quickSpin, [109](#)
- GevSCZoneCount
 - quickSpin, [109](#)
- GevSCZoneDirectionAll
 - quickSpin, [109](#)
- GevSecondURL
 - quickSpin, [109](#)
- GevStreamChannelSelector
 - quickSpin, [109](#)
- GevSupportedOption
 - quickSpin, [110](#)
- GevSupportedOptionSelector
 - quickSpin, [110](#)
- GevSupportedOptionSelector_Action
 - CameraDefsC.h, [252](#)

- GevSupportedOptionSelector_CCPApplicationSocket
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_CommandsConcatenation
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_DiscoveryAckDelay
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_DiscoveryAckDelayWritable
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_Event
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_EventData
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_ExtendedStatusCodes
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_HeartbeatDisable
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_IPConfigurationDHCP
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_IPConfigurationLLA
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_IPConfigurationPersistentIP
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_LinkSpeed
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_ManifestTable
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_MessageChannelSourceSocket
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_PacketResend
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_PendingAck
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_SerialNumber
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_StreamChannelSourceSocket
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_TestData
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_UserDefinedName
 - CameraDefsC.h, [252](#)
- GevSupportedOptionSelector_WriteMem
 - CameraDefsC.h, [252](#)
- GevTimestampTickFrequency
 - quickSpin, [110](#)
- GevVersionMajor
 - quickSpinTLDevice, [138](#)
 - quickSpinTLSystem, [154](#)
- GevVersionMinor
 - quickSpinTLDevice, [139](#)
 - quickSpinTLSystem, [154](#)
- GUIXMLLocation
 - quickSpinTLDevice, [139](#)
- GUIXMLLocation_Device
 - TransportLayerDefsC.h, [479](#)
- GUIXMLLocation_Host
 - TransportLayerDefsC.h, [479](#)
- GuiXmlManifestAddress
 - quickSpin, [110](#)
- GUIXMLPath
 - quickSpinTLDevice, [139](#)
- Guru
 - SpinnakerGenApiDefsC.h, [469](#)
- Height
 - quickSpin, [110](#)
- height
 - spinAVIOption, [156](#)
 - spinH264Option, [165](#)
 - spinMJPEGOption, [169](#)
- HeightMax
 - quickSpin, [110](#)
- HexNumber
 - SpinnakerGenApiDefsC.h, [468](#)
- HostAdapterDriverVersion
 - quickSpinTLInterface, [144](#)
- HostAdapterName
 - quickSpinTLInterface, [144](#)
- HostAdapterVendor
 - quickSpinTLInterface, [145](#)
- IBoolean Access, [31](#)
- ICategory Access, [31](#)
- ICommand Access, [31](#)
- idFrom
 - SpinnakerGenApiDefsC.h, [465](#)
- idNone
 - SpinnakerGenApiDefsC.h, [465](#)
- idTo
 - SpinnakerGenApiDefsC.h, [465](#)
- IEnumEntry Access, [31](#)
- IEnumeration Access, [30](#)
- IFloat Access, [30](#)
- IIPC
 - SpinnakerGenApiDefsC.h, [469](#)
- IInteger Access, [30](#)
- Image Access, [25](#)
- Image Processor Access, [25](#)
- ImageComponentEnable
 - quickSpin, [110](#)
- ImageComponentSelector
 - quickSpin, [110](#)
- ImageComponentSelector_Color
 - CameraDefsC.h, [252](#)
- ImageComponentSelector_Confidence
 - CameraDefsC.h, [253](#)
- ImageComponentSelector_Disparity
 - CameraDefsC.h, [253](#)
- ImageComponentSelector_Infrared
 - CameraDefsC.h, [252](#)
- ImageComponentSelector_Intensity
 - CameraDefsC.h, [252](#)
- ImageComponentSelector_Range
 - CameraDefsC.h, [252](#)
- ImageComponentSelector_Scatter
 - CameraDefsC.h, [253](#)
- ImageComponentSelector_Ultraviolet
 - CameraDefsC.h, [252](#)

- ImageCompressionBitrate
 - quickSpin, [111](#)
- ImageCompressionJPEGFormatOption
 - quickSpin, [111](#)
- ImageCompressionJPEGFormatOption_BaselineOptimized
 - CameraDefsC.h, [253](#)
- ImageCompressionJPEGFormatOption_BaselineStandard
 - CameraDefsC.h, [253](#)
- ImageCompressionJPEGFormatOption_Lossless
 - CameraDefsC.h, [253](#)
- ImageCompressionJPEGFormatOption_Progressive
 - CameraDefsC.h, [253](#)
- ImageCompressionMode
 - quickSpin, [111](#)
- ImageCompressionMode_Lossless
 - CameraDefsC.h, [254](#)
- ImageCompressionMode_Off
 - CameraDefsC.h, [254](#)
- ImageCompressionQuality
 - quickSpin, [111](#)
- ImageCompressionRateOption
 - quickSpin, [111](#)
- ImageCompressionRateOption_FixBitrate
 - CameraDefsC.h, [254](#)
- ImageCompressionRateOption_FixQuality
 - CameraDefsC.h, [254](#)
- ImageList Access, [24](#)
- ImageStatistics Access, [28](#)
- include/spinc/CameraDefsC.h, [175](#)
- include/spinc/ChunkDataDefC.h, [293](#)
- include/spinc/QuickSpinC.h, [294](#)
- include/spinc/QuickSpinDefsC.h, [296](#)
- include/spinc/SpinnakerC.h, [298](#)
- include/spinc/SpinnakerDefsC.h, [403](#)
- include/spinc/SpinnakerGenApiC.h, [419](#)
- include/spinc/SpinnakerGenApiDefsC.h, [460](#)
- include/spinc/SpinnakerPlatformC.h, [471](#)
- include/spinc/SpinVideoC.h, [472](#)
- include/spinc/TransportLayerDefsC.h, [474](#)
- include/spinc/TransportLayerDeviceC.h, [482](#)
- include/spinc/TransportLayerInterfaceC.h, [483](#)
- include/spinc/TransportLayerStreamC.h, [484](#)
- include/spinc/TransportLayerSystemC.h, [484](#)
- IncompatibleDeviceCount
 - quickSpinTLInterface, [145](#)
- IncompatibleDeviceID
 - quickSpinTLInterface, [145](#)
- IncompatibleDeviceModelName
 - quickSpinTLInterface, [145](#)
- IncompatibleDeviceSelector
 - quickSpinTLInterface, [145](#)
- IncompatibleDeviceVendorName
 - quickSpinTLInterface, [145](#)
- IncompatibleGevDeviceIPAddress
 - quickSpinTLInterface, [145](#)
- IncompatibleGevDeviceMACAddress
 - quickSpinTLInterface, [145](#)
- IncompatibleGevDeviceSubnetMask
 - quickSpinTLInterface, [146](#)
- Increasing
 - SpinnakerGenApiDefsC.h, [469](#)
- indexedColor_8bit
 - spinBMPOption, [157](#)
- IntegerNode
 - SpinnakerGenApiDefsC.h, [467](#)
- Interface Access, [24](#)
- InterfaceDisplayName
 - quickSpinTLInterface, [146](#)
 - quickSpinTLSystem, [154](#)
- InterfaceID
 - quickSpinTLInterface, [146](#)
 - quickSpinTLSystem, [154](#)
- InterfaceList Access, [24](#)
- InterfaceSelector
 - quickSpinTLSystem, [154](#)
- InterfaceType
 - quickSpinTLInterface, [146](#)
- InterfaceType_CameraLink
 - TransportLayerDefsC.h, [479](#)
- InterfaceType_CameraLinkHS
 - TransportLayerDefsC.h, [480](#)
- InterfaceType_CoaxPress
 - TransportLayerDefsC.h, [480](#)
- InterfaceType_Custom
 - TransportLayerDefsC.h, [480](#)
- InterfaceType_GigEVision
 - TransportLayerDefsC.h, [479](#)
- InterfaceType_USB3Vision
 - TransportLayerDefsC.h, [480](#)
- InterfaceUpdateList
 - quickSpinTLSystem, [154](#)
- interlaced
 - spinPNGOption, [171](#)
- intfIBase
 - SpinnakerGenApiDefsC.h, [466](#)
- intfIBoolean
 - SpinnakerGenApiDefsC.h, [466](#)
- intfICategory
 - SpinnakerGenApiDefsC.h, [466](#)
- intfICommand
 - SpinnakerGenApiDefsC.h, [466](#)
- intfIEnumEntry
 - SpinnakerGenApiDefsC.h, [466](#)
- intfIEnumeration
 - SpinnakerGenApiDefsC.h, [466](#)
- intfIFloat
 - SpinnakerGenApiDefsC.h, [466](#)
- intfIInteger
 - SpinnakerGenApiDefsC.h, [466](#)
- intfIPort
 - SpinnakerGenApiDefsC.h, [466](#)
- intfIRegister
 - SpinnakerGenApiDefsC.h, [466](#)
- intfIString
 - SpinnakerGenApiDefsC.h, [466](#)
- intfIValue

- SpinnakerGenApiDefsC.h, [466](#)
- Invisible
 - SpinnakerGenApiDefsC.h, [469](#)
- IPV4Address
 - SpinnakerGenApiDefsC.h, [468](#)
- IRegister Access, [31](#)
- IspEnable
 - quickSpin, [111](#)
- IValue Access, [30](#)
- Linear
 - SpinnakerGenApiDefsC.h, [468](#)
- LineFilterWidth
 - quickSpin, [111](#)
- LineFormat
 - quickSpin, [111](#)
- LineFormat_LVDS
 - CameraDefsC.h, [254](#)
- LineFormat_NoConnect
 - CameraDefsC.h, [254](#)
- LineFormat_OpenDrain
 - CameraDefsC.h, [254](#)
- LineFormat_OptoCoupled
 - CameraDefsC.h, [254](#)
- LineFormat_RS422
 - CameraDefsC.h, [254](#)
- LineFormat_TriState
 - CameraDefsC.h, [254](#)
- LineFormat_TTL
 - CameraDefsC.h, [254](#)
- LineInputFilterSelector
 - quickSpin, [112](#)
- LineInputFilterSelector_Debounce
 - CameraDefsC.h, [255](#)
- LineInputFilterSelector_Deg glitch
 - CameraDefsC.h, [255](#)
- LineInverter
 - quickSpin, [112](#)
- LineMode
 - quickSpin, [112](#)
- LineMode_Input
 - CameraDefsC.h, [255](#)
- LineMode_Output
 - CameraDefsC.h, [255](#)
- LinePitch
 - quickSpin, [112](#)
- LineSelector
 - quickSpin, [112](#)
- LineSelector_Line0
 - CameraDefsC.h, [255](#)
- LineSelector_Line1
 - CameraDefsC.h, [255](#)
- LineSelector_Line2
 - CameraDefsC.h, [255](#)
- LineSelector_Line3
 - CameraDefsC.h, [255](#)
- LineSource
 - quickSpin, [112](#)
- LineSource_AllPixel
 - CameraDefsC.h, [256](#)
- LineSource_AnyPixel
 - CameraDefsC.h, [256](#)
- LineSource_Counter0Active
 - CameraDefsC.h, [256](#)
- LineSource_Counter1Active
 - CameraDefsC.h, [256](#)
- LineSource_ExposureActive
 - CameraDefsC.h, [256](#)
- LineSource_FrameTriggerWait
 - CameraDefsC.h, [256](#)
- LineSource_Line0
 - CameraDefsC.h, [256](#)
- LineSource_Line1
 - CameraDefsC.h, [256](#)
- LineSource_Line2
 - CameraDefsC.h, [256](#)
- LineSource_Line3
 - CameraDefsC.h, [256](#)
- LineSource_LogicBlock0
 - CameraDefsC.h, [256](#)
- LineSource_LogicBlock1
 - CameraDefsC.h, [256](#)
- LineSource_Off
 - CameraDefsC.h, [256](#)
- LineSource_PPSSignal
 - CameraDefsC.h, [256](#)
- LineSource_SerialPort0
 - CameraDefsC.h, [256](#)
- LineSource_UserOutput0
 - CameraDefsC.h, [256](#)
- LineSource_UserOutput1
 - CameraDefsC.h, [256](#)
- LineSource_UserOutput2
 - CameraDefsC.h, [256](#)
- LineSource_UserOutput3
 - CameraDefsC.h, [256](#)
- LineStatus
 - quickSpin, [112](#)
- LineStatusAll
 - quickSpin, [112](#)
- LinkErrorCount
 - quickSpin, [113](#)
- LinkUptime
 - quickSpin, [113](#)
- listIncrement
 - SpinnakerGenApiDefsC.h, [465](#)
- LittleEndian
 - SpinnakerGenApiDefsC.h, [465](#)
- Logarithmic
 - SpinnakerGenApiDefsC.h, [468](#)
- Logging Event Data Access, [28](#)
- LogicBlockLUTInputActivation
 - quickSpin, [113](#)
- LogicBlockLUTInputActivation_AnyEdge
 - CameraDefsC.h, [256](#)
- LogicBlockLUTInputActivation_FallingEdge
 - CameraDefsC.h, [256](#)

LogicBlockLUTInputActivation_LevelHigh
CameraDefsC.h, [256](#)

LogicBlockLUTInputActivation_LevelLow
CameraDefsC.h, [256](#)

LogicBlockLUTInputActivation_RisingEdge
CameraDefsC.h, [256](#)

LogicBlockLUTInputSelector
quickSpin, [113](#)

LogicBlockLUTInputSelector_Input0
CameraDefsC.h, [257](#)

LogicBlockLUTInputSelector_Input1
CameraDefsC.h, [257](#)

LogicBlockLUTInputSelector_Input2
CameraDefsC.h, [257](#)

LogicBlockLUTInputSelector_Input3
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource
quickSpin, [113](#)

LogicBlockLUTInputSource_AcquisitionActive
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_Counter0End
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_Counter0Start
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_Counter1End
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_Counter1Start
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_ExposureEnd
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_ExposureStart
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_FrameTriggerWait
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_Line0
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_Line1
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_Line2
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_Line3
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_LogicBlock0
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_LogicBlock1
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_UserOutput0
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_UserOutput1
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_UserOutput2
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_UserOutput3
CameraDefsC.h, [257](#)

LogicBlockLUTInputSource_Zero
CameraDefsC.h, [257](#)

LogicBlockLUTOutputValue
quickSpin, [113](#)

LogicBlockLUTOutputValueAll
quickSpin, [113](#)

LogicBlockLUTRowIndex
quickSpin, [113](#)

LogicBlockLUTSelector
quickSpin, [114](#)

LogicBlockLUTSelector_Enable
CameraDefsC.h, [258](#)

LogicBlockLUTSelector_Value
CameraDefsC.h, [258](#)

LogicBlockSelector
quickSpin, [114](#)

LogicBlockSelector_LogicBlock0
CameraDefsC.h, [258](#)

LogicBlockSelector_LogicBlock1
CameraDefsC.h, [258](#)

LUTEnable
quickSpin, [114](#)

LUTIndex
quickSpin, [114](#)

LUTSelector
quickSpin, [114](#)

LUTSelector_LUT1
CameraDefsC.h, [258](#)

LUTValue
quickSpin, [114](#)

LUTValueAll
quickSpin, [114](#)

m_blackLevel
spinChunkData, [159](#)

m_compressionMode
spinChunkData, [159](#)

m_compressionRatio
spinChunkData, [159](#)

m_counterValue
spinChunkData, [159](#)

m_cRC
spinChunkData, [159](#)

m_encoderValue
spinChunkData, [159](#)

m_exposureEndLineStatusAll
spinChunkData, [159](#)

m_exposureTime
spinChunkData, [160](#)

m_frameID
spinChunkData, [160](#)

m_gain
spinChunkData, [160](#)

m_height
spinChunkData, [160](#)

m_image
spinChunkData, [160](#)

m_inferenceConfidence
spinChunkData, [160](#)

m_inferenceFrameID
spinChunkData, [160](#)

m_inferenceResult
spinChunkData, [160](#)

- m_linePitch
 - spinChunkData, [161](#)
- m_lineStatusAll
 - spinChunkData, [161](#)
- m_offsetX
 - spinChunkData, [161](#)
- m_offsetY
 - spinChunkData, [161](#)
- m_partSelector
 - spinChunkData, [161](#)
- m_pixelDynamicRangeMax
 - spinChunkData, [161](#)
- m_pixelDynamicRangeMin
 - spinChunkData, [161](#)
- m_scan3dAxisMax
 - spinChunkData, [161](#)
- m_scan3dAxisMin
 - spinChunkData, [162](#)
- m_scan3dCoordinateOffset
 - spinChunkData, [162](#)
- m_scan3dCoordinateReferenceValue
 - spinChunkData, [162](#)
- m_scan3dCoordinateScale
 - spinChunkData, [162](#)
- m_scan3dInvalidDataValue
 - spinChunkData, [162](#)
- m_scan3dTransformValue
 - spinChunkData, [162](#)
- m_scanLineSelector
 - spinChunkData, [162](#)
- m_sequencerSetActive
 - spinChunkData, [162](#)
- m_serialDataLength
 - spinChunkData, [163](#)
- m_streamChannelID
 - spinChunkData, [163](#)
- m_timerValue
 - spinChunkData, [163](#)
- m_timestamp
 - spinChunkData, [163](#)
- m_timestampLatchValue
 - spinChunkData, [163](#)
- m_transferBlockID
 - spinChunkData, [163](#)
- m_transferQueueCurrentBlockCount
 - spinChunkData, [163](#)
- m_width
 - spinChunkData, [163](#)
- MACAddress
 - SpinnakerGenApiDefsC.h, [468](#)
- major
 - spinLibraryVersion, [168](#)
- MaxDeviceResetTime
 - quickSpin, [114](#)
- minor
 - spinLibraryVersion, [168](#)
- NA
 - SpinnakerGenApiDefsC.h, [464](#)
- NI
 - SpinnakerGenApiDefsC.h, [464](#)
- No
 - SpinnakerGenApiDefsC.h, [470](#)
- NoCache
 - SpinnakerGenApiDefsC.h, [464](#)
- Node Access, [29](#)
- Node Map Access, [29](#)
- noIncrement
 - SpinnakerGenApiDefsC.h, [465](#)
- None
 - SpinnakerGenApiDefsC.h, [469](#)
- NUM_ACQUISITIONMODE
 - CameraDefsC.h, [208](#)
- NUM_ACQUISITIONSTATUSSELECTOR
 - CameraDefsC.h, [208](#)
- NUM_ACTIONUNCONDITIONALMODE
 - CameraDefsC.h, [209](#)
- NUM_ADCBITDEPTH
 - CameraDefsC.h, [209](#)
- NUM_AUTOALGORITHMSELECTOR
 - CameraDefsC.h, [209](#)
- NUM_AUTOEXPOSURECONTROLPRIORITY
 - CameraDefsC.h, [210](#)
- NUM_AUTOEXPOSURELIGHTINGMODE
 - CameraDefsC.h, [210](#)
- NUM_AUTOEXPOSUREMETERINGMODE
 - CameraDefsC.h, [211](#)
- NUM_AUTOEXPOSURETARGETGREYVALUEAUTO
 - CameraDefsC.h, [211](#)
- NUM_BALANCERATIOSELECTOR
 - CameraDefsC.h, [211](#)
- NUM_BALANCEWHITEAUTO
 - CameraDefsC.h, [212](#)
- NUM_BALANCEWHITEAUTOPROFILE
 - CameraDefsC.h, [212](#)
- NUM_BINNINGHORIZONTALMODE
 - CameraDefsC.h, [212](#)
- NUM_BINNINGSELECTOR
 - CameraDefsC.h, [213](#)
- NUM_BINNINGVERTICALMODE
 - CameraDefsC.h, [213](#)
- NUM_BLACKLEVELAUTO
 - CameraDefsC.h, [214](#)
- NUM_BLACKLEVELAUTOBALANCE
 - CameraDefsC.h, [213](#)
- NUM_BLACKLEVELSELECTOR
 - CameraDefsC.h, [214](#)
- NUM_CHUNKBLACKLEVELSELECTOR
 - CameraDefsC.h, [214](#)
- NUM_CHUNKCOUNTERSELECTOR
 - CameraDefsC.h, [215](#)
- NUM_CHUNKENCODERSELECTOR
 - CameraDefsC.h, [215](#)
- NUM_CHUNKENCODERSTATUS
 - CameraDefsC.h, [215](#)
- NUM_CHUNKEXPOSURETIMESELECTOR
 - CameraDefsC.h, [216](#)

- NUM_CHUNKGAINSELECTOR
 - CameraDefsC.h, [216](#)
- NUM_CHUNKIMAGECOMPONENT
 - CameraDefsC.h, [217](#)
- NUM_CHUNKPIXELFORMAT
 - CameraDefsC.h, [217](#)
- NUM_CHUNKREGIONID
 - CameraDefsC.h, [217](#)
- NUM_CHUNKSCAN3DCOORDINATEREFERENCESELECTOR
 - CameraDefsC.h, [218](#)
- NUM_CHUNKSCAN3DCOORDINATESELECTOR
 - CameraDefsC.h, [218](#)
- NUM_CHUNKSCAN3DCOORDINATESYSTEM
 - CameraDefsC.h, [218](#)
- NUM_CHUNKSCAN3DCOORDINATESYSTEMREFERENCESELECTOR
 - CameraDefsC.h, [219](#)
- NUM_CHUNKSCAN3DCOORDINATETRANSFORMSELECTOR
 - CameraDefsC.h, [219](#)
- NUM_CHUNKSCAN3DDISTANCEUNIT
 - CameraDefsC.h, [219](#)
- NUM_CHUNKSCAN3DOUTPUTMODE
 - CameraDefsC.h, [220](#)
- NUM_CHUNKSELECTOR
 - CameraDefsC.h, [221](#)
- NUM_CHUNKSOURCEID
 - CameraDefsC.h, [221](#)
- NUM_CHUNKTIMERSELECTOR
 - CameraDefsC.h, [222](#)
- NUM_CHUNKTRANSFERSTREAMID
 - CameraDefsC.h, [222](#)
- NUM_CLCONFIGURATION
 - CameraDefsC.h, [222](#)
- NUM_CLTIMESLOTSCOUNT
 - CameraDefsC.h, [223](#)
- NUM_COLORTRANSFORMATIONSELECTOR
 - CameraDefsC.h, [223](#)
- NUM_COLORTRANSFORMATIONVALUESELECTOR
 - CameraDefsC.h, [223](#)
- NUM_COMPRESSIONSATURATIONPRIORITY
 - CameraDefsC.h, [224](#)
- NUM_COUNTEREVENTACTIVATION
 - CameraDefsC.h, [224](#)
- NUM_COUNTEREVENTSOURCE
 - CameraDefsC.h, [225](#)
- NUM_COUNTERRESETACTIVATION
 - CameraDefsC.h, [225](#)
- NUM_COUNTERRESETSOURCE
 - CameraDefsC.h, [226](#)
- NUM_COUNTERSELECTOR
 - CameraDefsC.h, [226](#)
- NUM_COUNTERSTATUS
 - CameraDefsC.h, [226](#)
- NUM_COUNTERTRIGGERACTIVATION
 - CameraDefsC.h, [227](#)
- NUM_COUNTERTRIGGERSOURCE
 - CameraDefsC.h, [227](#)
- NUM_CXPCONNECTIONTESTMODE
 - CameraDefsC.h, [228](#)
- NUM_CXPLINKCONFIGURATION
 - CameraDefsC.h, [229](#)
- NUM_CXPLINKCONFIGURATIONPREFERRED
 - CameraDefsC.h, [230](#)
- NUM_CXPLINKCONFIGURATIONSTATUS
 - CameraDefsC.h, [231](#)
- NUM_CXPPOCXPSTATUS
 - CameraDefsC.h, [231](#)
- NUM_DECIMATIONHORIZONTALMODE
 - CameraDefsC.h, [231](#)
- NUM_DECIMATIONSELECTOR
 - CameraDefsC.h, [232](#)
- NUM_DECIMATIONVERTICALMODE
 - CameraDefsC.h, [232](#)
- NUM_DEFECTCORRECTIONMODE
 - CameraDefsC.h, [232](#)
- NUM_DEINTERLACING
 - CameraDefsC.h, [233](#)
- NUM_DEVICECHARACTERSET
 - CameraDefsC.h, [233](#)
- NUM_DEVICECLOCKSELECTOR
 - CameraDefsC.h, [233](#)
- NUM_DEVICECONNECTIONSTATUS
 - CameraDefsC.h, [233](#)
- NUM_DEVICEINDICATORMODE
 - CameraDefsC.h, [234](#)
- NUM_DEVICELINKHEARTBEATMODE
 - CameraDefsC.h, [234](#)
- NUM_DEVICELINKTHROUGHPUTLIMITMODE
 - CameraDefsC.h, [234](#)
- NUM_DEVICEPOWERSUPPLYSELECTOR
 - CameraDefsC.h, [235](#)
- NUM_DEVICEREGISTERSENDIANNESS
 - CameraDefsC.h, [235](#)
- NUM_DEVICESCANTYPE
 - CameraDefsC.h, [235](#)
- NUM_DEVICESERIALPORTBAUDRATE
 - CameraDefsC.h, [236](#)
- NUM_DEVICESERIALPORTSELECTOR
 - CameraDefsC.h, [236](#)
- NUM_DEVICESTREAMCHANNELENDIANNESS
 - CameraDefsC.h, [236](#)
- NUM_DEVICESTREAMCHANNELTYPE
 - CameraDefsC.h, [237](#)
- NUM_DEVICETAPGEOMETRY
 - CameraDefsC.h, [238](#)
- NUM_DEVICETEMPERATURESELECTOR
 - CameraDefsC.h, [238](#)
- NUM_DEVICETLTYPE
 - CameraDefsC.h, [238](#)
- NUM_DEVICETYPE
 - CameraDefsC.h, [239](#)
- NUM_ENCODERMODE
 - CameraDefsC.h, [239](#)
- NUM_ENCODEROUTPUTMODE
 - CameraDefsC.h, [240](#)
- NUM_ENCODERRESETACTIVATION
 - CameraDefsC.h, [240](#)

- NUM_ENCODERRESETSOURCE
 - CameraDefsC.h, [241](#)
- NUM_ENCODERSELECTOR
 - CameraDefsC.h, [241](#)
- NUM_ENCODERSOURCEA
 - CameraDefsC.h, [242](#)
- NUM_ENCODERSOURCEB
 - CameraDefsC.h, [242](#)
- NUM_ENCODERSTATUS
 - CameraDefsC.h, [242](#)
- NUM_EVENTNOTIFICATION
 - CameraDefsC.h, [243](#)
- NUM_EVENTSELECTOR
 - CameraDefsC.h, [243](#)
- NUM_EXPOSUREACTIVEMODE
 - CameraDefsC.h, [243](#)
- NUM_EXPOSUREAUTO
 - CameraDefsC.h, [244](#)
- NUM_EXPOSUREMODE
 - CameraDefsC.h, [244](#)
- NUM_EXPOSURETIMEMODE
 - CameraDefsC.h, [244](#)
- NUM_EXPOSURETIMESELECTOR
 - CameraDefsC.h, [245](#)
- NUM_FILEOPENMODE
 - CameraDefsC.h, [245](#)
- NUM_FILEOPERATIONSELECTOR
 - CameraDefsC.h, [246](#)
- NUM_FILEOPERATIONSTATUS
 - CameraDefsC.h, [246](#)
- NUM_FILESELECTOR
 - CameraDefsC.h, [246](#)
- NUM_GAINAUTO
 - CameraDefsC.h, [248](#)
- NUM_GAINAUTOBALANCE
 - CameraDefsC.h, [248](#)
- NUM_GAINSELECTOR
 - CameraDefsC.h, [248](#)
- NUM_GEVCPP
 - CameraDefsC.h, [249](#)
- NUM_GEVCURRENTPHYSICALLINKCONFIGURATION
 - CameraDefsC.h, [249](#)
- NUM_GEVGVCPEXTENDEDSTATUSCODESELECTOR
 - CameraDefsC.h, [249](#)
- NUM_GEVGVSPEXTENDEDIDMODE
 - CameraDefsC.h, [250](#)
- NUM_GEVIEEE1588CLOCKACCURACY
 - CameraDefsC.h, [250](#)
- NUM_GEVIEEE1588MODE
 - CameraDefsC.h, [250](#)
- NUM_GEVIEEE1588STATUS
 - CameraDefsC.h, [251](#)
- NUM_GEVIPCONFIGURATIONSTATUS
 - CameraDefsC.h, [251](#)
- NUM_GEVPHYSCALLINKCONFIGURATION
 - CameraDefsC.h, [251](#)
- NUM_GEVSUPPORTEDOPTIONSELECTOR
 - CameraDefsC.h, [252](#)
- NUM_IMAGECOMPONENTSELECTOR
 - CameraDefsC.h, [253](#)
- NUM_IMAGECOMPRESSIONJPEGFORMATOPTION
 - CameraDefsC.h, [253](#)
- NUM_IMAGECOMPRESSIONMODE
 - CameraDefsC.h, [254](#)
- NUM_IMAGECOMPRESSIONRATEOPTION
 - CameraDefsC.h, [254](#)
- NUM_LINEFORMAT
 - CameraDefsC.h, [254](#)
- NUM_LINEINPUTFILTERSELECTOR
 - CameraDefsC.h, [255](#)
- NUM_LINEMODE
 - CameraDefsC.h, [255](#)
- NUM_LINESELECTOR
 - CameraDefsC.h, [255](#)
- NUM_LINESOURCE
 - CameraDefsC.h, [256](#)
- NUM_LOGICBLOCKLUTINPUTACTIVATION
 - CameraDefsC.h, [256](#)
- NUM_LOGICBLOCKLUTINPUTSELECTOR
 - CameraDefsC.h, [257](#)
- NUM_LOGICBLOCKLUTINPUTSOURCE
 - CameraDefsC.h, [257](#)
- NUM_LOGICBLOCKLUTSELECTOR
 - CameraDefsC.h, [258](#)
- NUM_LOGICBLOCKSELECTOR
 - CameraDefsC.h, [258](#)
- NUM_LUTSELECTOR
 - CameraDefsC.h, [258](#)
- NUM_PIXELCOLORFILTER
 - CameraDefsC.h, [259](#)
- NUM_PIXELFORMAT
 - CameraDefsC.h, [264](#)
- NUM_PIXELFORMATINFOSELECTOR
 - CameraDefsC.h, [270](#)
- NUM_PIXELSIZE
 - CameraDefsC.h, [271](#)
- NUM_REGIONDESTINATION
 - CameraDefsC.h, [271](#)
- NUM_REGIONMODE
 - CameraDefsC.h, [271](#)
- NUM_REGIONSELECTOR
 - CameraDefsC.h, [272](#)
- NUM_RGBTRANSFORMLIGHTSOURCE
 - CameraDefsC.h, [272](#)
- NUM_SCAN3DCOORDINATEREFERENCESELECTOR
 - CameraDefsC.h, [273](#)
- NUM_SCAN3DCOORDINATESELECTOR
 - CameraDefsC.h, [273](#)
- NUM_SCAN3DCOORDINATESYSTEM
 - CameraDefsC.h, [273](#)
- NUM_SCAN3DCOORDINATESYSTEMREFERENCE
 - CameraDefsC.h, [274](#)
- NUM_SCAN3DCOORDINATETTRANSFORMSELECTOR
 - CameraDefsC.h, [274](#)
- NUM_SCAN3DDISTANCEUNIT
 - CameraDefsC.h, [274](#)

- NUM_SCAN3DOUTPUTMODE
 - CameraDefsC.h, [276](#)
- NUM_SENSORDIGITIZATIONTAPS
 - CameraDefsC.h, [277](#)
- NUM_SENSORSHUTTERMODE
 - CameraDefsC.h, [277](#)
- NUM_SENSORTAPS
 - CameraDefsC.h, [277](#)
- NUM_SEQUENCERCONFIGURATIONMODE
 - CameraDefsC.h, [278](#)
- NUM_SEQUENCERCONFIGURATIONVALID
 - CameraDefsC.h, [278](#)
- NUM_SEQUENCERMODE
 - CameraDefsC.h, [278](#)
- NUM_SEQUENCERSETVALID
 - CameraDefsC.h, [279](#)
- NUM_SEQUENCERTRIGGERACTIVATION
 - CameraDefsC.h, [279](#)
- NUM_SEQUENCERTRIGGERSOURCE
 - CameraDefsC.h, [279](#)
- NUM_SERIALPORTBAUDRATE
 - CameraDefsC.h, [280](#)
- NUM_SERIALPORTPARITY
 - CameraDefsC.h, [280](#)
- NUM_SERIALPORTSELECTOR
 - CameraDefsC.h, [280](#)
- NUM_SERIALPORTSOURCE
 - CameraDefsC.h, [281](#)
- NUM_SERIALPORTSTOPBITS
 - CameraDefsC.h, [281](#)
- NUM_SOFTWARESIGNALSELECTOR
 - CameraDefsC.h, [281](#)
- NUM_SOURCESELECTOR
 - CameraDefsC.h, [282](#)
- NUM_TESTPATTERN
 - CameraDefsC.h, [282](#)
- NUM_TESTPATTERNGENERATORSELECTOR
 - CameraDefsC.h, [282](#)
- NUM_TIMERSELECTOR
 - CameraDefsC.h, [283](#)
- NUM_TIMERSTATUS
 - CameraDefsC.h, [283](#)
- NUM_TIMERTRIGGERACTIVATION
 - CameraDefsC.h, [283](#)
- NUM_TIMERTRIGGERSOURCE
 - CameraDefsC.h, [285](#)
- NUM_TRANSFERCOMPONENTSELECTOR
 - CameraDefsC.h, [285](#)
- NUM_TRANSFERCONTROLMODE
 - CameraDefsC.h, [286](#)
- NUM_TRANSFEROPERATIONMODE
 - CameraDefsC.h, [286](#)
- NUM_TRANSFERQUEUEMODE
 - CameraDefsC.h, [286](#)
- NUM_TRANSFERSELECTOR
 - CameraDefsC.h, [286](#)
- NUM_TRANSFERSTATUSSELECTOR
 - CameraDefsC.h, [287](#)
- NUM_TRANSFERTRIGGERACTIVATION
 - CameraDefsC.h, [287](#)
- NUM_TRANSFERTRIGGERMODE
 - CameraDefsC.h, [288](#)
- NUM_TRANSFERTRIGGERSELECTOR
 - CameraDefsC.h, [288](#)
- NUM_TRANSFERTRIGGERSOURCE
 - CameraDefsC.h, [289](#)
- NUM_TRIGGERACTIVATION
 - CameraDefsC.h, [289](#)
- NUM_TRIGGERMODE
 - CameraDefsC.h, [290](#)
- NUM_TRIGGEROVERLAP
 - CameraDefsC.h, [290](#)
- NUM_TRIGGERSELECTOR
 - CameraDefsC.h, [290](#)
- NUM_TRIGGERSOURCE
 - CameraDefsC.h, [291](#)
- NUM_USEROUTPUTSELECTOR
 - CameraDefsC.h, [291](#)
- NUM_USERSETDEFAULT
 - CameraDefsC.h, [292](#)
- NUM_USERSETSELECTOR
 - CameraDefsC.h, [292](#)
- NUM_WHITECLIPSELECTOR
 - CameraDefsC.h, [292](#)
- NUMDEVICEACCESSSTATUS
 - TransportLayerDefsC.h, [476](#)
- NUMDEVICECURRENTSPEED
 - TransportLayerDefsC.h, [476](#)
- NUMDEVICEENDIANESSMECHANISM
 - TransportLayerDefsC.h, [478](#)
- NUMDEVICETYPE
 - TransportLayerDefsC.h, [478](#)
- NUMFILTERDRIVERSTATUS
 - TransportLayerDefsC.h, [478](#)
- NUMGENICAMXMLLOCATION
 - TransportLayerDefsC.h, [479](#)
- NUMGEVCCP
 - TransportLayerDefsC.h, [479](#)
- NUMGUIXMLLOCATION
 - TransportLayerDefsC.h, [479](#)
- NUMINTERFACETYPE
 - TransportLayerDefsC.h, [480](#)
- NUMPOESTATUS
 - TransportLayerDefsC.h, [480](#)
- NUMSTREAMBUFFERCOUNTMODE
 - TransportLayerDefsC.h, [480](#)
- NUMSTREAMBUFFERHANDLINGMODE
 - TransportLayerDefsC.h, [481](#)
- NUMSTREAMMODE
 - TransportLayerDefsC.h, [481](#)
- NUMSTREAMTYPE
 - TransportLayerDefsC.h, [482](#)
- NUMTLTYPE
 - TransportLayerDefsC.h, [482](#)
- OffsetX
 - quickSpin, [115](#)

- OffsetY
 - quickSpin, [115](#)
- PacketResendRequestCount
 - quickSpin, [115](#)
- PayloadSize
 - quickSpin, [115](#)
- PixelColorFilter
 - quickSpin, [115](#)
- PixelColorFilter_BayerBG
 - CameraDefsC.h, [259](#)
- PixelColorFilter_BayerGB
 - CameraDefsC.h, [259](#)
- PixelColorFilter_BayerGR
 - CameraDefsC.h, [259](#)
- PixelColorFilter_BayerRG
 - CameraDefsC.h, [258](#)
- PixelColorFilter_None
 - CameraDefsC.h, [258](#)
- PixelDynamicRangeMax
 - quickSpin, [115](#)
- PixelDynamicRangeMin
 - quickSpin, [115](#)
- PixelFormat
 - quickSpin, [115](#)
- PixelFormat_B10
 - CameraDefsC.h, [261](#)
- PixelFormat_B12
 - CameraDefsC.h, [261](#)
- PixelFormat_B12_Jpeg
 - CameraDefsC.h, [264](#)
- PixelFormat_B16
 - CameraDefsC.h, [261](#)
- PixelFormat_B8
 - CameraDefsC.h, [261](#)
- PixelFormat_BayerBG10
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerBG10p
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerBG10Packed
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerBG12
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerBG12p
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerBG12Packed
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerBG16
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerBG8
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerGB10
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerGB10p
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerGB10Packed
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerGB12
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerGB12p
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerGB12Packed
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerGB16
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerGB12p
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerGB12Packed
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerGR10
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerGR10p
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerGR10Packed
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerGR12
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerGR12p
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerGR12Packed
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerGR16
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerGR8
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerRG10
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerRG10p
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerRG10Packed
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerRG12
 - CameraDefsC.h, [260](#)
- PixelFormat_BayerRG12p
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerRG12Packed
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerRG16
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerRG8
 - CameraDefsC.h, [259](#)
- PixelFormat_BayerRGPolarized10p
 - CameraDefsC.h, [264](#)
- PixelFormat_BayerRGPolarized12p
 - CameraDefsC.h, [264](#)
- PixelFormat_BayerRGPolarized16
 - CameraDefsC.h, [264](#)
- PixelFormat_BayerRGPolarized8
 - CameraDefsC.h, [264](#)
- PixelFormat_BGR10
 - CameraDefsC.h, [261](#)
- PixelFormat_BGR10p
 - CameraDefsC.h, [261](#)
- PixelFormat_BGR12
 - CameraDefsC.h, [261](#)
- PixelFormat_BGR12p
 - CameraDefsC.h, [261](#)
- PixelFormat_BGR14
 - CameraDefsC.h, [261](#)

PixelFormat_BGR16
 CameraDefsC.h, [261](#)

PixelFormat_BGR565p
 CameraDefsC.h, [261](#)

PixelFormat_BGR8
 CameraDefsC.h, [259](#)

PixelFormat_BGRa10
 CameraDefsC.h, [261](#)

PixelFormat_BGRa10p
 CameraDefsC.h, [261](#)

PixelFormat_BGRa12
 CameraDefsC.h, [261](#)

PixelFormat_BGRa12p
 CameraDefsC.h, [261](#)

PixelFormat_BGRa14
 CameraDefsC.h, [261](#)

PixelFormat_BGRa16
 CameraDefsC.h, [261](#)

PixelFormat_BGRa8
 CameraDefsC.h, [259](#)

PixelFormat_BiColorBGRG10
 CameraDefsC.h, [262](#)

PixelFormat_BiColorBGRG10p
 CameraDefsC.h, [262](#)

PixelFormat_BiColorBGRG12
 CameraDefsC.h, [262](#)

PixelFormat_BiColorBGRG12p
 CameraDefsC.h, [262](#)

PixelFormat_BiColorBGRG8
 CameraDefsC.h, [262](#)

PixelFormat_BiColorRGBG10
 CameraDefsC.h, [262](#)

PixelFormat_BiColorRGBG10p
 CameraDefsC.h, [262](#)

PixelFormat_BiColorRGBG12
 CameraDefsC.h, [262](#)

PixelFormat_BiColorRGBG12p
 CameraDefsC.h, [262](#)

PixelFormat_BiColorRGBG8
 CameraDefsC.h, [262](#)

PixelFormat_Confidence1
 CameraDefsC.h, [262](#)

PixelFormat_Confidence16
 CameraDefsC.h, [262](#)

PixelFormat_Confidence1p
 CameraDefsC.h, [262](#)

PixelFormat_Confidence32f
 CameraDefsC.h, [262](#)

PixelFormat_Confidence8
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_A10p
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_A12p
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_A16
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_A32f
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_A8
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_ABC10p
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_ABC10p_Planar
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_ABC12p
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_ABC12p_Planar
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_ABC16
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_ABC16_Planar
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_ABC32f
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_ABC32f_Planar
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_ABC8
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_ABC8_Planar
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_AC10p
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_AC10p_Planar
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_AC12p
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_AC12p_Planar
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_AC16
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_AC16_Planar
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_AC32f
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_AC32f_Planar
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_AC8
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_AC8_Planar
 CameraDefsC.h, [261](#)

PixelFormat_Coord3D_B10p
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_B12p
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_B16
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_B32f
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_B8
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_C10p
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_C12p
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_C16
 CameraDefsC.h, [262](#)

PixelFormat_Coord3D_C32f
 CameraDefsC.h, [262](#)
PixelFormat_Coord3D_C8
 CameraDefsC.h, [262](#)
PixelFormat_G10
 CameraDefsC.h, [261](#)
PixelFormat_G12
 CameraDefsC.h, [261](#)
PixelFormat_G16
 CameraDefsC.h, [261](#)
PixelFormat_G8
 CameraDefsC.h, [261](#)
PixelFormat_GB12
 CameraDefsC.h, [264](#)
PixelFormat_GB12_Jpeg
 CameraDefsC.h, [264](#)
PixelFormat_GR12
 CameraDefsC.h, [264](#)
PixelFormat_GR12_Jpeg
 CameraDefsC.h, [264](#)
PixelFormat_JPEGColor8
 CameraDefsC.h, [264](#)
PixelFormat_JPEGMono8
 CameraDefsC.h, [264](#)
PixelFormat_LLCBayerRG8
 CameraDefsC.h, [264](#)
PixelFormat_LLCMono8
 CameraDefsC.h, [264](#)
PixelFormat_Mono10
 CameraDefsC.h, [260](#)
PixelFormat_Mono10p
 CameraDefsC.h, [260](#)
PixelFormat_Mono10Packed
 CameraDefsC.h, [259](#)
PixelFormat_Mono12
 CameraDefsC.h, [260](#)
PixelFormat_Mono12p
 CameraDefsC.h, [259](#)
PixelFormat_Mono12Packed
 CameraDefsC.h, [259](#)
PixelFormat_Mono14
 CameraDefsC.h, [260](#)
PixelFormat_Mono16
 CameraDefsC.h, [259](#)
PixelFormat_Mono16s
 CameraDefsC.h, [260](#)
PixelFormat_Mono1p
 CameraDefsC.h, [260](#)
PixelFormat_Mono2p
 CameraDefsC.h, [260](#)
PixelFormat_Mono32f
 CameraDefsC.h, [260](#)
PixelFormat_Mono4p
 CameraDefsC.h, [260](#)
PixelFormat_Mono8
 CameraDefsC.h, [259](#)
PixelFormat_Mono8s
 CameraDefsC.h, [260](#)
PixelFormat_Polarized10p
 CameraDefsC.h, [264](#)
PixelFormat_Polarized12p
 CameraDefsC.h, [264](#)
PixelFormat_Polarized16
 CameraDefsC.h, [264](#)
PixelFormat_Polarized8
 CameraDefsC.h, [264](#)
PixelFormat_R10
 CameraDefsC.h, [261](#)
PixelFormat_R12
 CameraDefsC.h, [261](#)
PixelFormat_R12_Jpeg
 CameraDefsC.h, [264](#)
PixelFormat_R16
 CameraDefsC.h, [261](#)
PixelFormat_R8
 CameraDefsC.h, [261](#)
PixelFormat_Raw16
 CameraDefsC.h, [264](#)
PixelFormat_Raw8
 CameraDefsC.h, [264](#)
PixelFormat_RGB10
 CameraDefsC.h, [260](#)
PixelFormat_RGB10_Planar
 CameraDefsC.h, [260](#)
PixelFormat_RGB10p
 CameraDefsC.h, [260](#)
PixelFormat_RGB10p32
 CameraDefsC.h, [260](#)
PixelFormat_RGB12
 CameraDefsC.h, [260](#)
PixelFormat_RGB12_Planar
 CameraDefsC.h, [260](#)
PixelFormat_RGB12p
 CameraDefsC.h, [260](#)
PixelFormat_RGB14
 CameraDefsC.h, [260](#)
PixelFormat_RGB16
 CameraDefsC.h, [260](#)
PixelFormat_RGB16_Planar
 CameraDefsC.h, [260](#)
PixelFormat_RGB16s
 CameraDefsC.h, [260](#)
PixelFormat_RGB32f
 CameraDefsC.h, [260](#)
PixelFormat_RGB565p
 CameraDefsC.h, [261](#)
PixelFormat_RGB8
 CameraDefsC.h, [260](#)
PixelFormat_RGB8_Planar
 CameraDefsC.h, [260](#)
PixelFormat_RGB8Packed
 CameraDefsC.h, [259](#)
PixelFormat_RGBa10
 CameraDefsC.h, [260](#)
PixelFormat_RGBa10p
 CameraDefsC.h, [260](#)

PixelFormat_RGBA12
CameraDefsC.h, [260](#)

PixelFormat_RGBA12p
CameraDefsC.h, [260](#)

PixelFormat_RGBA14
CameraDefsC.h, [260](#)

PixelFormat_RGBA16
CameraDefsC.h, [260](#)

PixelFormat_RGBA32f
CameraDefsC.h, [261](#)

PixelFormat_RGBA8
CameraDefsC.h, [260](#)

PixelFormat_SCF1WBWG10
CameraDefsC.h, [262](#)

PixelFormat_SCF1WBWG10p
CameraDefsC.h, [262](#)

PixelFormat_SCF1WBWG12
CameraDefsC.h, [262](#)

PixelFormat_SCF1WBWG12p
CameraDefsC.h, [262](#)

PixelFormat_SCF1WBWG14
CameraDefsC.h, [262](#)

PixelFormat_SCF1WBWG16
CameraDefsC.h, [262](#)

PixelFormat_SCF1WBWG8
CameraDefsC.h, [262](#)

PixelFormat_SCF1WGWB10
CameraDefsC.h, [262](#)

PixelFormat_SCF1WGWB10p
CameraDefsC.h, [262](#)

PixelFormat_SCF1WGWB12
CameraDefsC.h, [262](#)

PixelFormat_SCF1WGWB12p
CameraDefsC.h, [262](#)

PixelFormat_SCF1WGWB14
CameraDefsC.h, [262](#)

PixelFormat_SCF1WGWB16
CameraDefsC.h, [262](#)

PixelFormat_SCF1WGWB8
CameraDefsC.h, [262](#)

PixelFormat_SCF1WGWR10
CameraDefsC.h, [262](#)

PixelFormat_SCF1WGWR10p
CameraDefsC.h, [262](#)

PixelFormat_SCF1WGWR12
CameraDefsC.h, [263](#)

PixelFormat_SCF1WGWR12p
CameraDefsC.h, [263](#)

PixelFormat_SCF1WGWR14
CameraDefsC.h, [263](#)

PixelFormat_SCF1WGWR16
CameraDefsC.h, [263](#)

PixelFormat_SCF1WGWR8
CameraDefsC.h, [262](#)

PixelFormat_SCF1WRWG10
CameraDefsC.h, [263](#)

PixelFormat_SCF1WRWG10p
CameraDefsC.h, [263](#)

PixelFormat_SCF1WRWG12
CameraDefsC.h, [263](#)

PixelFormat_SCF1WRWG12p
CameraDefsC.h, [263](#)

PixelFormat_SCF1WRWG14
CameraDefsC.h, [263](#)

PixelFormat_SCF1WRWG16
CameraDefsC.h, [263](#)

PixelFormat_SCF1WRWG8
CameraDefsC.h, [263](#)

PixelFormat_YCbCr10_CbYCr
CameraDefsC.h, [263](#)

PixelFormat_YCbCr10p_CbYCr
CameraDefsC.h, [263](#)

PixelFormat_YCbCr12_CbYCr
CameraDefsC.h, [263](#)

PixelFormat_YCbCr12p_CbYCr
CameraDefsC.h, [263](#)

PixelFormat_YCbCr411_8
CameraDefsC.h, [259](#)

PixelFormat_YCbCr411_8_CbYYCrYY
CameraDefsC.h, [263](#)

PixelFormat_YCbCr422_10
CameraDefsC.h, [263](#)

PixelFormat_YCbCr422_10_CbYCrY
CameraDefsC.h, [263](#)

PixelFormat_YCbCr422_10p
CameraDefsC.h, [263](#)

PixelFormat_YCbCr422_10p_CbYCrY
CameraDefsC.h, [263](#)

PixelFormat_YCbCr422_12
CameraDefsC.h, [263](#)

PixelFormat_YCbCr422_12_CbYCrY
CameraDefsC.h, [263](#)

PixelFormat_YCbCr422_12p
CameraDefsC.h, [263](#)

PixelFormat_YCbCr422_12p_CbYCrY
CameraDefsC.h, [263](#)

PixelFormat_YCbCr422_8
CameraDefsC.h, [259](#)

PixelFormat_YCbCr422_8_CbYCrY
CameraDefsC.h, [263](#)

PixelFormat_YCbCr601_10_CbYCr
CameraDefsC.h, [263](#)

PixelFormat_YCbCr601_10p_CbYCr
CameraDefsC.h, [263](#)

PixelFormat_YCbCr601_12_CbYCr
CameraDefsC.h, [263](#)

PixelFormat_YCbCr601_12p_CbYCr
CameraDefsC.h, [263](#)

PixelFormat_YCbCr601_411_8_CbYYCrYY
CameraDefsC.h, [263](#)

PixelFormat_YCbCr601_422_10
CameraDefsC.h, [263](#)

PixelFormat_YCbCr601_422_10_CbYCrY
CameraDefsC.h, [263](#)

PixelFormat_YCbCr601_422_10p
CameraDefsC.h, [263](#)

- PixelFormat_YCbCr601_422_10p_CbYCrY
 - CameraDefsC.h, [263](#)
- PixelFormat_YCbCr601_422_12
 - CameraDefsC.h, [263](#)
- PixelFormat_YCbCr601_422_12_CbYCrY
 - CameraDefsC.h, [263](#)
- PixelFormat_YCbCr601_422_12p
 - CameraDefsC.h, [263](#)
- PixelFormat_YCbCr601_422_12p_CbYCrY
 - CameraDefsC.h, [263](#)
- PixelFormat_YCbCr601_422_8
 - CameraDefsC.h, [263](#)
- PixelFormat_YCbCr601_422_8_CbYCrY
 - CameraDefsC.h, [263](#)
- PixelFormat_YCbCr601_8_CbYCr
 - CameraDefsC.h, [263](#)
- PixelFormat_YCbCr709_10_CbYCr
 - CameraDefsC.h, [263](#)
- PixelFormat_YCbCr709_10p_CbYCr
 - CameraDefsC.h, [263](#)
- PixelFormat_YCbCr709_12_CbYCr
 - CameraDefsC.h, [263](#)
- PixelFormat_YCbCr709_12p_CbYCr
 - CameraDefsC.h, [263](#)
- PixelFormat_YCbCr709_411_8_CbYYCrYY
 - CameraDefsC.h, [264](#)
- PixelFormat_YCbCr709_422_10
 - CameraDefsC.h, [264](#)
- PixelFormat_YCbCr709_422_10_CbYCrY
 - CameraDefsC.h, [264](#)
- PixelFormat_YCbCr709_422_10p
 - CameraDefsC.h, [264](#)
- PixelFormat_YCbCr709_422_10p_CbYCrY
 - CameraDefsC.h, [264](#)
- PixelFormat_YCbCr709_422_12
 - CameraDefsC.h, [264](#)
- PixelFormat_YCbCr709_422_12_CbYCrY
 - CameraDefsC.h, [264](#)
- PixelFormat_YCbCr709_422_12p
 - CameraDefsC.h, [264](#)
- PixelFormat_YCbCr709_422_12p_CbYCrY
 - CameraDefsC.h, [264](#)
- PixelFormat_YCbCr709_422_8
 - CameraDefsC.h, [264](#)
- PixelFormat_YCbCr709_422_8_CbYCrY
 - CameraDefsC.h, [264](#)
- PixelFormat_YCbCr709_8_CbYCr
 - CameraDefsC.h, [263](#)
- PixelFormat_YCbCr8
 - CameraDefsC.h, [259](#)
- PixelFormat_YCbCr8_CbYCr
 - CameraDefsC.h, [263](#)
- PixelFormat_YUV411_8_YYYYYY
 - CameraDefsC.h, [264](#)
- PixelFormat_YUV411Packed
 - CameraDefsC.h, [259](#)
- PixelFormat_YUV422_8
 - CameraDefsC.h, [264](#)
- PixelFormat_YUV422_8_UYVY
 - CameraDefsC.h, [264](#)
- PixelFormat_YUV422Packed
 - CameraDefsC.h, [259](#)
- PixelFormat_YUV444Packed
 - CameraDefsC.h, [259](#)
- PixelFormat_YUV8_UYV
 - CameraDefsC.h, [264](#)
- PixelFormatInfoID
 - quickSpin, [116](#)
- PixelFormatInfoSelector
 - quickSpin, [116](#)
- PixelFormatInfoSelector_B10
 - CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_B12
 - CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_B16
 - CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_B8
 - CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BayerBG10
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerBG10p
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerBG12
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerBG12p
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerBG16
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerBG8
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerGB10
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerGB10p
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerGB12
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerGB12p
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerGB16
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerGB8
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerGR10
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerGR10p
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerGR12
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerGR12p
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerGR16
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerGR8
 - CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerRG10
 - CameraDefsC.h, [265](#)

- PixelFormatInfoSelector_BayerRG10p
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerRG12
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerRG12p
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerRG16
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerRG8
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_BayerRGPolarized10p
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_BayerRGPolarized12p
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_BayerRGPolarized16
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_BayerRGPolarized8
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_BGR10
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGR10p
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGR12
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGR12p
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGR14
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGR16
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGR565p
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGR8
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGRa10
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGRa10p
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGRa12
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGRa12p
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGRa14
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGRa16
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BGRa8
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_BiColorBGRG10
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_BiColorBGRG10p
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_BiColorBGRG12
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_BiColorBGRG12p
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_BiColorBGRG8
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_BiColorRGBG10
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_BiColorRGBG10p
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_BiColorRGBG12
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_BiColorRGBG12p
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_BiColorRGBG8
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Confidence1
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Confidence16
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Confidence1p
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Confidence32f
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Confidence8
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_A10p
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_A12p
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_A16
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_A32f
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_A8
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_ABC10p
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_Coord3D_ABC10p_Planar
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_Coord3D_ABC12p
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_Coord3D_ABC12p_Planar
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_Coord3D_ABC16
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_ABC16_Planar
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_ABC32f
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_ABC32f_Planar
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_ABC8
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_Coord3D_ABC8_Planar
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_Coord3D_AC10p
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_AC10p_Planar
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_AC12p
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_AC12p_Planar
CameraDefsC.h, [267](#)

- PixelFormatInfoSelector_Coord3D_AC16
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_AC16_Planar
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_AC32f
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_AC32f_Planar
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_AC8
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_AC8_Planar
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_B10p
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_B12p
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_B16
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_B32f
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_B8
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_C10p
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_C12p
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_C16
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_C32f
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_Coord3D_C8
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_G10
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_G12
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_G16
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_G8
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_JPEGColor8
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_JPEGMono8
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_LLCBayerRG8
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_LLCMono8
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_Mono10
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_Mono10p
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_Mono12
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_Mono12p
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_Mono14
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_Mono16
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_Mono16s
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_Mono1p
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_Mono2p
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_Mono32f
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_Mono4p
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_Mono8
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_Mono8s
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_Polarized10p
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_Polarized12p
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_Polarized16
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_Polarized8
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_R10
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_R12
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_R16
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_R8
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_RGB10
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_RGB10_Planar
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_RGB10p
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_RGB10p32
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_RGB12
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_RGB12_Planar
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_RGB12p
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_RGB14
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_RGB16
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_RGB16_Planar
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_RGB16s
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_RGB32f
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_RGB565p
CameraDefsC.h, [266](#)

- PixelFormatInfoSelector_RGB8
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_RGB8_Planar
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_RGBa10
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_RGBa10p
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_RGBa12
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_RGBa12p
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_RGBa14
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_RGBa16
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_RGBa32f
CameraDefsC.h, [266](#)
- PixelFormatInfoSelector_RGBa8
CameraDefsC.h, [265](#)
- PixelFormatInfoSelector_SCF1WBWG10
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_SCF1WBWG10p
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WBWG12
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WBWG12p
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WBWG14
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WBWG16
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WBWG8
CameraDefsC.h, [267](#)
- PixelFormatInfoSelector_SCF1WGWB10
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WGWB10p
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WGWB12
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WGWB12p
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WGWB14
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WGWB16
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WGWB8
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WGWR10
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WGWR10p
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WGWR12
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WGWR12p
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WGWR14
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WGWR16
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WGWR8
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WRWG10
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WRWG10p
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WRWG12
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WRWG12p
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WRWG14
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WRWG16
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_SCF1WRWG8
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_YCbCr10_CbYCr
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_YCbCr10p_CbYCr
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_YCbCr12_CbYCr
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr12p_CbYCr
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr411_8
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr411_8_CbYYCrYY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr422_10
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr422_10_CbYCrY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr422_10p
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr422_10p_CbYCrY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr422_12
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr422_12_CbYCrY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr422_12p
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr422_12p_CbYCrY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr422_8
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr422_8_CbYCrY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_10_CbYCr
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_10p_CbYCr
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_12_CbYCr
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_12p_CbYCr
CameraDefsC.h, [269](#)

- PixelFormatInfoSelector_YCbCr601_411_8_CbYYCrYY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_422_10
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_422_10_CbYCrY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_422_10p
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_422_10p_CbYCrY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_422_12
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_422_12_CbYCrY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_422_12p
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_422_12p_CbYCrY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_422_8
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_422_8_CbYCrY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr601_8_CbYCr
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr709_10_CbYCr
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr709_10p_CbYCr
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr709_12_CbYCr
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr709_12p_CbYCr
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr709_411_8_CbYYCrYY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr709_422_10
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr709_422_10_CbYCrY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr709_422_10p
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr709_422_10p_CbYCrY
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_YCbCr709_422_12
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_YCbCr709_422_12_CbYCrY
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_YCbCr709_422_12p
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_YCbCr709_422_12p_CbYCrY
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_YCbCr709_422_8
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr709_422_8_CbYCrY
CameraDefsC.h, [269](#)
- PixelFormatInfoSelector_YCbCr8
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_YCbCr8_CbYCr
CameraDefsC.h, [268](#)
- PixelFormatInfoSelector_YUV411_8_UYYVYY
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_YUV422_8
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_YUV422_8_UYVY
CameraDefsC.h, [270](#)
- PixelFormatInfoSelector_YUV8_UYV
CameraDefsC.h, [270](#)
- PixelSize
quickSpin, [116](#)
- PixelSize_Bpp1
CameraDefsC.h, [270](#)
- PixelSize_Bpp10
CameraDefsC.h, [270](#)
- PixelSize_Bpp12
CameraDefsC.h, [270](#)
- PixelSize_Bpp14
CameraDefsC.h, [270](#)
- PixelSize_Bpp16
CameraDefsC.h, [270](#)
- PixelSize_Bpp2
CameraDefsC.h, [270](#)
- PixelSize_Bpp20
CameraDefsC.h, [270](#)
- PixelSize_Bpp24
CameraDefsC.h, [270](#)
- PixelSize_Bpp30
CameraDefsC.h, [270](#)
- PixelSize_Bpp32
CameraDefsC.h, [270](#)
- PixelSize_Bpp36
CameraDefsC.h, [271](#)
- PixelSize_Bpp4
CameraDefsC.h, [270](#)
- PixelSize_Bpp48
CameraDefsC.h, [271](#)
- PixelSize_Bpp64
CameraDefsC.h, [271](#)
- PixelSize_Bpp8
CameraDefsC.h, [270](#)
- PixelSize_Bpp96
CameraDefsC.h, [271](#)
- POEStatus
quickSpinTLInterface, [146](#)
- POEStatus_NotSupported
TransportLayerDefsC.h, [480](#)
- POEStatus_PowerOff
TransportLayerDefsC.h, [480](#)
- POEStatus_PowerOn
TransportLayerDefsC.h, [480](#)
- PortNode
SpinnakerGenApiDefsC.h, [468](#)
- PowerSupplyCurrent
quickSpin, [116](#)
- PowerSupplyVoltage
quickSpin, [116](#)

- progressive
 - spinJPEGOption, 166
- PureNumber
 - SpinnakerGenApiDefsC.h, 468
- quality
 - spinJPEGOption, 166
 - spinJPG2Option, 167
 - spinMJPEGOption, 169
- quickSpin, 36
 - AasRoiEnable, 48
 - AasRoiHeight, 48
 - AasRoiOffsetX, 48
 - AasRoiOffsetY, 48
 - AasRoiWidth, 48
 - AcquisitionAbort, 49
 - AcquisitionArm, 49
 - AcquisitionBurstFrameCount, 49
 - AcquisitionFrameCount, 49
 - AcquisitionFrameRate, 49
 - AcquisitionFrameRateEnable, 49
 - AcquisitionLineRate, 49
 - AcquisitionMode, 49
 - AcquisitionResultingFrameRate, 50
 - AcquisitionStart, 50
 - AcquisitionStatus, 50
 - AcquisitionStatusSelector, 50
 - AcquisitionStop, 50
 - ActionDeviceKey, 50
 - ActionGroupKey, 50
 - ActionGroupMask, 50
 - ActionQueueSize, 51
 - ActionSelector, 51
 - ActionUnconditionalMode, 51
 - AdaptiveCompressionEnable, 51
 - AdcBitDepth, 51
 - aPAUSEMACCtrlFramesReceived, 51
 - aPAUSEMACCtrlFramesTransmitted, 51
 - AutoAlgorithmSelector, 51
 - AutoExposureControlLoopDamping, 52
 - AutoExposureControlPriority, 52
 - AutoExposureEVCompensation, 52
 - AutoExposureExposureTimeLowerLimit, 52
 - AutoExposureExposureTimeUpperLimit, 52
 - AutoExposureGainLowerLimit, 52
 - AutoExposureGainUpperLimit, 52
 - AutoExposureGreyValueLowerLimit, 52
 - AutoExposureGreyValueUpperLimit, 53
 - AutoExposureLightingMode, 53
 - AutoExposureMeteringMode, 53
 - AutoExposureTargetGreyValue, 53
 - AutoExposureTargetGreyValueAuto, 53
 - BalanceRatio, 53
 - BalanceRatioSelector, 53
 - BalanceWhiteAuto, 53
 - BalanceWhiteAutoDamping, 54
 - BalanceWhiteAutoLowerLimit, 54
 - BalanceWhiteAutoProfile, 54
 - BalanceWhiteAutoUpperLimit, 54
 - BinningHorizontal, 54
 - BinningHorizontalMode, 54
 - BinningSelector, 54
 - BinningVertical, 54
 - BinningVerticalMode, 55
 - BlackLevel, 55
 - BlackLevelAuto, 55
 - BlackLevelAutoBalance, 55
 - BlackLevelClampingEnable, 55
 - BlackLevelRaw, 55
 - BlackLevelSelector, 55
 - ChunkBlackLevel, 55
 - ChunkBlackLevelSelector, 56
 - ChunkCompressionMode, 56
 - ChunkCompressionRatio, 56
 - ChunkCounterSelector, 56
 - ChunkCounterValue, 56
 - ChunkCRC, 56
 - ChunkEnable, 56
 - ChunkEncoderSelector, 56
 - ChunkEncoderStatus, 57
 - ChunkEncoderValue, 57
 - ChunkExposureEndLineStatusAll, 57
 - ChunkExposureTime, 57
 - ChunkExposureTimeSelector, 57
 - ChunkFrameID, 57
 - ChunkGain, 57
 - ChunkGainSelector, 57
 - ChunkHeight, 58
 - ChunkImage, 58
 - ChunkImageComponent, 58
 - ChunkInferenceBoundingBoxResult, 58
 - ChunkInferenceConfidence, 58
 - ChunkInferenceFrameId, 58
 - ChunkInferenceResult, 58
 - ChunkLinePitch, 58
 - ChunkLineStatusAll, 59
 - ChunkModeActive, 59
 - ChunkOffsetX, 59
 - ChunkOffsetY, 59
 - ChunkPartSelector, 59
 - ChunkPixelDynamicRangeMax, 59
 - ChunkPixelDynamicRangeMin, 59
 - ChunkPixelFormat, 59
 - ChunkRegionID, 60
 - ChunkScan3dAxisMax, 60
 - ChunkScan3dAxisMin, 60
 - ChunkScan3dCoordinateOffset, 60
 - ChunkScan3dCoordinateReferenceSelector, 60
 - ChunkScan3dCoordinateReferenceValue, 60
 - ChunkScan3dCoordinateScale, 60
 - ChunkScan3dCoordinateSelector, 60
 - ChunkScan3dCoordinateSystem, 61
 - ChunkScan3dCoordinateSystemReference, 61
 - ChunkScan3dCoordinateTransformSelector, 61
 - ChunkScan3dDistanceUnit, 61
 - ChunkScan3dInvalidDataFlag, 61
 - ChunkScan3dInvalidDataValue, 61

ChunkScan3dOutputMode, 61
ChunkScan3dTransformValue, 61
ChunkScanLineSelector, 62
ChunkSelector, 62
ChunkSequencerSetActive, 62
ChunkSerialData, 62
ChunkSerialDataLength, 62
ChunkSerialReceiveOverflow, 62
ChunkSourceID, 62
ChunkStreamChannelID, 62
ChunkTimerSelector, 63
ChunkTimerValue, 63
ChunkTimestamp, 63
ChunkTimestampLatchValue, 63
ChunkTransferBlockID, 63
ChunkTransferQueueCurrentBlockCount, 63
ChunkTransferStreamID, 63
ChunkWidth, 63
CIConfiguration, 64
CITimeSlotsCount, 64
ColorTransformationEnable, 64
ColorTransformationSelector, 64
ColorTransformationValue, 64
ColorTransformationValueSelector, 64
CompressionRatio, 64
CompressionSaturationPriority, 64
CounterDelay, 65
CounterDuration, 65
CounterEventActivation, 65
CounterEventSource, 65
CounterReset, 65
CounterResetActivation, 65
CounterResetSource, 65
CounterSelector, 65
CounterStatus, 66
CounterTriggerActivation, 66
CounterTriggerSource, 66
CounterValue, 66
CounterValueAtReset, 66
CxpConnectionSelector, 66
CxpConnectionTestErrorCount, 66
CxpConnectionTestMode, 66
CxpConnectionTestPacketCount, 67
CxpLinkConfiguration, 67
CxpLinkConfigurationPreferred, 67
CxpLinkConfigurationStatus, 67
CxpPoCxpAuto, 67
CxpPoCxpStatus, 67
CxpPoCxpTripReset, 67
CxpPoCxpTurnOff, 67
DecimationHorizontal, 68
DecimationHorizontalMode, 68
DecimationSelector, 68
DecimationVertical, 68
DecimationVerticalMode, 68
DefectCorrectionMode, 68
DefectCorrectStaticEnable, 68
DefectTableApply, 68
DefectTableCoordinateX, 69
DefectTableCoordinateY, 69
DefectTableFactoryRestore, 69
DefectTableIndex, 69
DefectTablePixelCount, 69
DefectTableSave, 69
Deinterlacing, 69
DeviceCharacterSet, 69
DeviceClockFrequency, 70
DeviceClockSelector, 70
DeviceConnectionSelector, 70
DeviceConnectionSpeed, 70
DeviceConnectionStatus, 70
DeviceEventChannelCount, 70
DeviceFamilyName, 70
DeviceFeaturePersistenceEnd, 70
DeviceFeaturePersistenceStart, 71
DeviceFirmwareVersion, 71
DeviceGenCPVersionMajor, 71
DeviceGenCPVersionMinor, 71
DeviceID, 71
DeviceIndicatorMode, 71
DeviceLinkBandwidthReserve, 71
DeviceLinkCommandTimeout, 71
DeviceLinkConnectionCount, 72
DeviceLinkCurrentThroughput, 72
DeviceLinkHeartbeatMode, 72
DeviceLinkHeartbeatTimeout, 72
DeviceLinkSelector, 72
DeviceLinkSpeed, 72
DeviceLinkThroughputLimit, 72
DeviceLinkThroughputLimitMode, 72
DeviceManifestEntrySelector, 73
DeviceManifestPrimaryURL, 73
DeviceManifestSchemaMajorVersion, 73
DeviceManifestSchemaMinorVersion, 73
DeviceManifestSecondaryURL, 73
DeviceManifestXMLMajorVersion, 73
DeviceManifestXMLMinorVersion, 73
DeviceManifestXMLSubMinorVersion, 73
DeviceManufacturerInfo, 74
DeviceMaxThroughput, 74
DeviceModelName, 74
DevicePowerSupplySelector, 74
DeviceRegistersCheck, 74
DeviceRegistersEndianness, 74
DeviceRegistersStreamingEnd, 74
DeviceRegistersStreamingStart, 74
DeviceRegistersValid, 75
DeviceReset, 75
DeviceScanType, 75
DeviceSerialNumber, 75
DeviceSerialPortBaudRate, 75
DeviceSerialPortSelector, 75
DeviceSFNCVersionMajor, 75
DeviceSFNCVersionMinor, 75
DeviceSFNCVersionSubMinor, 76
DeviceStreamChannelCount, 76

DeviceStreamChannelEndianness, 76
DeviceStreamChannelLink, 76
DeviceStreamChannelPacketSize, 76
DeviceStreamChannelSelector, 76
DeviceStreamChannelType, 76
DeviceTapGeometry, 76
DeviceTemperature, 77
DeviceTemperatureSelector, 77
DeviceTLType, 77
DeviceTLVersionMajor, 77
DeviceTLVersionMinor, 77
DeviceTLVersionSubMinor, 77
DeviceType, 77
DeviceUptime, 77
DeviceUserID, 78
DeviceVendorName, 78
DeviceVersion, 78
EncoderDivider, 78
EncoderMode, 78
EncoderOutputMode, 78
EncoderReset, 78
EncoderResetActivation, 78
EncoderResetSource, 79
EncoderSelector, 79
EncoderSourceA, 79
EncoderSourceB, 79
EncoderStatus, 79
EncoderTimeout, 79
EncoderValue, 79
EncoderValueAtReset, 79
EnumerationCount, 80
EventAcquisitionEnd, 80
EventAcquisitionEndFrameID, 80
EventAcquisitionEndTimestamp, 80
EventAcquisitionError, 80
EventAcquisitionErrorFrameID, 80
EventAcquisitionErrorTimestamp, 80
EventAcquisitionStart, 80
EventAcquisitionStartFrameID, 81
EventAcquisitionStartTimestamp, 81
EventAcquisitionTransferEnd, 81
EventAcquisitionTransferEndFrameID, 81
EventAcquisitionTransferEndTimestamp, 81
EventAcquisitionTransferStart, 81
EventAcquisitionTransferStartFrameID, 81
EventAcquisitionTransferStartTimestamp, 81
EventAcquisitionTrigger, 82
EventAcquisitionTriggerFrameID, 82
EventAcquisitionTriggerTimestamp, 82
EventActionLate, 82
EventActionLateFrameID, 82
EventActionLateTimestamp, 82
EventCounter0End, 82
EventCounter0EndFrameID, 82
EventCounter0EndTimestamp, 83
EventCounter0Start, 83
EventCounter0StartFrameID, 83
EventCounter0StartTimestamp, 83
EventCounter1End, 83
EventCounter1EndFrameID, 83
EventCounter1EndTimestamp, 83
EventCounter1Start, 83
EventCounter1StartFrameID, 84
EventCounter1StartTimestamp, 84
EventEncoder0Restarted, 84
EventEncoder0RestartedFrameID, 84
EventEncoder0RestartedTimestamp, 84
EventEncoder0Stopped, 84
EventEncoder0StoppedFrameID, 84
EventEncoder0StoppedTimestamp, 84
EventEncoder1Restarted, 85
EventEncoder1RestartedFrameID, 85
EventEncoder1RestartedTimestamp, 85
EventEncoder1Stopped, 85
EventEncoder1StoppedFrameID, 85
EventEncoder1StoppedTimestamp, 85
EventError, 85
EventErrorCode, 85
EventErrorFrameID, 86
EventErrorTimestamp, 86
EventExposureEnd, 86
EventExposureEndFrameID, 86
EventExposureEndTimestamp, 86
EventExposureStart, 86
EventExposureStartFrameID, 86
EventExposureStartTimestamp, 86
EventFrameBurstEnd, 87
EventFrameBurstEndFrameID, 87
EventFrameBurstEndTimestamp, 87
EventFrameBurstStart, 87
EventFrameBurstStartFrameID, 87
EventFrameBurstStartTimestamp, 87
EventFrameEnd, 87
EventFrameEndFrameID, 87
EventFrameEndTimestamp, 88
EventFrameStart, 88
EventFrameStartFrameID, 88
EventFrameStartTimestamp, 88
EventFrameTransferEnd, 88
EventFrameTransferEndFrameID, 88
EventFrameTransferEndTimestamp, 88
EventFrameTransferStart, 88
EventFrameTransferStartFrameID, 89
EventFrameTransferStartTimestamp, 89
EventFrameTrigger, 89
EventFrameTriggerFrameID, 89
EventFrameTriggerTimestamp, 89
EventLine0AnyEdge, 89
EventLine0AnyEdgeFrameID, 89
EventLine0AnyEdgeTimestamp, 89
EventLine0FallingEdge, 90
EventLine0FallingEdgeFrameID, 90
EventLine0FallingEdgeTimestamp, 90
EventLine0RisingEdge, 90
EventLine0RisingEdgeFrameID, 90
EventLine0RisingEdgeTimestamp, 90

- EventLine1AnyEdge, 90
- EventLine1AnyEdgeFrameID, 90
- EventLine1AnyEdgeTimestamp, 91
- EventLine1FallingEdge, 91
- EventLine1FallingEdgeFrameID, 91
- EventLine1FallingEdgeTimestamp, 91
- EventLine1RisingEdge, 91
- EventLine1RisingEdgeFrameID, 91
- EventLine1RisingEdgeTimestamp, 91
- EventLinkSpeedChange, 91
- EventLinkSpeedChangeFrameID, 92
- EventLinkSpeedChangeTimestamp, 92
- EventLinkTrigger0, 92
- EventLinkTrigger0FrameID, 92
- EventLinkTrigger0Timestamp, 92
- EventLinkTrigger1, 92
- EventLinkTrigger1FrameID, 92
- EventLinkTrigger1Timestamp, 92
- EventNotification, 93
- EventSelector, 93
- EventSequencerSetChange, 93
- EventSequencerSetChangeFrameID, 93
- EventSequencerSetChangeTimestamp, 93
- EventSerialData, 93
- EventSerialDataLength, 93
- EventSerialPortReceive, 93
- EventSerialPortReceiveTimestamp, 94
- EventSerialReceiveOverflow, 94
- EventStream0TransferBlockEnd, 94
- EventStream0TransferBlockEndFrameID, 94
- EventStream0TransferBlockEndTimestamp, 94
- EventStream0TransferBlockStart, 94
- EventStream0TransferBlockStartFrameID, 94
- EventStream0TransferBlockStartTimestamp, 94
- EventStream0TransferBlockTrigger, 95
- EventStream0TransferBlockTriggerFrameID, 95
- EventStream0TransferBlockTriggerTimestamp, 95
- EventStream0TransferBurstEnd, 95
- EventStream0TransferBurstEndFrameID, 95
- EventStream0TransferBurstEndTimestamp, 95
- EventStream0TransferBurstStart, 95
- EventStream0TransferBurstStartFrameID, 95
- EventStream0TransferBurstStartTimestamp, 96
- EventStream0TransferEnd, 96
- EventStream0TransferEndFrameID, 96
- EventStream0TransferEndTimestamp, 96
- EventStream0TransferOverflow, 96
- EventStream0TransferOverflowFrameID, 96
- EventStream0TransferOverflowTimestamp, 96
- EventStream0TransferPause, 96
- EventStream0TransferPauseFrameID, 97
- EventStream0TransferPauseTimestamp, 97
- EventStream0TransferResume, 97
- EventStream0TransferResumeFrameID, 97
- EventStream0TransferResumeTimestamp, 97
- EventStream0TransferStart, 97
- EventStream0TransferStartFrameID, 97
- EventStream0TransferStartTimestamp, 97
- EventTest, 98
- EventTestTimestamp, 98
- EventTimer0End, 98
- EventTimer0EndFrameID, 98
- EventTimer0EndTimestamp, 98
- EventTimer0Start, 98
- EventTimer0StartFrameID, 98
- EventTimer0StartTimestamp, 98
- EventTimer1End, 99
- EventTimer1EndFrameID, 99
- EventTimer1EndTimestamp, 99
- EventTimer1Start, 99
- EventTimer1StartFrameID, 99
- EventTimer1StartTimestamp, 99
- ExposureActiveMode, 99
- ExposureAuto, 99
- ExposureMode, 100
- ExposureTime, 100
- ExposureTimeMode, 100
- ExposureTimeSelector, 100
- FactoryReset, 100
- FileAccessBuffer, 100
- FileAccessLength, 100
- FileAccessOffset, 100
- FileOpenMode, 101
- FileOperationExecute, 101
- FileOperationResult, 101
- FileOperationSelector, 101
- FileOperationStatus, 101
- FileSelector, 101
- FileSize, 101
- Gain, 101
- GainAuto, 102
- GainAutoBalance, 102
- GainSelector, 102
- Gamma, 102
- GammaEnable, 102
- GevActiveLinkCount, 102
- GevCCP, 102
- GevCurrentDefaultGateway, 102
- GevCurrentIPAddress, 103
- GevCurrentIPConfigurationDHCP, 103
- GevCurrentIPConfigurationLLA, 103
- GevCurrentIPConfigurationPersistentIP, 103
- GevCurrentPhysicalLinkConfiguration, 103
- GevCurrentSubnetMask, 103
- GevDiscoveryAckDelay, 103
- GevFirstURL, 103
- GevGVCPExtendedStatusCodes, 104
- GevGVCPExtendedStatusCodesSelector, 104
- GevGVCPHeartbeatDisable, 104
- GevGVCPPendingAck, 104
- GevGVCPPendingTimeout, 104
- GevGVSPExtendedIDMode, 104
- GevHeartbeatTimeout, 104
- GevIEEE1588, 104
- GevIEEE1588ClockAccuracy, 105
- GevIEEE1588Mode, 105

- GevIEEE1588Status, [105](#)
- GevInterfaceSelector, [105](#)
- GevIPConfigurationStatus, [105](#)
- GevMACAddress, [105](#)
- GevMCDA, [105](#)
- GevMCPHostPort, [105](#)
- GevMCRC, [106](#)
- GevMCSP, [106](#)
- GevMCTT, [106](#)
- GevNumberOfInterfaces, [106](#)
- GevPAUSEFrameReception, [106](#)
- GevPAUSEFrameTransmission, [106](#)
- GevPersistentDefaultGateway, [106](#)
- GevPersistentIPAddress, [106](#)
- GevPersistentSubnetMask, [107](#)
- GevPhysicalLinkConfiguration, [107](#)
- GevPrimaryApplicationIPAddress, [107](#)
- GevPrimaryApplicationSocket, [107](#)
- GevPrimaryApplicationSwitchoverKey, [107](#)
- GevSCCFGAllInTransmission, [107](#)
- GevSCCFGExtendedChunkData, [107](#)
- GevSCCFGPacketResendDestination, [107](#)
- GevSCCFGUnconditionalStreaming, [108](#)
- GevSCDA, [108](#)
- GevSCPD, [108](#)
- GevSCPDirection, [108](#)
- GevSCPHostPort, [108](#)
- GevSCPInterfaceIndex, [108](#)
- GevSCPSBigEndian, [108](#)
- GevSCPSDoNotFragment, [108](#)
- GevSCPSFireTestPacket, [109](#)
- GevSCPSPacketSize, [109](#)
- GevSCSP, [109](#)
- GevSCZoneConfigurationLock, [109](#)
- GevSCZoneCount, [109](#)
- GevSCZoneDirectionAll, [109](#)
- GevSecondURL, [109](#)
- GevStreamChannelSelector, [109](#)
- GevSupportedOption, [110](#)
- GevSupportedOptionSelector, [110](#)
- GevTimestampTickFrequency, [110](#)
- GuiXmlManifestAddress, [110](#)
- Height, [110](#)
- HeightMax, [110](#)
- ImageComponentEnable, [110](#)
- ImageComponentSelector, [110](#)
- ImageCompressionBitrate, [111](#)
- ImageCompressionJPEGFormatOption, [111](#)
- ImageCompressionMode, [111](#)
- ImageCompressionQuality, [111](#)
- ImageCompressionRateOption, [111](#)
- IspEnable, [111](#)
- LineFilterWidth, [111](#)
- LineFormat, [111](#)
- LineInputFilterSelector, [112](#)
- LineInverter, [112](#)
- LineMode, [112](#)
- LinePitch, [112](#)
- LineSelector, [112](#)
- LineSource, [112](#)
- LineStatus, [112](#)
- LineStatusAll, [112](#)
- LinkErrorCount, [113](#)
- LinkUptime, [113](#)
- LogicBlockLUTInputActivation, [113](#)
- LogicBlockLUTInputSelector, [113](#)
- LogicBlockLUTInputSource, [113](#)
- LogicBlockLUTOutputValue, [113](#)
- LogicBlockLUTOutputValueAll, [113](#)
- LogicBlockLUTRowIndex, [113](#)
- LogicBlockLUTSelector, [114](#)
- LogicBlockSelector, [114](#)
- LUTEnable, [114](#)
- LUTIndex, [114](#)
- LUTSelector, [114](#)
- LUTValue, [114](#)
- LUTValueAll, [114](#)
- MaxDeviceResetTime, [114](#)
- OffsetX, [115](#)
- OffsetY, [115](#)
- PacketResendRequestCount, [115](#)
- PayloadSize, [115](#)
- PixelColorFilter, [115](#)
- PixelDynamicRangeMax, [115](#)
- PixelDynamicRangeMin, [115](#)
- PixelFormat, [115](#)
- PixelFormatInfoID, [116](#)
- PixelFormatInfoSelector, [116](#)
- PixelSize, [116](#)
- PowerSupplyCurrent, [116](#)
- PowerSupplyVoltage, [116](#)
- RegionDestination, [116](#)
- RegionMode, [116](#)
- RegionSelector, [116](#)
- ReverseX, [117](#)
- ReverseY, [117](#)
- RgbTransformLightSource, [117](#)
- Saturation, [117](#)
- SaturationEnable, [117](#)
- Scan3dAxisMax, [117](#)
- Scan3dAxisMin, [117](#)
- Scan3dCoordinateOffset, [117](#)
- Scan3dCoordinateReferenceSelector, [118](#)
- Scan3dCoordinateReferenceValue, [118](#)
- Scan3dCoordinateScale, [118](#)
- Scan3dCoordinateSelector, [118](#)
- Scan3dCoordinateSystem, [118](#)
- Scan3dCoordinateSystemReference, [118](#)
- Scan3dCoordinateTransformSelector, [118](#)
- Scan3dDistanceUnit, [118](#)
- Scan3dInvalidDataFlag, [119](#)
- Scan3dInvalidDataValue, [119](#)
- Scan3dOutputMode, [119](#)
- Scan3dTransformValue, [119](#)
- SensorDescription, [119](#)
- SensorDigitizationTaps, [119](#)

- SensorHeight, [119](#)
- SensorShutterMode, [119](#)
- SensorTaps, [120](#)
- SensorWidth, [120](#)
- SequencerConfigurationMode, [120](#)
- SequencerConfigurationValid, [120](#)
- SequencerFeatureEnable, [120](#)
- SequencerMode, [120](#)
- SequencerPathSelector, [120](#)
- SequencerSetActive, [120](#)
- SequencerSetLoad, [121](#)
- SequencerSetNext, [121](#)
- SequencerSetSave, [121](#)
- SequencerSetSelector, [121](#)
- SequencerSetStart, [121](#)
- SequencerSetValid, [121](#)
- SequencerTriggerActivation, [121](#)
- SequencerTriggerSource, [121](#)
- SerialPortBaudRate, [122](#)
- SerialPortDataBits, [122](#)
- SerialPortParity, [122](#)
- SerialPortSelector, [122](#)
- SerialPortSource, [122](#)
- SerialPortStopBits, [122](#)
- SerialReceiveFramingErrorCount, [122](#)
- SerialReceiveParityErrorCount, [122](#)
- SerialReceiveQueueClear, [123](#)
- SerialReceiveQueueCurrentCharacterCount, [123](#)
- SerialReceiveQueueMaxCharacterCount, [123](#)
- SerialTransmitQueueCurrentCharacterCount, [123](#)
- SerialTransmitQueueMaxCharacterCount, [123](#)
- Sharpening, [123](#)
- SharpeningAuto, [123](#)
- SharpeningEnable, [123](#)
- SharpeningThreshold, [124](#)
- SoftwareSignalPulse, [124](#)
- SoftwareSignalSelector, [124](#)
- SourceCount, [124](#)
- SourceSelector, [124](#)
- Test0001, [124](#)
- TestEventGenerate, [124](#)
- TestPattern, [124](#)
- TestPatternGeneratorSelector, [125](#)
- TestPendingAck, [125](#)
- TimerDelay, [125](#)
- TimerDuration, [125](#)
- TimerReset, [125](#)
- TimerSelector, [125](#)
- TimerStatus, [125](#)
- TimerTriggerActivation, [125](#)
- TimerTriggerSource, [126](#)
- TimerValue, [126](#)
- Timestamp, [126](#)
- TimestampLatch, [126](#)
- TimestampLatchValue, [126](#)
- TimestampReset, [126](#)
- TLParamsLocked, [126](#)
- TransferAbort, [126](#)
- TransferBlockCount, [127](#)
- TransferBurstCount, [127](#)
- TransferComponentSelector, [127](#)
- TransferControlMode, [127](#)
- TransferOperationMode, [127](#)
- TransferPause, [127](#)
- TransferQueueCurrentBlockCount, [127](#)
- TransferQueueMaxBlockCount, [127](#)
- TransferQueueMode, [128](#)
- TransferQueueOverflowCount, [128](#)
- TransferResume, [128](#)
- TransferSelector, [128](#)
- TransferStart, [128](#)
- TransferStatus, [128](#)
- TransferStatusSelector, [128](#)
- TransferStop, [128](#)
- TransferStreamChannel, [129](#)
- TransferTriggerActivation, [129](#)
- TransferTriggerMode, [129](#)
- TransferTriggerSelector, [129](#)
- TransferTriggerSource, [129](#)
- TriggerActivation, [129](#)
- TriggerDelay, [129](#)
- TriggerDivider, [129](#)
- TriggerEventTest, [130](#)
- TriggerMode, [130](#)
- TriggerMultiplier, [130](#)
- TriggerOverlap, [130](#)
- TriggerSelector, [130](#)
- TriggerSoftware, [130](#)
- TriggerSource, [130](#)
- UserOutputSelector, [130](#)
- UserOutputValue, [131](#)
- UserOutputValueAll, [131](#)
- UserOutputValueAllMask, [131](#)
- UserSetDefault, [131](#)
- UserSetFeatureEnable, [131](#)
- UserSetLoad, [131](#)
- UserSetSave, [131](#)
- UserSetSelector, [131](#)
- V3_3Enable, [132](#)
- WhiteClip, [132](#)
- WhiteClipSelector, [132](#)
- Width, [132](#)
- WidthMax, [132](#)
- QuickSpin Access, [22](#)
- quickSpinBooleanNode
 - QuickSpinDefsC.h, [296](#)
- QuickSpinC.h
 - quickSpinInit, [294](#)
 - quickSpinInitEx, [294](#)
 - quickSpinTLDeviceInit, [295](#)
 - quickSpinTLInterfaceInit, [295](#)
 - quickSpinTLStreamInit, [295](#)
 - quickSpinTLSystemInit, [295](#)
- quickSpinCommandNode
 - QuickSpinDefsC.h, [297](#)
- QuickSpinDefsC.h

- quickSpinBooleanNode, 296
- quickSpinCommandNode, 297
- quickSpinEnumerationNode, 297
- quickSpinFloatNode, 297
- quickSpinIntegerNode, 297
- quickSpinRegisterNode, 297
- quickSpinStringNode, 297
- quickSpinEnumerationNode
 - QuickSpinDefsC.h, 297
- quickSpinFloatNode
 - QuickSpinDefsC.h, 297
- quickSpinInit
 - QuickSpinC.h, 294
- quickSpinInitEx
 - QuickSpinC.h, 294
- quickSpinIntegerNode
 - QuickSpinDefsC.h, 297
- quickSpinRegisterNode
 - QuickSpinDefsC.h, 297
- quickSpinStringNode
 - QuickSpinDefsC.h, 297
- quickSpinTLDevice, 133
 - DeviceAccessStatus, 133
 - DeviceBootloaderVersion, 134
 - DeviceCurrentSpeed, 134
 - DeviceDisplayName, 134
 - DeviceDriverVersion, 134
 - DeviceEndiannessMechanism, 134
 - DeviceID, 134
 - DeviceInstanceId, 134
 - DevicesUpdater, 134
 - DeviceLinkSpeed, 135
 - DeviceLocation, 135
 - DeviceModelName, 135
 - DeviceMulticastMonitorMode, 135
 - DevicePortId, 135
 - DeviceReset, 135
 - DeviceSerialNumber, 135
 - DeviceType, 135
 - DeviceU3VProtocol, 136
 - DeviceUserID, 136
 - DeviceVendorName, 136
 - DeviceVersion, 136
 - GenICamXMLLocation, 136
 - GenICamXMLPath, 136
 - GevCCP, 136
 - GevDeviceAutoForceIP, 136
 - GevDeviceDiscoverMaximumPacketSize, 137
 - GevDeviceForceGateway, 137
 - GevDeviceForceIP, 137
 - GevDeviceForceIPAddress, 137
 - GevDeviceForceSubnetMask, 137
 - GevDeviceGateway, 137
 - GevDeviceIPAddress, 137
 - GevDevicesWrongSubnet, 137
 - GevDeviceMACAddress, 138
 - GevDeviceMaximumPacketSize, 138
 - GevDeviceMaximumRetryCount, 138
 - GevDeviceModelsBigEndian, 138
 - GevDevicePort, 138
 - GevDeviceReadAndWriteTimeout, 138
 - GevDeviceSubnetMask, 138
 - GevVersionMajor, 138
 - GevVersionMinor, 139
 - GUIXMLLocation, 139
 - GUIXMLPath, 139
- quickSpinTLDeviceInit
 - QuickSpinC.h, 295
- quickSpinTLInterface, 139
 - ActionCommand, 140
 - DeviceAccessStatus, 140
 - DeviceCount, 140
 - DeviceID, 141
 - DeviceModelName, 141
 - DeviceSelector, 141
 - DeviceSerialNumber, 141
 - DeviceUnlock, 141
 - DeviceUpdateList, 141
 - DeviceVendorName, 141
 - FilterDriverStatus, 141
 - GevActionDeviceKey, 142
 - GevActionGroupKey, 142
 - GevActionGroupMask, 142
 - GevActionTime, 142
 - GevDeviceAutoForceIP, 142
 - GevDeviceForceGateway, 142
 - GevDeviceForceIP, 142
 - GevDeviceForceIPAddress, 142
 - GevDeviceForceSubnetMask, 143
 - GevDeviceGateway, 143
 - GevDeviceIPAddress, 143
 - GevDeviceMACAddress, 143
 - GevDeviceSubnetMask, 143
 - GevInterfaceGateway, 143
 - GevInterfaceGatewaySelector, 143
 - GevInterfaceMACAddress, 143
 - GevInterfaceMTU, 144
 - GevInterfaceReceiveLinkSpeed, 144
 - GevInterfaceSubnetIPAddress, 144
 - GevInterfaceSubnetMask, 144
 - GevInterfaceSubnetSelector, 144
 - GevInterfaceTransmitLinkSpeed, 144
 - HostAdapterDriverVersion, 144
 - HostAdapterName, 144
 - HostAdapterVendor, 145
 - IncompatibleDeviceCount, 145
 - IncompatibleDeviceID, 145
 - IncompatibleDeviceModelName, 145
 - IncompatibleDeviceSelector, 145
 - IncompatibleDeviceVendorName, 145
 - IncompatibleGevDeviceIPAddress, 145
 - IncompatibleGevDeviceMACAddress, 145
 - IncompatibleGevDeviceSubnetMask, 146
 - InterfaceDisplayName, 146
 - InterfaceID, 146
 - InterfaceType, 146

- POEStatus, 146
- quickSpinTLInterfaceInit
 - QuickSpinC.h, 295
- quickSpinTLStream, 147
 - StreamAnnounceBufferMinimum, 147
 - StreamAnnouncedBufferCount, 147
 - StreamBlockTransferSize, 147
 - StreamBufferAlignment, 148
 - StreamBufferCountManual, 148
 - StreamBufferCountMax, 148
 - StreamBufferCountMode, 148
 - StreamBufferCountResult, 148
 - StreamBufferHandlingMode, 148
 - StreamChunkCountMaximum, 148
 - StreamCRCCheckEnable, 148
 - StreamDeliveredFrameCount, 149
 - StreamDroppedFrameCount, 149
 - StreamID, 149
 - StreamIncompleteFrameCount, 149
 - StreamInputBufferCount, 149
 - StreamIsGrabbing, 149
 - StreamLostFrameCount, 149
 - StreamMissedPacketCount, 149
 - StreamMode, 150
 - StreamOutputBufferCount, 150
 - StreamPacketResendEnable, 150
 - StreamPacketResendMaxRequests, 150
 - StreamPacketResendReceivedPacketCount, 150
 - StreamPacketResendRequestCount, 150
 - StreamPacketResendRequestedPacketCount, 150
 - StreamPacketResendRequestSuccessCount, 150
 - StreamPacketResendTimeout, 151
 - StreamReceivedFrameCount, 151
 - StreamReceivedPacketCount, 151
 - StreamStartedFrameCount, 151
 - StreamType, 151
- quickSpinTLStreamInit
 - QuickSpinC.h, 295
- quickSpinTLSystem, 152
 - EnumerateGen2Cameras, 152
 - EnumerateGEVInterfaces, 152
 - EnumerateUSBInterfaces, 152
 - GenTLSFNCVersionMajor, 153
 - GenTLSFNCVersionMinor, 153
 - GenTLSFNCVersionSubMinor, 153
 - GenTLVersionMajor, 153
 - GenTLVersionMinor, 153
 - GevInterfaceDefaultGateway, 153
 - GevInterfaceDefaultIPAddress, 153
 - GevInterfaceDefaultSubnetMask, 153
 - GevInterfaceMACAddress, 154
 - GevVersionMajor, 154
 - GevVersionMinor, 154
 - InterfaceDisplayName, 154
 - InterfaceID, 154
 - InterfaceSelector, 154
 - InterfaceUpdateList, 154
 - TLDisplayName, 154
 - TLFileName, 155
 - TLID, 155
 - TLModelName, 155
 - TLPath, 155
 - TLType, 155
 - TLVendorName, 155
 - TLVersion, 155
- quickSpinTLSystemInit
 - QuickSpinC.h, 295
- RegionDestination
 - quickSpin, 116
- RegionDestination_Stream0
 - CameraDefsC.h, 271
- RegionDestination_Stream1
 - CameraDefsC.h, 271
- RegionDestination_Stream2
 - CameraDefsC.h, 271
- RegionMode
 - quickSpin, 116
- RegionMode_Off
 - CameraDefsC.h, 271
- RegionMode_On
 - CameraDefsC.h, 271
- RegionSelector
 - quickSpin, 116
- RegionSelector_All
 - CameraDefsC.h, 272
- RegionSelector_Region0
 - CameraDefsC.h, 272
- RegionSelector_Region1
 - CameraDefsC.h, 272
- RegionSelector_Region2
 - CameraDefsC.h, 272
- RegisterNode
 - SpinnakerGenApiDefsC.h, 468
- reserved
 - spinAVIOption, 156
 - spinBMPOption, 157
 - spinH264Option, 165
 - spinJPEGOption, 166
 - spinJPG2Option, 167
 - spinMJPEGOption, 169
 - spinPGMOption, 170
 - spinPNGOption, 171
 - spinPPMOption, 172
 - spinTIFFOption, 173
- ReverseX
 - quickSpin, 117
- ReverseY
 - quickSpin, 117
- RgbTransformLightSource
 - quickSpin, 117
- RgbTransformLightSource_Cloudy6500K
 - CameraDefsC.h, 272
- RgbTransformLightSource_CoolFluorescent4000K
 - CameraDefsC.h, 272
- RgbTransformLightSource_Custom
 - CameraDefsC.h, 272

- RgbTransformLightSource_Daylight5000K
 - CameraDefsC.h, [272](#)
- RgbTransformLightSource_General
 - CameraDefsC.h, [272](#)
- RgbTransformLightSource_Shade8000K
 - CameraDefsC.h, [272](#)
- RgbTransformLightSource_Tungsten2800K
 - CameraDefsC.h, [272](#)
- RgbTransformLightSource_WarmFluorescent3000K
 - CameraDefsC.h, [272](#)
- RO
 - SpinnakerGenApiDefsC.h, [464](#)
- RW
 - SpinnakerGenApiDefsC.h, [464](#)
- Saturation
 - quickSpin, [117](#)
- SaturationEnable
 - quickSpin, [117](#)
- Scan3dAxisMax
 - quickSpin, [117](#)
- Scan3dAxisMin
 - quickSpin, [117](#)
- Scan3dCoordinateOffset
 - quickSpin, [117](#)
- Scan3dCoordinateReferenceSelector
 - quickSpin, [118](#)
- Scan3dCoordinateReferenceSelector_RotationX
 - CameraDefsC.h, [273](#)
- Scan3dCoordinateReferenceSelector_RotationY
 - CameraDefsC.h, [273](#)
- Scan3dCoordinateReferenceSelector_RotationZ
 - CameraDefsC.h, [273](#)
- Scan3dCoordinateReferenceSelector_TranslationX
 - CameraDefsC.h, [273](#)
- Scan3dCoordinateReferenceSelector_TranslationY
 - CameraDefsC.h, [273](#)
- Scan3dCoordinateReferenceSelector_TranslationZ
 - CameraDefsC.h, [273](#)
- Scan3dCoordinateReferenceValue
 - quickSpin, [118](#)
- Scan3dCoordinateScale
 - quickSpin, [118](#)
- Scan3dCoordinateSelector
 - quickSpin, [118](#)
- Scan3dCoordinateSelector_CoordinateA
 - CameraDefsC.h, [273](#)
- Scan3dCoordinateSelector_CoordinateB
 - CameraDefsC.h, [273](#)
- Scan3dCoordinateSelector_CoordinateC
 - CameraDefsC.h, [273](#)
- Scan3dCoordinateSystem
 - quickSpin, [118](#)
- Scan3dCoordinateSystem_Cartesian
 - CameraDefsC.h, [273](#)
- Scan3dCoordinateSystem_Cylindrical
 - CameraDefsC.h, [273](#)
- Scan3dCoordinateSystem_Spherical
 - CameraDefsC.h, [273](#)
- Scan3dCoordinateSystemReference
 - quickSpin, [118](#)
- Scan3dCoordinateSystemReference_Anchor
 - CameraDefsC.h, [274](#)
- Scan3dCoordinateSystemReference_Transformed
 - CameraDefsC.h, [274](#)
- Scan3dCoordinateTransformSelector
 - quickSpin, [118](#)
- Scan3dCoordinateTransformSelector_RotationX
 - CameraDefsC.h, [274](#)
- Scan3dCoordinateTransformSelector_RotationY
 - CameraDefsC.h, [274](#)
- Scan3dCoordinateTransformSelector_RotationZ
 - CameraDefsC.h, [274](#)
- Scan3dCoordinateTransformSelector_TranslationX
 - CameraDefsC.h, [274](#)
- Scan3dCoordinateTransformSelector_TranslationY
 - CameraDefsC.h, [274](#)
- Scan3dCoordinateTransformSelector_TranslationZ
 - CameraDefsC.h, [274](#)
- Scan3dDistanceUnit
 - quickSpin, [118](#)
- Scan3dDistanceUnit_Inch
 - CameraDefsC.h, [274](#)
- Scan3dDistanceUnit_Millimeter
 - CameraDefsC.h, [274](#)
- Scan3dInvalidDataFlag
 - quickSpin, [119](#)
- Scan3dInvalidDataValue
 - quickSpin, [119](#)
- Scan3dOutputMode
 - quickSpin, [119](#)
- Scan3dOutputMode_CalibratedABC_Grid
 - CameraDefsC.h, [276](#)
- Scan3dOutputMode_CalibratedABC_PointCloud
 - CameraDefsC.h, [276](#)
- Scan3dOutputMode_CalibratedAC
 - CameraDefsC.h, [276](#)
- Scan3dOutputMode_CalibratedAC_Linescan
 - CameraDefsC.h, [276](#)
- Scan3dOutputMode_CalibratedC
 - CameraDefsC.h, [276](#)
- Scan3dOutputMode_CalibratedC_Linescan
 - CameraDefsC.h, [276](#)
- Scan3dOutputMode_DisparityC
 - CameraDefsC.h, [276](#)
- Scan3dOutputMode_DisparityC_Linescan
 - CameraDefsC.h, [276](#)
- Scan3dOutputMode_RectifiedC
 - CameraDefsC.h, [276](#)
- Scan3dOutputMode_RectifiedC_Linescan
 - CameraDefsC.h, [276](#)
- Scan3dOutputMode_UncalibratedC
 - CameraDefsC.h, [276](#)
- Scan3dTransformValue
 - quickSpin, [119](#)
- SensorDescription
 - quickSpin, [119](#)

- SensorDigitizationTaps
 - quickSpin, [119](#)
- SensorDigitizationTaps_Eight
 - CameraDefsC.h, [277](#)
- SensorDigitizationTaps_Four
 - CameraDefsC.h, [277](#)
- SensorDigitizationTaps_One
 - CameraDefsC.h, [276](#)
- SensorDigitizationTaps_Ten
 - CameraDefsC.h, [277](#)
- SensorDigitizationTaps_Three
 - CameraDefsC.h, [276](#)
- SensorDigitizationTaps_Two
 - CameraDefsC.h, [276](#)
- SensorHeight
 - quickSpin, [119](#)
- SensorShutterMode
 - quickSpin, [119](#)
- SensorShutterMode_Global
 - CameraDefsC.h, [277](#)
- SensorShutterMode_GlobalReset
 - CameraDefsC.h, [277](#)
- SensorShutterMode_Rolling
 - CameraDefsC.h, [277](#)
- SensorTaps
 - quickSpin, [120](#)
- SensorTaps_Eight
 - CameraDefsC.h, [277](#)
- SensorTaps_Four
 - CameraDefsC.h, [277](#)
- SensorTaps_One
 - CameraDefsC.h, [277](#)
- SensorTaps_Ten
 - CameraDefsC.h, [277](#)
- SensorTaps_Three
 - CameraDefsC.h, [277](#)
- SensorTaps_Two
 - CameraDefsC.h, [277](#)
- SensorWidth
 - quickSpin, [120](#)
- SequencerConfigurationMode
 - quickSpin, [120](#)
- SequencerConfigurationMode_Off
 - CameraDefsC.h, [278](#)
- SequencerConfigurationMode_On
 - CameraDefsC.h, [278](#)
- SequencerConfigurationValid
 - quickSpin, [120](#)
- SequencerConfigurationValid_No
 - CameraDefsC.h, [278](#)
- SequencerConfigurationValid_Yes
 - CameraDefsC.h, [278](#)
- SequencerFeatureEnable
 - quickSpin, [120](#)
- SequencerMode
 - quickSpin, [120](#)
- SequencerMode_Off
 - CameraDefsC.h, [278](#)
- SequencerMode_On
 - CameraDefsC.h, [278](#)
- SequencerPathSelector
 - quickSpin, [120](#)
- SequencerSetActive
 - quickSpin, [120](#)
- SequencerSetLoad
 - quickSpin, [121](#)
- SequencerSetNext
 - quickSpin, [121](#)
- SequencerSetSave
 - quickSpin, [121](#)
- SequencerSetSelector
 - quickSpin, [121](#)
- SequencerSetStart
 - quickSpin, [121](#)
- SequencerSetValid
 - quickSpin, [121](#)
- SequencerSetValid_No
 - CameraDefsC.h, [279](#)
- SequencerSetValid_Yes
 - CameraDefsC.h, [279](#)
- SequencerTriggerActivation
 - quickSpin, [121](#)
- SequencerTriggerActivation_AnyEdge
 - CameraDefsC.h, [279](#)
- SequencerTriggerActivation_FallingEdge
 - CameraDefsC.h, [279](#)
- SequencerTriggerActivation_LevelHigh
 - CameraDefsC.h, [279](#)
- SequencerTriggerActivation_LevelLow
 - CameraDefsC.h, [279](#)
- SequencerTriggerActivation_RisingEdge
 - CameraDefsC.h, [279](#)
- SequencerTriggerSource
 - quickSpin, [121](#)
- SequencerTriggerSource_FrameStart
 - CameraDefsC.h, [279](#)
- SequencerTriggerSource_Off
 - CameraDefsC.h, [279](#)
- SerialPortBaudRate
 - quickSpin, [122](#)
- SerialPortBaudRate_Baud115200
 - CameraDefsC.h, [280](#)
- SerialPortBaudRate_Baud1200
 - CameraDefsC.h, [280](#)
- SerialPortBaudRate_Baud14400
 - CameraDefsC.h, [280](#)
- SerialPortBaudRate_Baud19200
 - CameraDefsC.h, [280](#)
- SerialPortBaudRate_Baud230400
 - CameraDefsC.h, [280](#)
- SerialPortBaudRate_Baud2400
 - CameraDefsC.h, [280](#)
- SerialPortBaudRate_Baud300
 - CameraDefsC.h, [280](#)
- SerialPortBaudRate_Baud38400
 - CameraDefsC.h, [280](#)

- SerialPortBaudRate_Baud460800
 - CameraDefsC.h, [280](#)
- SerialPortBaudRate_Baud4800
 - CameraDefsC.h, [280](#)
- SerialPortBaudRate_Baud57600
 - CameraDefsC.h, [280](#)
- SerialPortBaudRate_Baud600
 - CameraDefsC.h, [280](#)
- SerialPortBaudRate_Baud921600
 - CameraDefsC.h, [280](#)
- SerialPortBaudRate_Baud9600
 - CameraDefsC.h, [280](#)
- SerialPortDataBits
 - quickSpin, [122](#)
- SerialPortParity
 - quickSpin, [122](#)
- SerialPortParity_Even
 - CameraDefsC.h, [280](#)
- SerialPortParity_Mark
 - CameraDefsC.h, [280](#)
- SerialPortParity_None
 - CameraDefsC.h, [280](#)
- SerialPortParity_Odd
 - CameraDefsC.h, [280](#)
- SerialPortParity_Space
 - CameraDefsC.h, [280](#)
- SerialPortSelector
 - quickSpin, [122](#)
- SerialPortSelector_SerialPort0
 - CameraDefsC.h, [280](#)
- SerialPortSource
 - quickSpin, [122](#)
- SerialPortSource_Line0
 - CameraDefsC.h, [281](#)
- SerialPortSource_Line1
 - CameraDefsC.h, [281](#)
- SerialPortSource_Line2
 - CameraDefsC.h, [281](#)
- SerialPortSource_Line3
 - CameraDefsC.h, [281](#)
- SerialPortSource_Off
 - CameraDefsC.h, [281](#)
- SerialPortStopBits
 - quickSpin, [122](#)
- SerialPortStopBits_Bits1
 - CameraDefsC.h, [281](#)
- SerialPortStopBits_Bits1AndAHalf
 - CameraDefsC.h, [281](#)
- SerialPortStopBits_Bits2
 - CameraDefsC.h, [281](#)
- SerialReceiveFramingErrorCount
 - quickSpin, [122](#)
- SerialReceiveParityErrorCount
 - quickSpin, [122](#)
- SerialReceiveQueueClear
 - quickSpin, [123](#)
- SerialReceiveQueueCurrentCharacterCount
 - quickSpin, [123](#)
- SerialReceiveQueueMaxCharacterCount
 - quickSpin, [123](#)
- SerialTransmitQueueCurrentCharacterCount
 - quickSpin, [123](#)
- SerialTransmitQueueMaxCharacterCount
 - quickSpin, [123](#)
- Sharpening
 - quickSpin, [123](#)
- SharpeningAuto
 - quickSpin, [123](#)
- SharpeningEnable
 - quickSpin, [123](#)
- SharpeningThreshold
 - quickSpin, [124](#)
- Signed
 - SpinnakerGenApiDefsC.h, [468](#)
- SoftwareSignalPulse
 - quickSpin, [124](#)
- SoftwareSignalSelector
 - quickSpin, [124](#)
- SoftwareSignalSelector_SoftwareSignal0
 - CameraDefsC.h, [281](#)
- SoftwareSignalSelector_SoftwareSignal1
 - CameraDefsC.h, [281](#)
- SoftwareSignalSelector_SoftwareSignal2
 - CameraDefsC.h, [281](#)
- SourceCount
 - quickSpin, [124](#)
- SourceSelector
 - quickSpin, [124](#)
- SourceSelector_All
 - CameraDefsC.h, [282](#)
- SourceSelector_Source0
 - CameraDefsC.h, [282](#)
- SourceSelector_Source1
 - CameraDefsC.h, [282](#)
- SourceSelector_Source2
 - CameraDefsC.h, [282](#)
- spinAccessMode
 - SpinnakerGenApiDefsC.h, [463](#)
- spinAcquisitionModeEnums
 - CameraDefsC.h, [208](#)
- spinAcquisitionStatusSelectorEnums
 - CameraDefsC.h, [208](#)
- spinActionCommandStatus
 - SpinnakerDefsC.h, [412](#)
- spinActionUnconditionalModeEnums
 - CameraDefsC.h, [208](#)
- spinAdcBitDepthEnums
 - CameraDefsC.h, [209](#)
- spinArrivalEventFunction
 - SpinnakerDefsC.h, [408](#)
- spinAutoAlgorithmSelectorEnums
 - CameraDefsC.h, [209](#)
- spinAutoExposureControlPriorityEnums
 - CameraDefsC.h, [209](#)
- spinAutoExposureLightingModeEnums
 - CameraDefsC.h, [210](#)

- spinAutoExposureMeteringModeEnums
 - CameraDefsC.h, [210](#)
- spinAutoExposureTargetGreyValueAutoEnums
 - CameraDefsC.h, [211](#)
- spinAVIOption, [156](#)
 - frameRate, [156](#)
 - height, [156](#)
 - reserved, [156](#)
 - width, [157](#)
- spinBalanceRatioSelectorEnums
 - CameraDefsC.h, [211](#)
- spinBalanceWhiteAutoEnums
 - CameraDefsC.h, [211](#)
- spinBalanceWhiteAutoProfileEnums
 - CameraDefsC.h, [212](#)
- spinBinningHorizontalModeEnums
 - CameraDefsC.h, [212](#)
- spinBinningSelectorEnums
 - CameraDefsC.h, [212](#)
- spinBinningVerticalModeEnums
 - CameraDefsC.h, [213](#)
- spinBlackLevelAutoBalanceEnums
 - CameraDefsC.h, [213](#)
- spinBlackLevelAutoEnums
 - CameraDefsC.h, [213](#)
- spinBlackLevelSelectorEnums
 - CameraDefsC.h, [214](#)
- spinBMPOption, [157](#)
 - indexedColor_8bit, [157](#)
 - reserved, [157](#)
- spinBooleanGetValue
 - SpinnakerGenApiC.h, [423](#)
- spinBooleanSetValue
 - SpinnakerGenApiC.h, [423](#)
- spinCachingMode
 - SpinnakerGenApiDefsC.h, [464](#)
- spinCamera
 - SpinnakerDefsC.h, [408](#)
- spinCameraBeginAcquisition
 - SpinnakerC.h, [307](#)
- spinCameraDelInit
 - SpinnakerC.h, [308](#)
- spinCameraDiscoverMaxPacketSize
 - SpinnakerC.h, [308](#)
- spinCameraEndAcquisition
 - SpinnakerC.h, [309](#)
- spinCameraForceIP
 - SpinnakerC.h, [309](#)
- spinCameraGetAccessMode
 - SpinnakerC.h, [309](#)
- spinCameraGetGuiXml
 - SpinnakerC.h, [310](#)
- spinCameraGetNextImage
 - SpinnakerC.h, [310](#)
- spinCameraGetNextImageEx
 - SpinnakerC.h, [311](#)
- spinCameraGetNextImageSync
 - SpinnakerC.h, [311](#)
- spinCameraGetNodeMap
 - SpinnakerC.h, [312](#)
- spinCameraGetTLDeviceNodeMap
 - SpinnakerC.h, [313](#)
- spinCameraGetTLStreamNodeMap
 - SpinnakerC.h, [313](#)
- spinCameraGetUniqueID
 - SpinnakerC.h, [313](#)
- spinCameraInit
 - SpinnakerC.h, [314](#)
- spinCamerasInitialized
 - SpinnakerC.h, [314](#)
- spinCamerasStreaming
 - SpinnakerC.h, [315](#)
- spinCamerasValid
 - SpinnakerC.h, [315](#)
- spinCameraList
 - SpinnakerDefsC.h, [408](#)
- spinCameraListAppend
 - SpinnakerC.h, [316](#)
- spinCameraListClear
 - SpinnakerC.h, [316](#)
- spinCameraListCreateEmpty
 - SpinnakerC.h, [317](#)
- spinCameraListDestroy
 - SpinnakerC.h, [317](#)
- spinCameraListGet
 - SpinnakerC.h, [318](#)
- spinCameraListGetBySerial
 - SpinnakerC.h, [318](#)
- spinCameraListGetSize
 - SpinnakerC.h, [319](#)
- spinCameraListRemove
 - SpinnakerC.h, [319](#)
- spinCameraListRemoveBySerial
 - SpinnakerC.h, [320](#)
- spinCameraReadPort
 - SpinnakerC.h, [320](#)
- spinCameraRegisterDeviceEventHandler
 - SpinnakerC.h, [320](#)
- spinCameraRegisterDeviceEventHandlerEx
 - SpinnakerC.h, [321](#)
- spinCameraRegisterImageEventHandler
 - SpinnakerC.h, [321](#)
- spinCameraRegisterImageEventHandlerEx
 - SpinnakerC.h, [322](#)
- spinCameraRegisterImageListEventHandler
 - SpinnakerC.h, [322](#)
- spinCameraRelease
 - SpinnakerC.h, [323](#)
- spinCameraUnregisterDeviceEventHandler
 - SpinnakerC.h, [323](#)
- spinCameraUnregisterImageEventHandler
 - SpinnakerC.h, [324](#)
- spinCameraUnregisterImageListEventHandler
 - SpinnakerC.h, [324](#)
- spinCameraWritePort
 - SpinnakerC.h, [325](#)

- spinCategoryGetFeatureByIndex
 - SpinnakerGenApiC.h, [424](#)
- spinCategoryGetNumFeatures
 - SpinnakerGenApiC.h, [424](#)
- spinCategoryReleaseNode
 - SpinnakerGenApiC.h, [425](#)
- spinChunkBlackLevelSelectorEnums
 - CameraDefsC.h, [214](#)
- spinChunkCounterSelectorEnums
 - CameraDefsC.h, [214](#)
- spinChunkData, [158](#)
 - m_blackLevel, [159](#)
 - m_compressionMode, [159](#)
 - m_compressionRatio, [159](#)
 - m_counterValue, [159](#)
 - m_cRC, [159](#)
 - m_encoderValue, [159](#)
 - m_exposureEndLineStatusAll, [159](#)
 - m_exposureTime, [160](#)
 - m_frameID, [160](#)
 - m_gain, [160](#)
 - m_height, [160](#)
 - m_image, [160](#)
 - m_inferenceConfidence, [160](#)
 - m_inferenceFrameId, [160](#)
 - m_inferenceResult, [160](#)
 - m_linePitch, [161](#)
 - m_lineStatusAll, [161](#)
 - m_offsetX, [161](#)
 - m_offsetY, [161](#)
 - m_partSelector, [161](#)
 - m_pixelDynamicRangeMax, [161](#)
 - m_pixelDynamicRangeMin, [161](#)
 - m_scan3dAxisMax, [161](#)
 - m_scan3dAxisMin, [162](#)
 - m_scan3dCoordinateOffset, [162](#)
 - m_scan3dCoordinateReferenceValue, [162](#)
 - m_scan3dCoordinateScale, [162](#)
 - m_scan3dInvalidDataValue, [162](#)
 - m_scan3dTransformValue, [162](#)
 - m_scanLineSelector, [162](#)
 - m_sequencerSetActive, [162](#)
 - m_serialDataLength, [163](#)
 - m_streamChannelID, [163](#)
 - m_timerValue, [163](#)
 - m_timestamp, [163](#)
 - m_timestampLatchValue, [163](#)
 - m_transferBlockID, [163](#)
 - m_transferQueueCurrentBlockCount, [163](#)
 - m_width, [163](#)
- spinChunkEncoderSelectorEnums
 - CameraDefsC.h, [215](#)
- spinChunkEncoderStatusEnums
 - CameraDefsC.h, [215](#)
- spinChunkExposureTimeSelectorEnums
 - CameraDefsC.h, [215](#)
- spinChunkGainSelectorEnums
 - CameraDefsC.h, [216](#)
- spinChunkImageComponentEnums
 - CameraDefsC.h, [216](#)
- spinChunkPixelFormatEnums
 - CameraDefsC.h, [217](#)
- spinChunkRegionIDEnums
 - CameraDefsC.h, [217](#)
- spinChunkScan3dCoordinateReferenceSelectorEnums
 - CameraDefsC.h, [217](#)
- spinChunkScan3dCoordinateSelectorEnums
 - CameraDefsC.h, [218](#)
- spinChunkScan3dCoordinateSystemEnums
 - CameraDefsC.h, [218](#)
- spinChunkScan3dCoordinateSystemReferenceEnums
 - CameraDefsC.h, [218](#)
- spinChunkScan3dCoordinateTransformSelectorEnums
 - CameraDefsC.h, [219](#)
- spinChunkScan3dDistanceUnitEnums
 - CameraDefsC.h, [219](#)
- spinChunkScan3dOutputModeEnums
 - CameraDefsC.h, [219](#)
- spinChunkSelectorEnums
 - CameraDefsC.h, [220](#)
- spinChunkSourceIDEnums
 - CameraDefsC.h, [221](#)
- spinChunkTimerSelectorEnums
 - CameraDefsC.h, [221](#)
- spinChunkTransferStreamIDEnums
 - CameraDefsC.h, [222](#)
- spinCIConfigurationEnums
 - CameraDefsC.h, [222](#)
- spinCITimeSlotsCountEnums
 - CameraDefsC.h, [222](#)
- spinColorProcessingAlgorithm
 - SpinnakerDefsC.h, [413](#)
- spinColorTransformationSelectorEnums
 - CameraDefsC.h, [223](#)
- spinColorTransformationValueSelectorEnums
 - CameraDefsC.h, [223](#)
- spinCommandExecute
 - SpinnakerGenApiC.h, [425](#)
- spinCommandIsDone
 - SpinnakerGenApiC.h, [426](#)
- spinCompressionSaturationPriorityEnums
 - CameraDefsC.h, [224](#)
- spinCounterEventActivationEnums
 - CameraDefsC.h, [224](#)
- spinCounterEventSourceEnums
 - CameraDefsC.h, [224](#)
- spinCounterResetActivationEnums
 - CameraDefsC.h, [225](#)
- spinCounterResetSourceEnums
 - CameraDefsC.h, [225](#)
- spinCounterSelectorEnums
 - CameraDefsC.h, [226](#)
- spinCounterStatusEnums
 - CameraDefsC.h, [226](#)
- spinCounterTriggerActivationEnums
 - CameraDefsC.h, [227](#)

- spinCounterTriggerSourceEnums
 - CameraDefsC.h, [227](#)
- spinCxpConnectionTestModeEnums
 - CameraDefsC.h, [228](#)
- spinCxpLinkConfigurationEnums
 - CameraDefsC.h, [228](#)
- spinCxpLinkConfigurationPreferredEnums
 - CameraDefsC.h, [229](#)
- spinCxpLinkConfigurationStatusEnums
 - CameraDefsC.h, [230](#)
- spinCxpPoCxpStatusEnums
 - CameraDefsC.h, [231](#)
- spinDecimationHorizontalModeEnums
 - CameraDefsC.h, [231](#)
- spinDecimationSelectorEnums
 - CameraDefsC.h, [231](#)
- spinDecimationVerticalModeEnums
 - CameraDefsC.h, [232](#)
- spinDefectCorrectionModeEnums
 - CameraDefsC.h, [232](#)
- spinDeinterlacingEnums
 - CameraDefsC.h, [232](#)
- spinDeviceArrivalEventHandler
 - SpinnakerDefsC.h, [409](#)
- spinDeviceArrivalEventHandlerCreate
 - SpinnakerC.h, [325](#)
- spinDeviceArrivalEventHandlerDestroy
 - SpinnakerC.h, [325](#)
- spinDeviceCharacterSetEnums
 - CameraDefsC.h, [233](#)
- spinDeviceClockSelectorEnums
 - CameraDefsC.h, [233](#)
- spinDeviceConnectionStatusEnums
 - CameraDefsC.h, [233](#)
- spinDeviceEventData
 - SpinnakerDefsC.h, [409](#)
- spinDeviceEventFunction
 - SpinnakerDefsC.h, [409](#)
- spinDeviceEventGetId
 - SpinnakerC.h, [326](#)
- spinDeviceEventGetName
 - SpinnakerC.h, [326](#)
- spinDeviceEventGetPayloadData
 - SpinnakerC.h, [327](#)
- spinDeviceEventGetPayloadDataSize
 - SpinnakerC.h, [327](#)
- spinDeviceEventHandler
 - SpinnakerDefsC.h, [409](#)
- spinDeviceEventHandlerCreate
 - SpinnakerC.h, [328](#)
- spinDeviceEventHandlerDestroy
 - SpinnakerC.h, [328](#)
- spinDeviceIndicatorModeEnums
 - CameraDefsC.h, [234](#)
- spinDeviceLinkHeartbeatModeEnums
 - CameraDefsC.h, [234](#)
- spinDeviceLinkThroughputLimitModeEnums
 - CameraDefsC.h, [234](#)
- spinDevicePowerSupplySelectorEnums
 - CameraDefsC.h, [234](#)
- spinDeviceRegistersEndiannessEnums
 - CameraDefsC.h, [235](#)
- spinDeviceRemovalEventHandler
 - SpinnakerDefsC.h, [409](#)
- spinDeviceRemovalEventHandlerCreate
 - SpinnakerC.h, [329](#)
- spinDeviceRemovalEventHandlerDestroy
 - SpinnakerC.h, [329](#)
- spinDeviceScanTypeEnums
 - CameraDefsC.h, [235](#)
- spinDeviceSerialPortBaudRateEnums
 - CameraDefsC.h, [235](#)
- spinDeviceSerialPortSelectorEnums
 - CameraDefsC.h, [236](#)
- spinDeviceStreamChannelEndiannessEnums
 - CameraDefsC.h, [236](#)
- spinDeviceStreamChannelTypeEnums
 - CameraDefsC.h, [236](#)
- spinDeviceTapGeometryEnums
 - CameraDefsC.h, [237](#)
- spinDeviceTemperatureSelectorEnums
 - CameraDefsC.h, [238](#)
- spinDeviceTLTypeEnums
 - CameraDefsC.h, [238](#)
- spinDeviceTypeEnums
 - CameraDefsC.h, [239](#)
- spinDisplayNotation
 - SpinnakerGenApiDefsC.h, [464](#)
- spinEncoderModeEnums
 - CameraDefsC.h, [239](#)
- spinEncoderOutputModeEnums
 - CameraDefsC.h, [239](#)
- spinEncoderResetActivationEnums
 - CameraDefsC.h, [240](#)
- spinEncoderResetSourceEnums
 - CameraDefsC.h, [240](#)
- spinEncoderSelectorEnums
 - CameraDefsC.h, [241](#)
- spinEncoderSourceAEnums
 - CameraDefsC.h, [242](#)
- spinEncoderSourceBEnums
 - CameraDefsC.h, [242](#)
- spinEncoderStatusEnums
 - CameraDefsC.h, [242](#)
- spinEndianess
 - SpinnakerGenApiDefsC.h, [464](#)
- spinEnumerationEntryGetEnumValue
 - SpinnakerGenApiC.h, [426](#)
- spinEnumerationEntryGetIntValue
 - SpinnakerGenApiC.h, [427](#)
- spinEnumerationEntryGetSymbolic
 - SpinnakerGenApiC.h, [427](#)
- spinEnumerationGetCurrentEntry
 - SpinnakerGenApiC.h, [428](#)
- spinEnumerationGetEntryByIndex
 - SpinnakerGenApiC.h, [428](#)

- spinEnumerationGetEntryByName
 - SpinnakerGenApiC.h, [429](#)
- spinEnumerationGetNumEntries
 - SpinnakerGenApiC.h, [429](#)
- spinEnumerationReleaseNode
 - SpinnakerGenApiC.h, [430](#)
- spinEnumerationSetEnumValue
 - SpinnakerGenApiC.h, [430](#)
- spinEnumerationSetIntValue
 - SpinnakerGenApiC.h, [431](#)
- spinError
 - SpinnakerDefsC.h, [413](#)
- spinErrorGetLast
 - SpinnakerC.h, [330](#)
- spinErrorGetLastBuildDate
 - SpinnakerC.h, [330](#)
- spinErrorGetLastBuildTime
 - SpinnakerC.h, [331](#)
- spinErrorGetLastFileName
 - SpinnakerC.h, [331](#)
- spinErrorGetLastFullMessage
 - SpinnakerC.h, [332](#)
- spinErrorGetLastFunctionName
 - SpinnakerC.h, [332](#)
- spinErrorGetLastLineNumber
 - SpinnakerC.h, [333](#)
- spinErrorGetLastMessage
 - SpinnakerC.h, [333](#)
- spinEventNotificationEnums
 - CameraDefsC.h, [243](#)
- spinEventSelectorEnums
 - CameraDefsC.h, [243](#)
- spinExposureActiveModeEnums
 - CameraDefsC.h, [243](#)
- spinExposureAutoEnums
 - CameraDefsC.h, [243](#)
- spinExposureModeEnums
 - CameraDefsC.h, [244](#)
- spinExposureTimeModeEnums
 - CameraDefsC.h, [244](#)
- spinExposureTimeSelectorEnums
 - CameraDefsC.h, [245](#)
- spinFileOpenModeEnums
 - CameraDefsC.h, [245](#)
- spinFileOperationSelectorEnums
 - CameraDefsC.h, [245](#)
- spinFileOperationStatusEnums
 - CameraDefsC.h, [246](#)
- spinFileSelectorEnums
 - CameraDefsC.h, [246](#)
- spinFloatGetMax
 - SpinnakerGenApiC.h, [431](#)
- spinFloatGetMin
 - SpinnakerGenApiC.h, [432](#)
- spinFloatGetRepresentation
 - SpinnakerGenApiC.h, [432](#)
- spinFloatGetUnit
 - SpinnakerGenApiC.h, [433](#)
- spinFloatGetValue
 - SpinnakerGenApiC.h, [433](#)
- spinFloatGetValueEx
 - SpinnakerGenApiC.h, [434](#)
- spinFloatSetValue
 - SpinnakerGenApiC.h, [434](#)
- spinFloatSetValueEx
 - SpinnakerGenApiC.h, [435](#)
- spinGainAutoBalanceEnums
 - CameraDefsC.h, [246](#)
- spinGainAutoEnums
 - CameraDefsC.h, [248](#)
- spinGainSelectorEnums
 - CameraDefsC.h, [248](#)
- spinGevCCPEnums
 - CameraDefsC.h, [248](#)
- spinGevCurrentPhysicalLinkConfigurationEnums
 - CameraDefsC.h, [249](#)
- spinGevGVCPExtendedStatusCodesSelectorEnums
 - CameraDefsC.h, [249](#)
- spinGevGVSPExtendedIDModeEnums
 - CameraDefsC.h, [249](#)
- spinGevIEEE1588ClockAccuracyEnums
 - CameraDefsC.h, [250](#)
- spinGevIEEE1588ModeEnums
 - CameraDefsC.h, [250](#)
- spinGevIEEE1588StatusEnums
 - CameraDefsC.h, [250](#)
- spinGevIPConfigurationStatusEnums
 - CameraDefsC.h, [251](#)
- spinGevPhysicalLinkConfigurationEnums
 - CameraDefsC.h, [251](#)
- spinGevSupportedOptionSelectorEnums
 - CameraDefsC.h, [251](#)
- spinH264Option, [164](#)
 - bitrate, [164](#)
 - frameRate, [164](#)
 - height, [165](#)
 - reserved, [165](#)
 - width, [165](#)
- spinImage
 - SpinnakerDefsC.h, [409](#)
- spinImageCalculateStatistics
 - SpinnakerC.h, [334](#)
- spinImageCheckCRC
 - SpinnakerC.h, [334](#)
- spinImageChunkDataGetFloatValue
 - SpinnakerC.h, [335](#)
- spinImageChunkDataGetIntValue
 - SpinnakerC.h, [335](#)
- spinImageComponentSelectorEnums
 - CameraDefsC.h, [252](#)
- spinImageCompressionJPEGFormatOptionEnums
 - CameraDefsC.h, [253](#)
- spinImageCompressionModeEnums
 - CameraDefsC.h, [253](#)
- spinImageCompressionRateOptionEnums
 - CameraDefsC.h, [254](#)

- spinImageCreate
 - SpinnakerC.h, [335](#)
- spinImageCreateEmpty
 - SpinnakerC.h, [336](#)
- spinImageCreateEx
 - SpinnakerC.h, [336](#)
- spinImageCreateEx2
 - SpinnakerC.h, [337](#)
- spinImageDeepCopy
 - SpinnakerC.h, [337](#)
- spinImageDestroy
 - SpinnakerC.h, [338](#)
- spinImageEventFunction
 - SpinnakerDefsC.h, [410](#)
- spinImageEventHandler
 - SpinnakerDefsC.h, [410](#)
- spinImageEventHandlerCreate
 - SpinnakerC.h, [338](#)
- spinImageEventHandlerDestroy
 - SpinnakerC.h, [339](#)
- spinImageFileFormat
 - SpinnakerDefsC.h, [415](#)
- spinImageGetBitsPerPixel
 - SpinnakerC.h, [339](#)
- spinImageGetBufferSize
 - SpinnakerC.h, [340](#)
- spinImageGetChunkLayoutID
 - SpinnakerC.h, [340](#)
- spinImageGetColorProcessing
 - SpinnakerC.h, [341](#)
- spinImageGetData
 - SpinnakerC.h, [341](#)
- spinImageGetFrameID
 - SpinnakerC.h, [342](#)
- spinImageGetHeight
 - SpinnakerC.h, [342](#)
- spinImageGetID
 - SpinnakerC.h, [343](#)
- spinImageGetOffsetX
 - SpinnakerC.h, [343](#)
- spinImageGetOffsetY
 - SpinnakerC.h, [344](#)
- spinImageGetPaddingX
 - SpinnakerC.h, [344](#)
- spinImageGetPaddingY
 - SpinnakerC.h, [345](#)
- spinImageGetPayloadType
 - SpinnakerC.h, [345](#)
- spinImageGetPixelFormat
 - SpinnakerC.h, [346](#)
- spinImageGetPixelFormatName
 - SpinnakerC.h, [346](#)
- spinImageGetPrivateData
 - SpinnakerC.h, [347](#)
- spinImageGetSize
 - SpinnakerC.h, [347](#)
- spinImageGetStatus
 - SpinnakerC.h, [348](#)
- spinImageGetStatusDescription
 - SpinnakerC.h, [348](#)
- spinImageGetStride
 - SpinnakerC.h, [349](#)
- spinImageGetTimeStamp
 - SpinnakerC.h, [349](#)
- spinImageGetTLPayloadType
 - SpinnakerC.h, [350](#)
- spinImageGetTLPixelFormat
 - SpinnakerC.h, [350](#)
- spinImageGetTLPixelFormatNamespace
 - SpinnakerC.h, [351](#)
- spinImageGetValidPayloadSize
 - SpinnakerC.h, [351](#)
- spinImageGetWidth
 - SpinnakerC.h, [352](#)
- spinImageHasCRC
 - SpinnakerC.h, [352](#)
- spinImageIsIncomplete
 - SpinnakerC.h, [353](#)
- spinImageList
 - SpinnakerDefsC.h, [410](#)
- spinImageListAppend
 - SpinnakerC.h, [353](#)
- spinImageListClear
 - SpinnakerC.h, [354](#)
- spinImageListCreateEmpty
 - SpinnakerC.h, [354](#)
- spinImageListDestroy
 - SpinnakerC.h, [355](#)
- spinImageListEventFunction
 - SpinnakerDefsC.h, [410](#)
- spinImageListEventHandler
 - SpinnakerDefsC.h, [410](#)
- spinImageListEventHandlerCreate
 - SpinnakerC.h, [355](#)
- spinImageListEventHandlerDestroy
 - SpinnakerC.h, [356](#)
- spinImageListGet
 - SpinnakerC.h, [356](#)
- spinImageListGetByPixelFormat
 - SpinnakerC.h, [357](#)
- spinImageListGetSize
 - SpinnakerC.h, [357](#)
- spinImageListLoad
 - SpinnakerC.h, [358](#)
- spinImageListRelease
 - SpinnakerC.h, [358](#)
- spinImageListRemove
 - SpinnakerC.h, [358](#)
- spinImageListRemoveByPixelFormat
 - SpinnakerC.h, [359](#)
- spinImageListSave
 - SpinnakerC.h, [359](#)
- spinImageProcessor
 - SpinnakerDefsC.h, [410](#)
- spinImageProcessorApplyGamma
 - SpinnakerC.h, [360](#)

- spinImageProcessorConvert
 - SpinnakerC.h, [360](#)
- spinImageProcessorConvertImageList
 - SpinnakerC.h, [361](#)
- spinImageProcessorCreate
 - SpinnakerC.h, [362](#)
- spinImageProcessorDestroy
 - SpinnakerC.h, [362](#)
- spinImageProcessorGetColorProcessing
 - SpinnakerC.h, [363](#)
- spinImageProcessorGetNumDecompressionThreads
 - SpinnakerC.h, [363](#)
- spinImageProcessorSetColorProcessing
 - SpinnakerC.h, [363](#)
- spinImageProcessorSetNumDecompressionThreads
 - SpinnakerC.h, [364](#)
- spinImageRelease
 - SpinnakerC.h, [364](#)
- spinImageReset
 - SpinnakerC.h, [365](#)
- spinImageResetEx
 - SpinnakerC.h, [365](#)
- spinImageSave
 - SpinnakerC.h, [366](#)
- spinImageSaveBmp
 - SpinnakerC.h, [367](#)
- spinImageSaveFromExt
 - SpinnakerC.h, [367](#)
- spinImageSaveJpeg
 - SpinnakerC.h, [367](#)
- spinImageSaveJpg2
 - SpinnakerC.h, [368](#)
- spinImageSavePgm
 - SpinnakerC.h, [368](#)
- spinImageSavePng
 - SpinnakerC.h, [369](#)
- spinImageSavePpm
 - SpinnakerC.h, [369](#)
- spinImageSaveTiff
 - SpinnakerC.h, [370](#)
- spinImageStatistics
 - SpinnakerDefsC.h, [411](#)
- spinImageStatisticsCreate
 - SpinnakerC.h, [370](#)
- spinImageStatisticsDestroy
 - SpinnakerC.h, [371](#)
- spinImageStatisticsDisableAll
 - SpinnakerC.h, [371](#)
- spinImageStatisticsEnableAll
 - SpinnakerC.h, [372](#)
- spinImageStatisticsEnableGreyOnly
 - SpinnakerC.h, [372](#)
- spinImageStatisticsEnableHslOnly
 - SpinnakerC.h, [373](#)
- spinImageStatisticsEnableRgbOnly
 - SpinnakerC.h, [373](#)
- spinImageStatisticsGetAll
 - SpinnakerC.h, [373](#)
- spinImageStatisticsGetChannelStatus
 - SpinnakerC.h, [374](#)
- spinImageStatisticsGetHistogram
 - SpinnakerC.h, [375](#)
- spinImageStatisticsGetMean
 - SpinnakerC.h, [375](#)
- spinImageStatisticsGetNumPixelValues
 - SpinnakerC.h, [376](#)
- spinImageStatisticsGetPixelValueRange
 - SpinnakerC.h, [376](#)
- spinImageStatisticsGetRange
 - SpinnakerC.h, [377](#)
- spinImageStatisticsSetChannelStatus
 - SpinnakerC.h, [377](#)
- spinImageStatus
 - SpinnakerDefsC.h, [415](#)
- spinIncMode
 - SpinnakerGenApiDefsC.h, [465](#)
- spinInputDirection
 - SpinnakerGenApiDefsC.h, [465](#)
- spinIntegerGetInc
 - SpinnakerGenApiC.h, [435](#)
- spinIntegerGetMax
 - SpinnakerGenApiC.h, [436](#)
- spinIntegerGetMin
 - SpinnakerGenApiC.h, [436](#)
- spinIntegerGetRepresentation
 - SpinnakerGenApiC.h, [437](#)
- spinIntegerGetValue
 - SpinnakerGenApiC.h, [437](#)
- spinIntegerGetValueEx
 - SpinnakerGenApiC.h, [438](#)
- spinIntegerSetValue
 - SpinnakerGenApiC.h, [438](#)
- spinIntegerSetValueEx
 - SpinnakerGenApiC.h, [439](#)
- spinInterface
 - SpinnakerDefsC.h, [411](#)
- spinInterfaceEventHandler
 - SpinnakerDefsC.h, [411](#)
- spinInterfaceEventHandlerCreate
 - SpinnakerC.h, [378](#)
- spinInterfaceEventHandlerDestroy
 - SpinnakerC.h, [378](#)
- spinInterfaceGetCameras
 - SpinnakerC.h, [379](#)
- spinInterfaceGetCamerasEx
 - SpinnakerC.h, [379](#)
- spinInterfaceGetTLNodeMap
 - SpinnakerC.h, [380](#)
- spinInterfaceIsInUse
 - SpinnakerC.h, [380](#)
- spinInterfaceList
 - SpinnakerDefsC.h, [411](#)
- spinInterfaceListClear
 - SpinnakerC.h, [381](#)
- spinInterfaceListCreateEmpty
 - SpinnakerC.h, [381](#)

- spinInterfaceListDestroy
 - SpinnakerC.h, [382](#)
- spinInterfaceListGet
 - SpinnakerC.h, [382](#)
- spinInterfaceListGetSize
 - SpinnakerC.h, [383](#)
- spinInterfaceRegisterDeviceArrivalEventHandler
 - SpinnakerC.h, [383](#)
- spinInterfaceRegisterDeviceRemovalEventHandler
 - SpinnakerC.h, [384](#)
- spinInterfaceRegisterInterfaceEventHandler
 - SpinnakerC.h, [384](#)
- spinInterfaceRelease
 - SpinnakerC.h, [385](#)
- spinInterfaceSendActionCommand
 - SpinnakerC.h, [385](#)
- spinInterfaceType
 - SpinnakerGenApiDefsC.h, [465](#)
- spinInterfaceUnregisterDeviceArrivalEventHandler
 - SpinnakerC.h, [386](#)
- spinInterfaceUnregisterDeviceRemovalEventHandler
 - SpinnakerC.h, [386](#)
- spinInterfaceUnregisterInterfaceEventHandler
 - SpinnakerC.h, [387](#)
- spinInterfaceUpdateCameras
 - SpinnakerC.h, [387](#)
- spinJPEGOption, [165](#)
 - progressive, [166](#)
 - quality, [166](#)
 - reserved, [166](#)
- spinJPG2Option, [166](#)
 - quality, [167](#)
 - reserved, [167](#)
- spinLibraryVersion, [167](#)
 - build, [168](#)
 - major, [168](#)
 - minor, [168](#)
 - type, [168](#)
- spinLineFormatEnums
 - CameraDefsC.h, [254](#)
- spinLineInputFilterSelectorEnums
 - CameraDefsC.h, [254](#)
- spinLineModeEnums
 - CameraDefsC.h, [255](#)
- spinLineSelectorEnums
 - CameraDefsC.h, [255](#)
- spinLineSourceEnums
 - CameraDefsC.h, [255](#)
- spinLinkType
 - SpinnakerGenApiDefsC.h, [466](#)
- spinLogDataGetCategoryName
 - SpinnakerC.h, [388](#)
- spinLogDataGetLogMessage
 - SpinnakerC.h, [388](#)
- spinLogDataGetNDC
 - SpinnakerC.h, [389](#)
- spinLogDataGetPriority
 - SpinnakerC.h, [389](#)
- spinLogDataGetPriorityName
 - SpinnakerC.h, [390](#)
- spinLogDataGetThreadName
 - SpinnakerC.h, [390](#)
- spinLogDataGetTimestamp
 - SpinnakerC.h, [391](#)
- spinLogEventData
 - SpinnakerDefsC.h, [411](#)
- spinLogEventFunction
 - SpinnakerDefsC.h, [411](#)
- spinLogEventHandler
 - SpinnakerDefsC.h, [412](#)
- spinLogEventHandlerCreate
 - SpinnakerC.h, [391](#)
- spinLogEventHandlerDestroy
 - SpinnakerC.h, [392](#)
- spinLogicBlockLUTInputActivationEnums
 - CameraDefsC.h, [256](#)
- spinLogicBlockLUTInputSelectorEnums
 - CameraDefsC.h, [256](#)
- spinLogicBlockLUTInputSourceEnums
 - CameraDefsC.h, [257](#)
- spinLogicBlockLUTSelectorEnums
 - CameraDefsC.h, [257](#)
- spinLogicBlockSelectorEnums
 - CameraDefsC.h, [258](#)
- spinLUTSelectorEnums
 - CameraDefsC.h, [258](#)
- spinMJPEGOption, [168](#)
 - frameRate, [169](#)
 - height, [169](#)
 - quality, [169](#)
 - reserved, [169](#)
 - width, [170](#)
- Spinnaker C API, [23](#)
- Spinnaker C Definitions, [21](#)
- Spinnaker C Enumerations, [29](#)
- Spinnaker C Function Signatures, [29](#)
- Spinnaker C GenICam API, [29](#)
- Spinnaker C GenICam Enumerations, [32](#)
- Spinnaker C GenICam Handles, [31](#)
- Spinnaker C Handles, [29](#)
- Spinnaker C QuickSpin API, [22](#)
- Spinnaker C Structures, [29](#)
- SPINNAKER_ACTION_COMMAND_STATUS_ACTION_LATE
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_ACTION_COMMAND_STATUS_ERROR
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_ACTION_COMMAND_STATUS_NO_REF_TIME
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_ACTION_COMMAND_STATUS_OK
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_ACTION_COMMAND_STATUS_OVERFLOW
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_BILINEAR
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_DIRECTIONAL_FILTER
 - SpinnakerDefsC.h, [413](#)

- SPINNAKER_COLOR_PROCESSING_ALGORITHM_EDGE_SENSING
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_HQ_SPINNAKER
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_IPP_SPINNAKER
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_NEAREST_NEIGHBOR
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_NEAREST_NEIGHBOR_INVALID
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_NONE_SPINNAKER
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_RIGID_SPINNAKER
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_WEIGHTED_FILTER_INDEX
 - SpinnakerDefsC.h, [413](#)
- SPINNAKER_ERR_ABORT
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_ACCESS_DENIED
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_BUFFER_TOO_SMALL
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_BUSY
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_CUSTOM_ID
 - SpinnakerDefsC.h, [415](#)
- SPINNAKER_ERR_ERROR
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_GENICAM_ACCESS
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_GENICAM_BAD_ALLOCATION
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_GENICAM_DYNAMIC_CAST
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_GENICAM_GENERIC
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_GENICAM_INVALID_ARGUMENT
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_GENICAM_LOGICAL
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_GENICAM_OUT_OF_RANGE
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_GENICAM_PROPERTY
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_GENICAM_RUN_TIME
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_GENICAM_TIMEOUT
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_IM_COLOR_CONVERSION
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_IM_CONVERT
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_IM_COPY
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_IM_HISTOGRAM_MEAN
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_IM_HISTOGRAM_RANGE
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_IM_MALLOC
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_IM_MIN_MAX
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_IM_NOT_SUPPORTED
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_INVALID_ADDRESS
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_INVALID_BUFFER
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_INVALID_HANDLE
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_INVALID_ID
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_INVALID_PARAMETER
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_INVALID_VALUE
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_IO
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_NO_DATA
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_NOT_AVAILABLE
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_NOT_IMPLEMENTED
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_NOT_INITIALIZED
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_OUT_OF_MEMORY
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_PARSING_CHUNK_DATA
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_RESOURCE_EXHAUSTED
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_RESOURCE_IN_USE
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_SUCCESS
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_ERR_TIMEOUT
 - SpinnakerDefsC.h, [414](#)
- SPINNAKER_IMAGE_FILE_FORMAT_BMP
 - SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_FORCE_32BITS
 - SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_FROM_FILE_EXT
 - SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_JPEG
 - SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_JPEG2000
 - SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_PGM
 - SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_PNG
 - SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_PPM
 - SpinnakerDefsC.h, [415](#)

- SPINNAKER_IMAGE_FILE_FORMAT_RAW
SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_TIFF
SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_STATUS_CHUNK_DATA_INVALID
SpinnakerDefsC.h, [416](#)
- SPINNAKER_IMAGE_STATUS_CRC_CHECK_FAILED
SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_STATUS_DATA_INCOMPLETE
SpinnakerDefsC.h, [416](#)
- SPINNAKER_IMAGE_STATUS_DATA_OVERFLOW
SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_STATUS_INFO_INCONSISTENT
SpinnakerDefsC.h, [416](#)
- SPINNAKER_IMAGE_STATUS_LEADER_BUFFER_SIZE_INCONSISTENT
SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_STATUS_MISSING_LEADER
SpinnakerDefsC.h, [416](#)
- SPINNAKER_IMAGE_STATUS_MISSING_PACKETS
SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_STATUS_MISSING_TRAILER
SpinnakerDefsC.h, [416](#)
- SPINNAKER_IMAGE_STATUS_NO_ERROR
SpinnakerDefsC.h, [415](#)
- SPINNAKER_IMAGE_STATUS_NO_SYSTEM_RESOURCES
SpinnakerDefsC.h, [416](#)
- SPINNAKER_IMAGE_STATUS_PACKETID_INCONSISTENT
SpinnakerDefsC.h, [416](#)
- SPINNAKER_IMAGE_STATUS_TRAILER_BUFFER_SIZE_INCONSISTENT
SpinnakerDefsC.h, [416](#)
- SPINNAKER_IMAGE_STATUS_UNKNOWN_ERROR
SpinnakerDefsC.h, [415](#)
- SPINNAKER_LOG_LEVEL_ALERT
SpinnakerDefsC.h, [416](#)
- SPINNAKER_LOG_LEVEL_CRIT
SpinnakerDefsC.h, [416](#)
- SPINNAKER_LOG_LEVEL_DEBUG
SpinnakerDefsC.h, [416](#)
- SPINNAKER_LOG_LEVEL_ERROR
SpinnakerDefsC.h, [416](#)
- SPINNAKER_LOG_LEVEL_FATAL
SpinnakerDefsC.h, [416](#)
- SPINNAKER_LOG_LEVEL_INFO
SpinnakerDefsC.h, [416](#)
- SPINNAKER_LOG_LEVEL_NOTICE
SpinnakerDefsC.h, [416](#)
- SPINNAKER_LOG_LEVEL_NOTSET
SpinnakerDefsC.h, [416](#)
- SPINNAKER_LOG_LEVEL_OFF
SpinnakerDefsC.h, [416](#)
- SPINNAKER_LOG_LEVEL_WARN
SpinnakerDefsC.h, [416](#)
- SPINNAKER_PIXELFORMAT_NAMESPACE_CUSTOM_ID
SpinnakerDefsC.h, [418](#)
- SPINNAKER_STATISTICS_CHANNEL_BLUE
SpinnakerDefsC.h, [417](#)
- SPINNAKER_STATISTICS_CHANNEL_GREEN
SpinnakerDefsC.h, [417](#)
- SPINNAKER_STATISTICS_CHANNEL_GREY
SpinnakerDefsC.h, [417](#)
- SPINNAKER_STATISTICS_CHANNEL_HUE
SpinnakerDefsC.h, [417](#)
- SPINNAKER_STATISTICS_CHANNEL_LIGHTNESS
SpinnakerDefsC.h, [417](#)
- SPINNAKER_STATISTICS_CHANNEL_NUM_CHANNELS
SpinnakerDefsC.h, [417](#)
- SPINNAKER_STATISTICS_CHANNEL_RED
SpinnakerDefsC.h, [417](#)
- SPINNAKER_STATISTICS_CHANNEL_SATURATION
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TIFF_COMPRESS_METHOD_ADOBE_DEFLATE
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TIFF_COMPRESS_METHOD_CCITTFAX3
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TIFF_COMPRESS_METHOD_CCITTFAX4
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TIFF_COMPRESS_METHOD_DEFLATE
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TIFF_COMPRESS_METHOD_JPG
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TIFF_COMPRESS_METHOD_LZW
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TIFF_COMPRESS_METHOD_NONE
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TIFF_COMPRESS_METHOD_PACKBITS
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TLPAYLOAD_TYPE_CHUNK_DATA
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TLPAYLOAD_TYPE_CHUNK_ONLY
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TLPAYLOAD_TYPE_CUSTOM_ID
SpinnakerDefsC.h, [418](#)
- SPINNAKER_TLPAYLOAD_TYPE_DEVICE_SPECIFIC
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TLPAYLOAD_TYPE_FILE
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TLPAYLOAD_TYPE_H264
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TLPAYLOAD_TYPE_IMAGE
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TLPAYLOAD_TYPE_JPEG
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TLPAYLOAD_TYPE_JPEG2000
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TLPAYLOAD_TYPE_JPEG_LOSSLESS_COMPRESSED
SpinnakerDefsC.h, [418](#)
- SPINNAKER_TLPAYLOAD_TYPE_LOSSLESS_COMPRESSED
SpinnakerDefsC.h, [418](#)
- SPINNAKER_TLPAYLOAD_TYPE_LOSSY_COMPRESSED
SpinnakerDefsC.h, [418](#)
- SPINNAKER_TLPAYLOAD_TYPE_MULTI_PART
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TLPAYLOAD_TYPE_RAW_DATA
SpinnakerDefsC.h, [417](#)
- SPINNAKER_TLPAYLOAD_TYPE_UNKNOWN
SpinnakerDefsC.h, [417](#)

- SPINNAKER_TLPIXELFORMAT_NAMESPACE_GEV
 - SpinnakerDefsC.h, [418](#)
- SPINNAKER_TLPIXELFORMAT_NAMESPACE_IIDC
 - SpinnakerDefsC.h, [418](#)
- SPINNAKER_TLPIXELFORMAT_NAMESPACE_PFNC_16BIT
 - SpinnakerDefsC.h, [418](#)
- SPINNAKER_TLPIXELFORMAT_NAMESPACE_PFNC_32BIT
 - SpinnakerDefsC.h, [418](#)
- SPINNAKER_TLPIXELFORMAT_NAMESPACE_UNKNOWN
 - SpinnakerDefsC.h, [418](#)
- SpinnakerC.h
 - spinCameraBeginAcquisition, [307](#)
 - spinCameraDelInit, [308](#)
 - spinCameraDiscoverMaxPacketSize, [308](#)
 - spinCameraEndAcquisition, [309](#)
 - spinCameraForceIP, [309](#)
 - spinCameraGetAccessMode, [309](#)
 - spinCameraGetGuiXml, [310](#)
 - spinCameraGetNextImage, [310](#)
 - spinCameraGetNextImageEx, [311](#)
 - spinCameraGetNextImageSync, [311](#)
 - spinCameraGetNodeMap, [312](#)
 - spinCameraGetTLDeviceNodeMap, [313](#)
 - spinCameraGetTLStreamNodeMap, [313](#)
 - spinCameraGetUniqueID, [313](#)
 - spinCameraInit, [314](#)
 - spinCamerasInitialized, [314](#)
 - spinCamerasStreaming, [315](#)
 - spinCamerasValid, [315](#)
 - spinCameraListAppend, [316](#)
 - spinCameraListClear, [316](#)
 - spinCameraListCreateEmpty, [317](#)
 - spinCameraListDestroy, [317](#)
 - spinCameraListGet, [318](#)
 - spinCameraListGetBySerial, [318](#)
 - spinCameraListGetSize, [319](#)
 - spinCameraListRemove, [319](#)
 - spinCameraListRemoveBySerial, [320](#)
 - spinCameraReadPort, [320](#)
 - spinCameraRegisterDeviceEventHandler, [320](#)
 - spinCameraRegisterDeviceEventHandlerEx, [321](#)
 - spinCameraRegisterImageEventHandler, [321](#)
 - spinCameraRegisterImageEventHandlerEx, [322](#)
 - spinCameraRegisterImageListEventHandler, [322](#)
 - spinCameraRelease, [323](#)
 - spinCameraUnregisterDeviceEventHandler, [323](#)
 - spinCameraUnregisterImageEventHandler, [324](#)
 - spinCameraUnregisterImageListEventHandler, [324](#)
 - spinCameraWritePort, [325](#)
 - spinDeviceArrivalEventHandlerCreate, [325](#)
 - spinDeviceArrivalEventHandlerDestroy, [325](#)
 - spinDeviceEventGetId, [326](#)
 - spinDeviceEventGetName, [326](#)
 - spinDeviceEventGetPayloadData, [327](#)
 - spinDeviceEventGetPayloadDataSize, [327](#)
 - spinDeviceEventHandlerCreate, [328](#)
 - spinDeviceEventHandlerDestroy, [328](#)
 - spinDeviceRemovalEventHandlerCreate, [329](#)
 - spinDeviceRemovalEventHandlerDestroy, [329](#)
 - spinErrorGetLast, [330](#)
 - spinErrorGetLastBuildDate, [330](#)
 - spinErrorGetLastBuildTime, [331](#)
 - spinErrorGetLastFileName, [331](#)
 - spinErrorGetLastFullMessage, [332](#)
 - spinErrorGetLastFunctionName, [332](#)
 - spinErrorGetLastLineNumber, [333](#)
 - spinErrorGetLastMessage, [333](#)
 - spinImageCalculateStatistics, [334](#)
 - spinImageCheckCRC, [334](#)
 - spinImageChunkDataGetFloatValue, [335](#)
 - spinImageChunkDataGetIntValue, [335](#)
 - spinImageCreate, [335](#)
 - spinImageCreateEmpty, [336](#)
 - spinImageCreateEx, [336](#)
 - spinImageCreateEx2, [337](#)
 - spinImageDeepCopy, [337](#)
 - spinImageDestroy, [338](#)
 - spinImageEventHandlerCreate, [338](#)
 - spinImageEventHandlerDestroy, [339](#)
 - spinImageGetBitsPerPixel, [339](#)
 - spinImageGetBufferSize, [340](#)
 - spinImageGetChunkLayoutID, [340](#)
 - spinImageGetColorProcessing, [341](#)
 - spinImageGetData, [341](#)
 - spinImageGetFrameID, [342](#)
 - spinImageGetHeight, [342](#)
 - spinImageGetID, [343](#)
 - spinImageGetOffsetX, [343](#)
 - spinImageGetOffsetY, [344](#)
 - spinImageGetPaddingX, [344](#)
 - spinImageGetPaddingY, [345](#)
 - spinImageGetPayloadType, [345](#)
 - spinImageGetPixelFormat, [346](#)
 - spinImageGetPixelFormatName, [346](#)
 - spinImageGetPrivateData, [347](#)
 - spinImageGetSize, [347](#)
 - spinImageGetStatus, [348](#)
 - spinImageGetStatusDescription, [348](#)
 - spinImageGetStride, [349](#)
 - spinImageGetTimeStamp, [349](#)
 - spinImageGetTLPayloadType, [350](#)
 - spinImageGetTLPixelFormat, [350](#)
 - spinImageGetTLPixelFormatNamespace, [351](#)
 - spinImageGetValidPayloadSize, [351](#)
 - spinImageGetWidth, [352](#)
 - spinImageHasCRC, [352](#)
 - spinImageIsIncomplete, [353](#)
 - spinImageListAppend, [353](#)
 - spinImageListClear, [354](#)
 - spinImageListCreateEmpty, [354](#)
 - spinImageListDestroy, [355](#)
 - spinImageListEventHandlerCreate, [355](#)
 - spinImageListEventHandlerDestroy, [356](#)
 - spinImageListGet, [356](#)
 - spinImageListGetByPixelFormat, [357](#)
 - spinImageListGetSize, [357](#)

- spinImageListLoad, 358
- spinImageListRelease, 358
- spinImageListRemove, 358
- spinImageListRemoveByPixelFormat, 359
- spinImageListSave, 359
- spinImageProcessorApplyGamma, 360
- spinImageProcessorConvert, 360
- spinImageProcessorConvertImageList, 361
- spinImageProcessorCreate, 362
- spinImageProcessorDestroy, 362
- spinImageProcessorGetColorProcessing, 363
- spinImageProcessorGetNumDecompression-Threads, 363
- spinImageProcessorSetColorProcessing, 363
- spinImageProcessorSetNumDecompression-Threads, 364
- spinImageRelease, 364
- spinImageReset, 365
- spinImageResetEx, 365
- spinImageSave, 366
- spinImageSaveBmp, 367
- spinImageSaveFromExt, 367
- spinImageSaveJpeg, 367
- spinImageSaveJpg2, 368
- spinImageSavePgm, 368
- spinImageSavePng, 369
- spinImageSavePpm, 369
- spinImageSaveTiff, 370
- spinImageStatisticsCreate, 370
- spinImageStatisticsDestroy, 371
- spinImageStatisticsDisableAll, 371
- spinImageStatisticsEnableAll, 372
- spinImageStatisticsEnableGreyOnly, 372
- spinImageStatisticsEnableHslOnly, 373
- spinImageStatisticsEnableRgbOnly, 373
- spinImageStatisticsGetAll, 373
- spinImageStatisticsGetChannelStatus, 374
- spinImageStatisticsGetHistogram, 375
- spinImageStatisticsGetMean, 375
- spinImageStatisticsGetNumPixelValues, 376
- spinImageStatisticsGetPixelValueRange, 376
- spinImageStatisticsGetRange, 377
- spinImageStatisticsSetChannelStatus, 377
- spinInterfaceEventHandlerCreate, 378
- spinInterfaceEventHandlerDestroy, 378
- spinInterfaceGetCameras, 379
- spinInterfaceGetCamerasEx, 379
- spinInterfaceGetTLNodeMap, 380
- spinInterfaceIsInUse, 380
- spinInterfaceListClear, 381
- spinInterfaceListCreateEmpty, 381
- spinInterfaceListDestroy, 382
- spinInterfaceListGet, 382
- spinInterfaceListGetSize, 383
- spinInterfaceRegisterDeviceArrivalEventHandler, 383
- spinInterfaceRegisterDeviceRemovalEventHandler, 384
- spinInterfaceRegisterInterfaceEventHandler, 384
- spinInterfaceRelease, 385
- spinInterfaceSendActionCommand, 385
- spinInterfaceUnregisterDeviceArrivalEventHandler, 386
- spinInterfaceUnregisterDeviceRemovalEventHandler, 386
- spinInterfaceUnregisterInterfaceEventHandler, 387
- spinInterfaceUpdateCameras, 387
- spinLogDataGetCategoryName, 388
- spinLogDataGetLogMessage, 388
- spinLogDataGetNDC, 389
- spinLogDataGetPriority, 389
- spinLogDataGetPriorityName, 390
- spinLogDataGetThreadName, 390
- spinLogDataGetTimestamp, 391
- spinLogEventHandlerCreate, 391
- spinLogEventHandlerDestroy, 392
- spinSystemGetCameras, 392
- spinSystemGetCamerasEx, 393
- spinSystemGetInstance, 393
- spinSystemGetInterfaces, 394
- spinSystemGetLibraryVersion, 394
- spinSystemGetLoggingLevel, 394
- spinSystemGetTLNodeMap, 395
- spinSystemIsInUse, 395
- spinSystemRegisterDeviceArrivalEventHandler, 396
- spinSystemRegisterDeviceRemovalEventHandler, 396
- spinSystemRegisterInterfaceEventHandler, 397
- spinSystemRegisterLogEventHandler, 397
- spinSystemReleaseInstance, 398
- spinSystemSendActionCommand, 398
- spinSystemSetLoggingLevel, 399
- spinSystemUnregisterAllLogEventHandlers, 400
- spinSystemUnregisterDeviceArrivalEventHandler, 400
- spinSystemUnregisterDeviceRemovalEventHandler, 401
- spinSystemUnregisterInterfaceEventHandler, 401
- spinSystemUnregisterLogEventHandler, 402
- spinSystemUpdateCameras, 402
- spinSystemUpdateCamerasEx, 403
- SPINNAKERC_API
 - SpinnakerPlatformC.h, 471
- SpinnakerDefsC.h
 - bool8_t, 408
 - False, 418
 - spinActionCommandStatus, 412
 - spinArrivalEventFunction, 408
 - spinCamera, 408
 - spinCameraList, 408
 - spinColorProcessingAlgorithm, 413
 - spinDeviceArrivalEventHandler, 409
 - spinDeviceEventData, 409
 - spinDeviceEventFunction, 409
 - spinDeviceEventHandler, 409

- spinDeviceRemovalEventHandler, [409](#)
- spinError, [413](#)
- spinImage, [409](#)
- spinImageEventFunction, [410](#)
- spinImageEventHandler, [410](#)
- spinImageFileFormat, [415](#)
- spinImageList, [410](#)
- spinImageListEventFunction, [410](#)
- spinImageListEventHandler, [410](#)
- spinImageProcessor, [410](#)
- spinImageStatistics, [411](#)
- spinImageStatus, [415](#)
- spinInterface, [411](#)
- spinInterfaceEventHandler, [411](#)
- spinInterfaceList, [411](#)
- spinLogEventData, [411](#)
- spinLogEventFunction, [411](#)
- spinLogEventHandler, [412](#)
- SPINNAKER_ACTION_COMMAND_STATUS_ACTION_LAST, [413](#)
- SPINNAKER_ACTION_COMMAND_STATUS_ERROR, [413](#)
- SPINNAKER_ACTION_COMMAND_STATUS_NO_REF_TARGET, [413](#)
- SPINNAKER_ACTION_COMMAND_STATUS_OK, [413](#)
- SPINNAKER_ACTION_COMMAND_STATUS_OVERFLOW, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_BILinear, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_DIRECT, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_EDGE_SENSING, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_HQ_LINEAR, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_IPP, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_NEAREST, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_NEAREST_INTEGER, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_NONE, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_RIGOROUS, [413](#)
- SPINNAKER_COLOR_PROCESSING_ALGORITHM_WEIGHTED, [413](#)
- SPINNAKER_ERR_ABORT, [414](#)
- SPINNAKER_ERR_ACCESS_DENIED, [414](#)
- SPINNAKER_ERR_BUFFER_TOO_SMALL, [414](#)
- SPINNAKER_ERR_BUSY, [414](#)
- SPINNAKER_ERR_CUSTOM_ID, [415](#)
- SPINNAKER_ERR_ERROR, [414](#)
- SPINNAKER_ERR_GENICAM_ACCESS, [414](#)
- SPINNAKER_ERR_GENICAM_BAD_ALLOCATION, [414](#)
- SPINNAKER_ERR_GENICAM_DYNAMIC_CAST, [414](#)
- SPINNAKER_ERR_GENICAM_GENERIC, [414](#)
- SPINNAKER_ERR_GENICAM_INVALID_ARGUMENT, [414](#)
- SPINNAKER_ERR_GENICAM_LOGICAL, [414](#)
- SPINNAKER_ERR_GENICAM_OUT_OF_RANGE, [414](#)
- SPINNAKER_ERR_GENICAM_PROPERTY, [414](#)
- SPINNAKER_ERR_GENICAM_RUN_TIME, [414](#)
- SPINNAKER_ERR_GENICAM_TIMEOUT, [414](#)
- SPINNAKER_ERR_IM_COLOR_CONVERSION, [414](#)
- SPINNAKER_ERR_IM_CONVERT, [414](#)
- SPINNAKER_ERR_IM_COPY, [414](#)
- SPINNAKER_ERR_IM_HISTOGRAM_MEAN, [414](#)
- SPINNAKER_ERR_IM_HISTOGRAM_RANGE, [414](#)
- SPINNAKER_ERR_IM_MALLOC, [414](#)
- SPINNAKER_ERR_IM_MIN_MAX, [414](#)
- SPINNAKER_ERR_IM_NOT_SUPPORTED, [414](#)
- SPINNAKER_ERR_INVALID_ADDRESS, [414](#)
- SPINNAKER_ERR_INVALID_BUFFER, [414](#)
- SPINNAKER_ERR_INVALID_HANDLE, [414](#)
- SPINNAKER_ERR_INVALID_ID, [414](#)
- SPINNAKER_ERR_INVALID_INDEX, [414](#)
- SPINNAKER_ERR_INVALID_PARAMETER, [414](#)
- SPINNAKER_ERR_INVALID_VALUE, [414](#)
- SPINNAKER_ERR_IO, [414](#)
- SPINNAKER_ERR_NO_DATA, [414](#)
- SPINNAKER_ERR_NOT_AVAILABLE, [414](#)
- SPINNAKER_ERR_NOT_IMPLEMENTED, [414](#)
- SPINNAKER_ERR_NOT_INITIALIZED, [414](#)
- SPINNAKER_ERR_OUT_OF_MEMORY, [414](#)
- SPINNAKER_ERR_PARSING_CHUNK_DATA, [414](#)
- SPINNAKER_ERR_RESOURCE_EXHAUSTED, [414](#)
- SPINNAKER_ERR_RESOURCE_IN_USE, [414](#)
- SPINNAKER_ERR_SUCCESS, [414](#)
- SPINNAKER_ERR_TIMEOUT, [414](#)
- SPINNAKER_IMAGE_FILE_FORMAT_BMP, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_FORCE_32BITS, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_FROM_FILE_EXT, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_JPEG, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_JPEG2000, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_PGM, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_PNG, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_PPM, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_RAW, [415](#)
- SPINNAKER_IMAGE_FILE_FORMAT_TIFF, [415](#)
- SPINNAKER_IMAGE_STATUS_CHUNK_DATA_INVALID, [416](#)
- SPINNAKER_IMAGE_STATUS_CRC_CHECK_FAILED, [415](#)
- SPINNAKER_IMAGE_STATUS_DATA_INCOMPLETE, [415](#)

- 416
- SPINNAKER_IMAGE_STATUS_DATA_OVERFLOW, 415
- SPINNAKER_IMAGE_STATUS_INFO_INCONSISTENT, 416
- SPINNAKER_IMAGE_STATUS_LEADER_BUFFER_SIZE_INCONSISTENT, 415
- SPINNAKER_IMAGE_STATUS_MISSING_LEADER, 416
- SPINNAKER_IMAGE_STATUS_MISSING_PACKETS, 415
- SPINNAKER_IMAGE_STATUS_MISSING_TRAILER, 416
- SPINNAKER_IMAGE_STATUS_NO_ERROR, 415
- SPINNAKER_IMAGE_STATUS_NO_SYSTEM_RESOURCES, 416
- SPINNAKER_IMAGE_STATUS_PACKETID_INCONSISTENT, 416
- SPINNAKER_IMAGE_STATUS_TRAILER_BUFFER_SIZE_INCONSISTENT, 416
- SPINNAKER_IMAGE_STATUS_UNKNOWN_ERROR, 415
- SPINNAKER_LOG_LEVEL_ALERT, 416
- SPINNAKER_LOG_LEVEL_CRIT, 416
- SPINNAKER_LOG_LEVEL_DEBUG, 416
- SPINNAKER_LOG_LEVEL_ERROR, 416
- SPINNAKER_LOG_LEVEL_FATAL, 416
- SPINNAKER_LOG_LEVEL_INFO, 416
- SPINNAKER_LOG_LEVEL_NOTICE, 416
- SPINNAKER_LOG_LEVEL_NOTSET, 416
- SPINNAKER_LOG_LEVEL_OFF, 416
- SPINNAKER_LOG_LEVEL_WARN, 416
- SPINNAKER_PIXELFORMAT_NAMESPACE_CUSTOM_ID, 418
- SPINNAKER_STATISTICS_CHANNEL_BLUE, 417
- SPINNAKER_STATISTICS_CHANNEL_GREEN, 417
- SPINNAKER_STATISTICS_CHANNEL_GREY, 417
- SPINNAKER_STATISTICS_CHANNEL_HUE, 417
- SPINNAKER_STATISTICS_CHANNEL_LIGHTNESS, 417
- SPINNAKER_STATISTICS_CHANNEL_NUM_CHANNELS, 417
- SPINNAKER_STATISTICS_CHANNEL_RED, 417
- SPINNAKER_STATISTICS_CHANNEL_SATURATION, 417
- SPINNAKER_TIFF_COMPRESS_METHOD_ADOBE_DEFLATE, 417
- SPINNAKER_TIFF_COMPRESS_METHOD_CCITTFAX3, 417
- SPINNAKER_TIFF_COMPRESS_METHOD_CCITTFAX4, 417
- SPINNAKER_TIFF_COMPRESS_METHOD_DEFLATE, 417
- SPINNAKER_TIFF_COMPRESS_METHOD_JPG, 417
- SPINNAKER_TIFF_COMPRESS_METHOD_LZW, 417
- SPINNAKER_TIFF_COMPRESS_METHOD_NONE, 417
- SPINNAKER_TIFF_COMPRESS_METHOD_PACKBITS, 417
- SPINNAKER_TLPAYLOAD_TYPE_CHUNK_DATA, 417
- SPINNAKER_TLPAYLOAD_TYPE_CHUNK_ONLY, 417
- SPINNAKER_TLPAYLOAD_TYPE_CUSTOM_ID, 418
- SPINNAKER_TLPAYLOAD_TYPE_DEVICE_SPECIFIC, 417
- SPINNAKER_TLPAYLOAD_TYPE_FILE, 417
- SPINNAKER_TLPAYLOAD_TYPE_H264, 417
- SPINNAKER_TLPAYLOAD_TYPE_IMAGE, 417
- SPINNAKER_TLPAYLOAD_TYPE_JPEG, 417
- SPINNAKER_TLPAYLOAD_TYPE_JPEG2000, 417
- SPINNAKER_TLPAYLOAD_TYPE_JPEG_LOSSLESS_COMPRESSED, 418
- SPINNAKER_TLPAYLOAD_TYPE_LOSSLESS_COMPRESSED, 418
- SPINNAKER_TLPAYLOAD_TYPE_LOSSY_COMPRESSED, 418
- SPINNAKER_TLPAYLOAD_TYPE_MULTI_PART, 417
- SPINNAKER_TLPAYLOAD_TYPE_RAW_DATA, 417
- SPINNAKER_TLPAYLOAD_TYPE_UNKNOWN, 417
- SPINNAKER_TLPIXELFORMAT_NAMESPACE_GEV, 418
- SPINNAKER_TLPIXELFORMAT_NAMESPACE_IIDC, 418
- SPINNAKER_TLPIXELFORMAT_NAMESPACE_PFNC_16BIT, 418
- SPINNAKER_TLPIXELFORMAT_NAMESPACE_PFNC_32BIT, 418
- SPINNAKER_TLPIXELFORMAT_NAMESPACE_UNKNOWN, 418
- spinnakerLogLevel, 416
- spinRemovalEventFunction, 412
- spinStatisticsChannel, 416
- spinSystem, 412
- spinTIFFCompressionMethod, 417
- spinTLPayloadType, 417
- spinPixelFormatNamespace, 418
- spinVideo, 412
- SpinnakerGenApiC.h, True, 418
- spinBooleanGetValue, 423
- spinBooleanSetValue, 423
- spinCategoryGetFeatureByIndex, 424
- spinCategoryGetNumFeatures, 424
- spinCategoryReleaseNode, 425
- spinCommandExecute, 425

- [spinCommandIsDone](#), 426
- [spinEnumerationEntryGetEnumValue](#), 426
- [spinEnumerationEntryGetIntValue](#), 427
- [spinEnumerationEntryGetSymbolic](#), 427
- [spinEnumerationGetCurrentEntry](#), 428
- [spinEnumerationGetEntryByIndex](#), 428
- [spinEnumerationGetEntryByName](#), 429
- [spinEnumerationGetNumEntries](#), 429
- [spinEnumerationReleaseNode](#), 430
- [spinEnumerationSetEnumValue](#), 430
- [spinEnumerationSetIntValue](#), 431
- [spinFloatGetMax](#), 431
- [spinFloatGetMin](#), 432
- [spinFloatGetRepresentation](#), 432
- [spinFloatGetUnit](#), 433
- [spinFloatGetValue](#), 433
- [spinFloatGetValueEx](#), 434
- [spinFloatSetValue](#), 434
- [spinFloatSetValueEx](#), 435
- [spinIntegerGetInc](#), 435
- [spinIntegerGetMax](#), 436
- [spinIntegerGetMin](#), 436
- [spinIntegerGetRepresentation](#), 437
- [spinIntegerGetValue](#), 437
- [spinIntegerGetValueEx](#), 438
- [spinIntegerSetValue](#), 438
- [spinIntegerSetValueEx](#), 439
- [spinNodeDeregisterCallback](#), 439
- [spinNodeFromString](#), 440
- [spinNodeFromStringEx](#), 440
- [spinNodeGetAccessMode](#), 441
- [spinNodeGetCachingMode](#), 441
- [spinNodeGetDescription](#), 442
- [spinNodeGetDisplayName](#), 442
- [spinNodeGetImposedAccessMode](#), 443
- [spinNodeGetImposedVisibility](#), 443
- [spinNodeGetName](#), 444
- [spinNodeGetNameSpace](#), 444
- [spinNodeGetPollingTime](#), 445
- [spinNodeGetToolTip](#), 445
- [spinNodeGetType](#), 446
- [spinNodeGetVisibility](#), 446
- [spinNodeInvalidateNode](#), 447
- [spinNodesAvailable](#), 447
- [spinNodesEqual](#), 448
- [spinNodesImplemented](#), 448
- [spinNodesReadable](#), 449
- [spinNodesWritable](#), 449
- [spinNodeMapGetNode](#), 450
- [spinNodeMapGetNodeByIndex](#), 450
- [spinNodeMapGetNumNodes](#), 451
- [spinNodeMapPoll](#), 451
- [spinNodeMapReleaseNode](#), 452
- [spinNodeRegisterCallback](#), 452
- [spinNodeToString](#), 453
- [spinNodeToStringEx](#), 453
- [spinRegisterGet](#), 454
- [spinRegisterGetAddress](#), 454

- [spinRegisterGetEx](#), 455
- [spinRegisterGetLength](#), 455
- [spinRegisterSet](#), 456
- [spinRegisterSetEx](#), 456
- [spinRegisterSetReference](#), 457
- [spinStringGetMaxLength](#), 457
- [spinStringGetValue](#), 458
- [spinStringGetValueEx](#), 458
- [spinStringSetValue](#), 459
- [spinStringSetValueEx](#), 459
- SpinnakerGenApiDefs.h
 - [_CycleDetectAccesMode](#), 464
 - [_UndefinedAccesMode](#), 464
 - [_UndefinedCachingMode](#), 464
 - [_UndefinedEDisplayNotation](#), 464
 - [_UndefinedESlope](#), 469
 - [_UndefinedEXMLValidation](#), 470
 - [_UndefinedEndian](#), 465
 - [_UndefinedNameSpace](#), 467
 - [_UndefinedRepresentation](#), 468
 - [_UndefinedSign](#), 468
 - [_UndefinedStandardNameSpace](#), 469
 - [_UndefinedVisibility](#), 469
 - [_UndefinedYesNo](#), 470
- [Automatic](#), 469
- [BaseNode](#), 467
- [Beginner](#), 469
- [BigEndian](#), 465
- [Boolean](#), 468
- [BooleanNode](#), 467
- [CategoryNode](#), 468
- [CL](#), 469
- [CommandNode](#), 467
- [ctAllDependingNodes](#), 467
- [ctAllTerminalNodes](#), 467
- [ctDependingChildren](#), 467
- [ctInvalidators](#), 467
- [ctReadingChildren](#), 467
- [ctWritingChildren](#), 467
- [Custom](#), 467
- [Decreasing](#), 469
- [EnumEntryNode](#), 468
- [EnumerationNode](#), 468
- [Expert](#), 469
- [fixedIncrement](#), 465
- [FloatNode](#), 467
- [fnAutomatic](#), 464
- [fnFixed](#), 464
- [fnScientific](#), 464
- [GEV](#), 469
- [Guru](#), 469
- [HexNumber](#), 468
- [idFrom](#), 465
- [idNone](#), 465
- [idTo](#), 465
- [IIDC](#), 469
- [Increasing](#), 469
- [IntegerNode](#), 467

- intflBase, [466](#)
- intflBoolean, [466](#)
- intflCategory, [466](#)
- intflCommand, [466](#)
- intflEnumEntry, [466](#)
- intflEnumeration, [466](#)
- intflFloat, [466](#)
- intflInteger, [466](#)
- intflPort, [466](#)
- intflRegister, [466](#)
- intflString, [466](#)
- intflValue, [466](#)
- Invisible, [469](#)
- IPv4Address, [468](#)
- Linear, [468](#)
- listIncrement, [465](#)
- LittleEndian, [465](#)
- Logarithmic, [468](#)
- MACAddress, [468](#)
- NA, [464](#)
- NI, [464](#)
- No, [470](#)
- NoCache, [464](#)
- noIncrement, [465](#)
- None, [469](#)
- PortNode, [468](#)
- PureNumber, [468](#)
- RegisterNode, [468](#)
- RO, [464](#)
- RW, [464](#)
- Signed, [468](#)
- spinAccessMode, [463](#)
- spinCachingMode, [464](#)
- spinDisplayNotation, [464](#)
- spinEndianess, [464](#)
- spinIncMode, [465](#)
- spinInputDirection, [465](#)
- spinInterfaceType, [465](#)
- spinLinkType, [466](#)
- spinNameSpace, [467](#)
- spinNodeCallbackFunction, [463](#)
- spinNodeCallbackHandle, [463](#)
- spinNodeHandle, [463](#)
- spinNodeMapHandle, [463](#)
- spinNodeType, [467](#)
- spinRepresentation, [468](#)
- spinSign, [468](#)
- spinSlope, [468](#)
- spinStandardNameSpace, [469](#)
- spinVisibility, [469](#)
- spinXMLValidation, [470](#)
- spinYesNo, [470](#)
- Standard, [467](#)
- StringNode, [467](#)
- UnknownNode, [468](#)
- Unsigned, [468](#)
- USB, [469](#)
- ValueNode, [467](#)
- Varying, [469](#)
- WO, [464](#)
- WriteAround, [464](#)
- WriteThrough, [464](#)
- xvAll, [470](#)
- xvCycles, [470](#)
- xvDefault, [470](#)
- xvLoad, [470](#)
- xvSFNC, [470](#)
- Yes, [470](#)
- spinnakerLogLevel
 - SpinnakerDefsC.h, [416](#)
- SpinnakerPlatformC.h
 - SPINNAKER_API, [471](#)
- spinNameSpace
 - SpinnakerGenApiDefsC.h, [467](#)
- spinNodeCallbackFunction
 - SpinnakerGenApiDefsC.h, [463](#)
- spinNodeCallbackHandle
 - SpinnakerGenApiDefsC.h, [463](#)
- spinNodeDeregisterCallback
 - SpinnakerGenApiC.h, [439](#)
- spinNodeFromString
 - SpinnakerGenApiC.h, [440](#)
- spinNodeFromStringEx
 - SpinnakerGenApiC.h, [440](#)
- spinNodeGetAccessMode
 - SpinnakerGenApiC.h, [441](#)
- spinNodeGetCachingMode
 - SpinnakerGenApiC.h, [441](#)
- spinNodeGetDescription
 - SpinnakerGenApiC.h, [442](#)
- spinNodeGetDisplayName
 - SpinnakerGenApiC.h, [442](#)
- spinNodeGetImposedAccessMode
 - SpinnakerGenApiC.h, [443](#)
- spinNodeGetImposedVisibility
 - SpinnakerGenApiC.h, [443](#)
- spinNodeGetName
 - SpinnakerGenApiC.h, [444](#)
- spinNodeGetNameSpace
 - SpinnakerGenApiC.h, [444](#)
- spinNodeGetPollingTime
 - SpinnakerGenApiC.h, [445](#)
- spinNodeGetToolTip
 - SpinnakerGenApiC.h, [445](#)
- spinNodeGetType
 - SpinnakerGenApiC.h, [446](#)
- spinNodeGetVisibility
 - SpinnakerGenApiC.h, [446](#)
- spinNodeHandle
 - SpinnakerGenApiDefsC.h, [463](#)
- spinNodeInvalidateNode
 - SpinnakerGenApiC.h, [447](#)
- spinNodesAvailable
 - SpinnakerGenApiC.h, [447](#)
- spinNodesEqual
 - SpinnakerGenApiC.h, [448](#)

- spinNodeIsImplemented
 - SpinnakerGenApiC.h, [448](#)
- spinNodeIsReadable
 - SpinnakerGenApiC.h, [449](#)
- spinNodeIsWritable
 - SpinnakerGenApiC.h, [449](#)
- spinNodeMapGetNode
 - SpinnakerGenApiC.h, [450](#)
- spinNodeMapGetNodeByIndex
 - SpinnakerGenApiC.h, [450](#)
- spinNodeMapGetNumNodes
 - SpinnakerGenApiC.h, [451](#)
- spinNodeMapHandle
 - SpinnakerGenApiDefsC.h, [463](#)
- spinNodeMapPoll
 - SpinnakerGenApiC.h, [451](#)
- spinNodeMapReleaseNode
 - SpinnakerGenApiC.h, [452](#)
- spinNodeRegisterCallback
 - SpinnakerGenApiC.h, [452](#)
- spinNodeToString
 - SpinnakerGenApiC.h, [453](#)
- spinNodeToStringEx
 - SpinnakerGenApiC.h, [453](#)
- spinNodeType
 - SpinnakerGenApiDefsC.h, [467](#)
- spinPGMOption, [170](#)
 - binaryFile, [170](#)
 - reserved, [170](#)
- spinPixelColorFilterEnums
 - CameraDefsC.h, [258](#)
- spinPixelFormatEnums
 - CameraDefsC.h, [259](#)
- spinPixelFormatInfoSelectorEnums
 - CameraDefsC.h, [264](#)
- spinPixelSizeEnums
 - CameraDefsC.h, [270](#)
- spinPNGOption, [171](#)
 - compressionLevel, [171](#)
 - interlaced, [171](#)
 - reserved, [171](#)
- spinPPMOption, [172](#)
 - binaryFile, [172](#)
 - reserved, [172](#)
- spinRegionDestinationEnums
 - CameraDefsC.h, [271](#)
- spinRegionModeEnums
 - CameraDefsC.h, [271](#)
- spinRegionSelectorEnums
 - CameraDefsC.h, [271](#)
- spinRegisterGet
 - SpinnakerGenApiC.h, [454](#)
- spinRegisterGetAddress
 - SpinnakerGenApiC.h, [454](#)
- spinRegisterGetEx
 - SpinnakerGenApiC.h, [455](#)
- spinRegisterGetLength
 - SpinnakerGenApiC.h, [455](#)
- spinRegisterSet
 - SpinnakerGenApiC.h, [456](#)
- spinRegisterSetEx
 - SpinnakerGenApiC.h, [456](#)
- spinRegisterSetReference
 - SpinnakerGenApiC.h, [457](#)
- spinRemovalEventFunction
 - SpinnakerDefsC.h, [412](#)
- spinRepresentation
 - SpinnakerGenApiDefsC.h, [468](#)
- spinRgbTransformLightSourceEnums
 - CameraDefsC.h, [272](#)
- spinScan3dCoordinateReferenceSelectorEnums
 - CameraDefsC.h, [272](#)
- spinScan3dCoordinateSelectorEnums
 - CameraDefsC.h, [273](#)
- spinScan3dCoordinateSystemEnums
 - CameraDefsC.h, [273](#)
- spinScan3dCoordinateSystemReferenceEnums
 - CameraDefsC.h, [273](#)
- spinScan3dCoordinateTransformSelectorEnums
 - CameraDefsC.h, [274](#)
- spinScan3dDistanceUnitEnums
 - CameraDefsC.h, [274](#)
- spinScan3dOutputModeEnums
 - CameraDefsC.h, [274](#)
- spinSensorDigitizationTapsEnums
 - CameraDefsC.h, [276](#)
- spinSensorShutterModeEnums
 - CameraDefsC.h, [277](#)
- spinSensorTapsEnums
 - CameraDefsC.h, [277](#)
- spinSequencerConfigurationModeEnums
 - CameraDefsC.h, [277](#)
- spinSequencerConfigurationValidEnums
 - CameraDefsC.h, [278](#)
- spinSequencerModeEnums
 - CameraDefsC.h, [278](#)
- spinSequencerSetValidEnums
 - CameraDefsC.h, [278](#)
- spinSequencerTriggerActivationEnums
 - CameraDefsC.h, [279](#)
- spinSequencerTriggerSourceEnums
 - CameraDefsC.h, [279](#)
- spinSerialPortBaudRateEnums
 - CameraDefsC.h, [279](#)
- spinSerialPortParityEnums
 - CameraDefsC.h, [280](#)
- spinSerialPortSelectorEnums
 - CameraDefsC.h, [280](#)
- spinSerialPortSourceEnums
 - CameraDefsC.h, [281](#)
- spinSerialPortStopBitsEnums
 - CameraDefsC.h, [281](#)
- spinSign
 - SpinnakerGenApiDefsC.h, [468](#)
- spinSlope
 - SpinnakerGenApiDefsC.h, [468](#)

- spinSoftwareSignalSelectorEnums
 - CameraDefsC.h, [281](#)
- spinSourceSelectorEnums
 - CameraDefsC.h, [282](#)
- spinStandardNameSpace
 - SpinnakerGenApiDefsC.h, [469](#)
- spinStatisticsChannel
 - SpinnakerDefsC.h, [416](#)
- spinStringGetMaxLength
 - SpinnakerGenApiC.h, [457](#)
- spinStringGetValue
 - SpinnakerGenApiC.h, [458](#)
- spinStringGetValueEx
 - SpinnakerGenApiC.h, [458](#)
- spinStringSetValue
 - SpinnakerGenApiC.h, [459](#)
- spinStringSetValueEx
 - SpinnakerGenApiC.h, [459](#)
- spinSystem
 - SpinnakerDefsC.h, [412](#)
- spinSystemGetCameras
 - SpinnakerC.h, [392](#)
- spinSystemGetCamerasEx
 - SpinnakerC.h, [393](#)
- spinSystemGetInstance
 - SpinnakerC.h, [393](#)
- spinSystemGetInterfaces
 - SpinnakerC.h, [394](#)
- spinSystemGetLibraryVersion
 - SpinnakerC.h, [394](#)
- spinSystemGetLoggingLevel
 - SpinnakerC.h, [394](#)
- spinSystemGetTLNodeMap
 - SpinnakerC.h, [395](#)
- spinSystemIsInUse
 - SpinnakerC.h, [395](#)
- spinSystemRegisterDeviceArrivalEventHandler
 - SpinnakerC.h, [396](#)
- spinSystemRegisterDeviceRemovalEventHandler
 - SpinnakerC.h, [396](#)
- spinSystemRegisterInterfaceEventHandler
 - SpinnakerC.h, [397](#)
- spinSystemRegisterLogEventHandler
 - SpinnakerC.h, [397](#)
- spinSystemReleaseInstance
 - SpinnakerC.h, [398](#)
- spinSystemSendActionCommand
 - SpinnakerC.h, [398](#)
- spinSystemSetLoggingLevel
 - SpinnakerC.h, [399](#)
- spinSystemUnregisterAllLogEventHandlers
 - SpinnakerC.h, [400](#)
- spinSystemUnregisterDeviceArrivalEventHandler
 - SpinnakerC.h, [400](#)
- spinSystemUnregisterDeviceRemovalEventHandler
 - SpinnakerC.h, [401](#)
- spinSystemUnregisterInterfaceEventHandler
 - SpinnakerC.h, [401](#)
- spinSystemUnregisterLogEventHandler
 - SpinnakerC.h, [402](#)
- spinSystemUpdateCameras
 - SpinnakerC.h, [402](#)
- spinSystemUpdateCamerasEx
 - SpinnakerC.h, [403](#)
- spinTestPatternEnums
 - CameraDefsC.h, [282](#)
- spinTestPatternGeneratorSelectorEnums
 - CameraDefsC.h, [282](#)
- spinTIFFCompressionMethod
 - SpinnakerDefsC.h, [417](#)
- spinTIFFOption, [173](#)
 - compression, [173](#)
 - reserved, [173](#)
- spinTimerSelectorEnums
 - CameraDefsC.h, [283](#)
- spinTimerStatusEnums
 - CameraDefsC.h, [283](#)
- spinTimerTriggerActivationEnums
 - CameraDefsC.h, [283](#)
- spinTimerTriggerSourceEnums
 - CameraDefsC.h, [284](#)
- spinTLDeviceAccessStatusEnums
 - TransportLayerDefsC.h, [476](#)
- spinTLDeviceCurrentSpeedEnums
 - TransportLayerDefsC.h, [476](#)
- spinTLDeviceEndianessMechanismEnums
 - TransportLayerDefsC.h, [476](#)
- spinTLDeviceTypeEnums
 - TransportLayerDefsC.h, [478](#)
- spinTLFilterDriverStatusEnums
 - TransportLayerDefsC.h, [478](#)
- spinTLGenICamXMLLocationEnums
 - TransportLayerDefsC.h, [478](#)
- spinTLGevCCPEnums
 - TransportLayerDefsC.h, [479](#)
- spinTLGUIXMLLocationEnums
 - TransportLayerDefsC.h, [479](#)
- spinTLInterfaceTypeEnums
 - TransportLayerDefsC.h, [479](#)
- spinTLPayloadType
 - SpinnakerDefsC.h, [417](#)
- spinTLPixelFormatNamespace
 - SpinnakerDefsC.h, [418](#)
- spinTLPOEStatusEnums
 - TransportLayerDefsC.h, [480](#)
- spinTLStreamBufferCountModeEnums
 - TransportLayerDefsC.h, [480](#)
- spinTLStreamBufferHandlingModeEnums
 - TransportLayerDefsC.h, [480](#)
- spinTLStreamModeEnums
 - TransportLayerDefsC.h, [481](#)
- spinTLStreamTypeEnums
 - TransportLayerDefsC.h, [481](#)
- spinTLTLTypeEnums
 - TransportLayerDefsC.h, [482](#)
- spinTransferComponentSelectorEnums

- CameraDefsC.h, [285](#)
- spinTransferControlModeEnums
 - CameraDefsC.h, [285](#)
- spinTransferOperationModeEnums
 - CameraDefsC.h, [286](#)
- spinTransferQueueModeEnums
 - CameraDefsC.h, [286](#)
- spinTransferSelectorEnums
 - CameraDefsC.h, [286](#)
- spinTransferStatusSelectorEnums
 - CameraDefsC.h, [287](#)
- spinTransferTriggerActivationEnums
 - CameraDefsC.h, [287](#)
- spinTransferTriggerModeEnums
 - CameraDefsC.h, [287](#)
- spinTransferTriggerSelectorEnums
 - CameraDefsC.h, [288](#)
- spinTransferTriggerSourceEnums
 - CameraDefsC.h, [288](#)
- spinTriggerActivationEnums
 - CameraDefsC.h, [289](#)
- spinTriggerModeEnums
 - CameraDefsC.h, [290](#)
- spinTriggerOverlapEnums
 - CameraDefsC.h, [290](#)
- spinTriggerSelectorEnums
 - CameraDefsC.h, [290](#)
- spinTriggerSourceEnums
 - CameraDefsC.h, [290](#)
- spinUserOutputSelectorEnums
 - CameraDefsC.h, [291](#)
- spinUserSetDefaultEnums
 - CameraDefsC.h, [291](#)
- spinUserSetSelectorEnums
 - CameraDefsC.h, [292](#)
- spinVideo
 - SpinnakerDefsC.h, [412](#)
- SpinVideo Recording Access, [32](#)
- spinVideoAppend
 - SpinVideoC.h, [472](#)
- SpinVideoC.h
 - spinVideoAppend, [472](#)
 - spinVideoClose, [472](#)
 - spinVideoOpenH264, [473](#)
 - spinVideoOpenMJPEG, [473](#)
 - spinVideoOpenUncompressed, [473](#)
 - spinVideoSetMaximumFileSize, [473](#)
- spinVideoClose
 - SpinVideoC.h, [472](#)
- spinVideoOpenH264
 - SpinVideoC.h, [473](#)
- spinVideoOpenMJPEG
 - SpinVideoC.h, [473](#)
- spinVideoOpenUncompressed
 - SpinVideoC.h, [473](#)
- spinVideoSetMaximumFileSize
 - SpinVideoC.h, [473](#)
- spinVisibility
 - SpinnakerGenApiDefsC.h, [469](#)
- spinWhiteClipSelectorEnums
 - CameraDefsC.h, [292](#)
- spinXMLValidation
 - SpinnakerGenApiDefsC.h, [470](#)
- spinYesNo
 - SpinnakerGenApiDefsC.h, [470](#)
- Standard
 - SpinnakerGenApiDefsC.h, [467](#)
- Status
 - actionCommandResult, [35](#)
- StreamAnnounceBufferMinimum
 - quickSpinTLStream, [147](#)
- StreamAnnouncedBufferCount
 - quickSpinTLStream, [147](#)
- StreamBlockTransferSize
 - quickSpinTLStream, [147](#)
- StreamBufferAlignment
 - quickSpinTLStream, [148](#)
- StreamBufferCountManual
 - quickSpinTLStream, [148](#)
- StreamBufferCountMax
 - quickSpinTLStream, [148](#)
- StreamBufferCountMode
 - quickSpinTLStream, [148](#)
- StreamBufferCountMode_Manual
 - TransportLayerDefsC.h, [480](#)
- StreamBufferCountResult
 - quickSpinTLStream, [148](#)
- StreamBufferHandlingMode
 - quickSpinTLStream, [148](#)
- StreamBufferHandlingMode_NewestFirst
 - TransportLayerDefsC.h, [481](#)
- StreamBufferHandlingMode_NewestOnly
 - TransportLayerDefsC.h, [481](#)
- StreamBufferHandlingMode_OldestFirst
 - TransportLayerDefsC.h, [481](#)
- StreamBufferHandlingMode_OldestFirstOverwrite
 - TransportLayerDefsC.h, [481](#)
- StreamChunkCountMaximum
 - quickSpinTLStream, [148](#)
- StreamCRCCheckEnable
 - quickSpinTLStream, [148](#)
- StreamDeliveredFrameCount
 - quickSpinTLStream, [149](#)
- StreamDroppedFrameCount
 - quickSpinTLStream, [149](#)
- StreamID
 - quickSpinTLStream, [149](#)
- StreamIncompleteFrameCount
 - quickSpinTLStream, [149](#)
- StreamInputBufferCount
 - quickSpinTLStream, [149](#)
- StreamIsGrabbing
 - quickSpinTLStream, [149](#)
- StreamLostFrameCount
 - quickSpinTLStream, [149](#)
- StreamMissedPacketCount

- quickSpinTLStream, [149](#)
- StreamMode
 - quickSpinTLStream, [150](#)
- StreamMode_LWF
 - TransportLayerDefsC.h, [481](#)
- StreamMode_MVA
 - TransportLayerDefsC.h, [481](#)
- StreamMode_Socket
 - TransportLayerDefsC.h, [481](#)
- StreamOutputBufferCount
 - quickSpinTLStream, [150](#)
- StreamPacketResendEnable
 - quickSpinTLStream, [150](#)
- StreamPacketResendMaxRequests
 - quickSpinTLStream, [150](#)
- StreamPacketResendReceivedPacketCount
 - quickSpinTLStream, [150](#)
- StreamPacketResendRequestCount
 - quickSpinTLStream, [150](#)
- StreamPacketResendRequestedPacketCount
 - quickSpinTLStream, [150](#)
- StreamPacketResendRequestSuccessCount
 - quickSpinTLStream, [150](#)
- StreamPacketResendTimeout
 - quickSpinTLStream, [151](#)
- StreamReceivedFrameCount
 - quickSpinTLStream, [151](#)
- StreamReceivedPacketCount
 - quickSpinTLStream, [151](#)
- StreamStartedFrameCount
 - quickSpinTLStream, [151](#)
- StreamType
 - quickSpinTLStream, [151](#)
- StreamType_CameraLink
 - TransportLayerDefsC.h, [482](#)
- StreamType_CameraLinkHS
 - TransportLayerDefsC.h, [482](#)
- StreamType_CoaxPress
 - TransportLayerDefsC.h, [482](#)
- StreamType_Custom
 - TransportLayerDefsC.h, [482](#)
- StreamType_GigEVision
 - TransportLayerDefsC.h, [482](#)
- StreamType_USB3Vision
 - TransportLayerDefsC.h, [482](#)
- String Access, [30](#)
- StringNode
 - SpinnakerGenApiDefsC.h, [467](#)
- System Access, [23](#)
- Test0001
 - quickSpin, [124](#)
- TestEventGenerate
 - quickSpin, [124](#)
- TestPattern
 - quickSpin, [124](#)
- TestPattern_Increment
 - CameraDefsC.h, [282](#)
- TestPattern_Off
 - CameraDefsC.h, [282](#)
- TestPattern_SensorTestPattern
 - CameraDefsC.h, [282](#)
- TestPatternGeneratorSelector
 - quickSpin, [125](#)
- TestPatternGeneratorSelector_PipelineStart
 - CameraDefsC.h, [282](#)
- TestPatternGeneratorSelector_Sensor
 - CameraDefsC.h, [282](#)
- TestPendingAck
 - quickSpin, [125](#)
- TimerDelay
 - quickSpin, [125](#)
- TimerDuration
 - quickSpin, [125](#)
- TimerReset
 - quickSpin, [125](#)
- TimerSelector
 - quickSpin, [125](#)
- TimerSelector_Timer0
 - CameraDefsC.h, [283](#)
- TimerSelector_Timer1
 - CameraDefsC.h, [283](#)
- TimerSelector_Timer2
 - CameraDefsC.h, [283](#)
- TimerStatus
 - quickSpin, [125](#)
- TimerStatus_TimerActive
 - CameraDefsC.h, [283](#)
- TimerStatus_TimerCompleted
 - CameraDefsC.h, [283](#)
- TimerStatus_TimerIdle
 - CameraDefsC.h, [283](#)
- TimerStatus_TimerTriggerWait
 - CameraDefsC.h, [283](#)
- TimerTriggerActivation
 - quickSpin, [125](#)
- TimerTriggerActivation_AnyEdge
 - CameraDefsC.h, [283](#)
- TimerTriggerActivation_FallingEdge
 - CameraDefsC.h, [283](#)
- TimerTriggerActivation_LevelHigh
 - CameraDefsC.h, [283](#)
- TimerTriggerActivation_LevelLow
 - CameraDefsC.h, [283](#)
- TimerTriggerActivation_RisingEdge
 - CameraDefsC.h, [283](#)
- TimerTriggerSource
 - quickSpin, [126](#)
- TimerTriggerSource_AcquisitionEnd
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_AcquisitionStart
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_AcquisitionTrigger
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Action0
 - CameraDefsC.h, [285](#)
- TimerTriggerSource_Action1

- CameraDefsC.h, [285](#)
- TimerTriggerSource_Action2
 - CameraDefsC.h, [285](#)
- TimerTriggerSource_Counter0End
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Counter0Start
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Counter1End
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Counter1Start
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Counter2End
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Counter2Start
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Encoder0
 - CameraDefsC.h, [285](#)
- TimerTriggerSource_Encoder1
 - CameraDefsC.h, [285](#)
- TimerTriggerSource_Encoder2
 - CameraDefsC.h, [285](#)
- TimerTriggerSource_ExposureEnd
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_ExposureStart
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_FrameBurstEnd
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_FrameBurstStart
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_FrameEnd
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_FrameStart
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_FrameTrigger
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Line0
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Line1
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Line2
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_LineEnd
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_LineStart
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_LineTrigger
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_LinkTrigger0
 - CameraDefsC.h, [285](#)
- TimerTriggerSource_LinkTrigger1
 - CameraDefsC.h, [285](#)
- TimerTriggerSource_LinkTrigger2
 - CameraDefsC.h, [285](#)
- TimerTriggerSource_Off
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_SoftwareSignal0
 - CameraDefsC.h, [285](#)
- TimerTriggerSource_SoftwareSignal1
 - CameraDefsC.h, [285](#)
- TimerTriggerSource_SoftwareSignal2
 - CameraDefsC.h, [285](#)
- TimerTriggerSource_Timer0End
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Timer0Start
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Timer1End
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Timer1Start
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Timer2End
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_Timer2Start
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_UserOutput0
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_UserOutput1
 - CameraDefsC.h, [284](#)
- TimerTriggerSource_UserOutput2
 - CameraDefsC.h, [284](#)
- TimerValue
 - quickSpin, [126](#)
- Timestamp
 - quickSpin, [126](#)
- TimestampLatch
 - quickSpin, [126](#)
- TimestampLatchValue
 - quickSpin, [126](#)
- TimestampReset
 - quickSpin, [126](#)
- TLDevice Structures, [32](#)
- TLDisplayName
 - quickSpinTLSystem, [154](#)
- TLFileName
 - quickSpinTLSystem, [155](#)
- TLID
 - quickSpinTLSystem, [155](#)
- TLInterface Structures, [32](#)
- TLModelName
 - quickSpinTLSystem, [155](#)
- TLParamsLocked
 - quickSpin, [126](#)
- TLPath
 - quickSpinTLSystem, [155](#)
- TLStream Structures, [33](#)
- TLSystem Structures, [33](#)
- TLType
 - quickSpinTLSystem, [155](#)
- TLType_CameraLink
 - TransportLayerDefsC.h, [482](#)
- TLType_CameraLinkHS
 - TransportLayerDefsC.h, [482](#)
- TLType_CoaxPress
 - TransportLayerDefsC.h, [482](#)
- TLType_Custom
 - TransportLayerDefsC.h, [482](#)
- TLType_GigEVision

- TransportLayerDefsC.h, [482](#)
- TlType_Mixed
 - TransportLayerDefsC.h, [482](#)
- TlType_USB3Vision
 - TransportLayerDefsC.h, [482](#)
- TlVendorName
 - quickSpinTlSystem, [155](#)
- TlVersion
 - quickSpinTlSystem, [155](#)
- TransferAbort
 - quickSpin, [126](#)
- TransferBlockCount
 - quickSpin, [127](#)
- TransferBurstCount
 - quickSpin, [127](#)
- TransferComponentSelector
 - quickSpin, [127](#)
- TransferComponentSelector_All
 - CameraDefsC.h, [285](#)
- TransferComponentSelector_Blue
 - CameraDefsC.h, [285](#)
- TransferComponentSelector_Green
 - CameraDefsC.h, [285](#)
- TransferComponentSelector_Red
 - CameraDefsC.h, [285](#)
- TransferControlMode
 - quickSpin, [127](#)
- TransferControlMode_Automatic
 - CameraDefsC.h, [286](#)
- TransferControlMode_Basic
 - CameraDefsC.h, [286](#)
- TransferControlMode_UserControlled
 - CameraDefsC.h, [286](#)
- TransferOperationMode
 - quickSpin, [127](#)
- TransferOperationMode_Continuous
 - CameraDefsC.h, [286](#)
- TransferOperationMode_MultiBlock
 - CameraDefsC.h, [286](#)
- TransferPause
 - quickSpin, [127](#)
- TransferQueueCurrentBlockCount
 - quickSpin, [127](#)
- TransferQueueMaxBlockCount
 - quickSpin, [127](#)
- TransferQueueMode
 - quickSpin, [128](#)
- TransferQueueMode_FirstInFirstOut
 - CameraDefsC.h, [286](#)
- TransferQueueOverflowCount
 - quickSpin, [128](#)
- TransferResume
 - quickSpin, [128](#)
- TransferSelector
 - quickSpin, [128](#)
- TransferSelector_All
 - CameraDefsC.h, [286](#)
- TransferSelector_Stream0
 - CameraDefsC.h, [286](#)
- TransferSelector_Stream1
 - CameraDefsC.h, [286](#)
- TransferSelector_Stream2
 - CameraDefsC.h, [286](#)
- TransferStart
 - quickSpin, [128](#)
- TransferStatus
 - quickSpin, [128](#)
- TransferStatusSelector
 - quickSpin, [128](#)
- TransferStatusSelector_Paused
 - CameraDefsC.h, [287](#)
- TransferStatusSelector_QueueOverflow
 - CameraDefsC.h, [287](#)
- TransferStatusSelector_Stopped
 - CameraDefsC.h, [287](#)
- TransferStatusSelector_Stopping
 - CameraDefsC.h, [287](#)
- TransferStatusSelector_Streaming
 - CameraDefsC.h, [287](#)
- TransferStop
 - quickSpin, [128](#)
- TransferStreamChannel
 - quickSpin, [129](#)
- TransferTriggerActivation
 - quickSpin, [129](#)
- TransferTriggerActivation_AnyEdge
 - CameraDefsC.h, [287](#)
- TransferTriggerActivation_FallingEdge
 - CameraDefsC.h, [287](#)
- TransferTriggerActivation_LevelHigh
 - CameraDefsC.h, [287](#)
- TransferTriggerActivation_LevelLow
 - CameraDefsC.h, [287](#)
- TransferTriggerActivation_RisingEdge
 - CameraDefsC.h, [287](#)
- TransferTriggerMode
 - quickSpin, [129](#)
- TransferTriggerMode_Off
 - CameraDefsC.h, [288](#)
- TransferTriggerMode_On
 - CameraDefsC.h, [288](#)
- TransferTriggerSelector
 - quickSpin, [129](#)
- TransferTriggerSelector_TransferAbort
 - CameraDefsC.h, [288](#)
- TransferTriggerSelector_TransferActive
 - CameraDefsC.h, [288](#)
- TransferTriggerSelector_TransferBurstStart
 - CameraDefsC.h, [288](#)
- TransferTriggerSelector_TransferBurstStop
 - CameraDefsC.h, [288](#)
- TransferTriggerSelector_TransferPause
 - CameraDefsC.h, [288](#)
- TransferTriggerSelector_TransferResume
 - CameraDefsC.h, [288](#)
- TransferTriggerSelector_TransferStart

- CameraDefsC.h, [288](#)
- TransferTriggerSelector_TransferStop
 - CameraDefsC.h, [288](#)
- TransferTriggerSource
 - quickSpin, [129](#)
- TransferTriggerSource_Action0
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_Action1
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_Action2
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_Counter0End
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_Counter0Start
 - CameraDefsC.h, [288](#)
- TransferTriggerSource_Counter1End
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_Counter1Start
 - CameraDefsC.h, [288](#)
- TransferTriggerSource_Counter2End
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_Counter2Start
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_Line0
 - CameraDefsC.h, [288](#)
- TransferTriggerSource_Line1
 - CameraDefsC.h, [288](#)
- TransferTriggerSource_Line2
 - CameraDefsC.h, [288](#)
- TransferTriggerSource_SoftwareSignal0
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_SoftwareSignal1
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_SoftwareSignal2
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_Timer0End
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_Timer0Start
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_Timer1End
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_Timer1Start
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_Timer2End
 - CameraDefsC.h, [289](#)
- TransferTriggerSource_Timer2Start
 - CameraDefsC.h, [289](#)
- Transport Layer Enumerations, [32](#)
- TransportLayerDefsC.h
 - DeviceAccessStatus_Busy, [476](#)
 - DeviceAccessStatus_NoAccess, [476](#)
 - DeviceAccessStatus_OpenReadOnly, [476](#)
 - DeviceAccessStatus_OpenReadWrite, [476](#)
 - DeviceAccessStatus_ReadOnly, [476](#)
 - DeviceAccessStatus_ReadWrite, [476](#)
 - DeviceAccessStatus_Unknown, [476](#)
 - DeviceCurrentSpeed_FullSpeed, [476](#)
 - DeviceCurrentSpeed_HighSpeed, [476](#)
 - DeviceCurrentSpeed_LowSpeed, [476](#)
 - DeviceCurrentSpeed_SuperSpeed, [476](#)
 - DeviceCurrentSpeed_UnknownSpeed, [476](#)
 - DeviceEndiannessMechanism_Legacy, [478](#)
 - DeviceEndiannessMechanism_Standard, [478](#)
 - DeviceType_CameraLink, [478](#)
 - DeviceType_CameraLinkHS, [478](#)
 - DeviceType_CoaXPress, [478](#)
 - DeviceType_Custom, [478](#)
 - DeviceType_GigEVision, [478](#)
 - DeviceType_USB3Vision, [478](#)
 - FilterDriverStatus_Disabled, [478](#)
 - FilterDriverStatus_Enabled, [478](#)
 - FilterDriverStatus_NotSupported, [478](#)
 - GenICamXMLLocation_Device, [479](#)
 - GenICamXMLLocation_Host, [479](#)
 - GevCCP_EnumEntry_GevCCP_ControlAccess, [479](#)
 - GevCCP_EnumEntry_GevCCP_ExclusiveAccess, [479](#)
 - GevCCP_EnumEntry_GevCCP_OpenAccess, [479](#)
 - GUIXMLLocation_Device, [479](#)
 - GUIXMLLocation_Host, [479](#)
 - InterfaceType_CameraLink, [479](#)
 - InterfaceType_CameraLinkHS, [480](#)
 - InterfaceType_CoaXPress, [480](#)
 - InterfaceType_Custom, [480](#)
 - InterfaceType_GigEVision, [479](#)
 - InterfaceType_USB3Vision, [480](#)
 - NUMDEVICEACCESSSTATUS, [476](#)
 - NUMDEVICECURRENTSPEED, [476](#)
 - NUMDEVICEENDIANESSMECHANISM, [478](#)
 - NUMDEVICETYPE, [478](#)
 - NUMFILTERDRIVERSTATUS, [478](#)
 - NUMGENICAMXMLLOCATION, [479](#)
 - NUMGEVCCP, [479](#)
 - NUMGUIXMLLOCATION, [479](#)
 - NUMINTERFACETYPE, [480](#)
 - NUMPOESTATUS, [480](#)
 - NUMSTREAMBUFFERCOUNTMODE, [480](#)
 - NUMSTREAMBUFFERHANDLINGMODE, [481](#)
 - NUMSTREAMMODE, [481](#)
 - NUMSTREAMTYPE, [482](#)
 - NUMTLTYPE, [482](#)
 - POEStatus_NotSupported, [480](#)
 - POEStatus_PowerOff, [480](#)
 - POEStatus_PowerOn, [480](#)
 - spinTLDeviceAccessStatusEnums, [476](#)
 - spinTLDeviceCurrentSpeedEnums, [476](#)
 - spinTLDeviceEndiannessMechanismEnums, [476](#)
 - spinTLDeviceTypeEnums, [478](#)
 - spinTLFilterDriverStatusEnums, [478](#)
 - spinTLGenICamXMLLocationEnums, [478](#)
 - spinTLGevCCPEnums, [479](#)
 - spinTLGUIXMLLocationEnums, [479](#)
 - spinTLInterfaceTypeEnums, [479](#)
 - spinTLPOEStatusEnums, [480](#)
 - spinTLStreamBufferCountModeEnums, [480](#)

- spinTLStreamBufferHandlingModeEnums, [480](#)
- spinTLStreamModeEnums, [481](#)
- spinTLStreamTypeEnums, [481](#)
- spinTLTLTypeEnums, [482](#)
- StreamBufferCountMode_Manual, [480](#)
- StreamBufferHandlingMode_NewestFirst, [481](#)
- StreamBufferHandlingMode_NewestOnly, [481](#)
- StreamBufferHandlingMode_OldestFirst, [481](#)
- StreamBufferHandlingMode_OldestFirstOverwrite, [481](#)
- StreamMode_LWF, [481](#)
- StreamMode_MVA, [481](#)
- StreamMode_Socket, [481](#)
- StreamType_CameraLink, [482](#)
- StreamType_CameraLinkHS, [482](#)
- StreamType_CoaXPress, [482](#)
- StreamType_Custom, [482](#)
- StreamType_GigEVision, [482](#)
- StreamType_USB3Vision, [482](#)
- TLType_CameraLink, [482](#)
- TLType_CameraLinkHS, [482](#)
- TLType_CoaXPress, [482](#)
- TLType_Custom, [482](#)
- TLType_GigEVision, [482](#)
- TLType_Mixed, [482](#)
- TLType_USB3Vision, [482](#)
- TriggerActivation
 - quickSpin, [129](#)
- TriggerActivation_AnyEdge
 - CameraDefsC.h, [289](#)
- TriggerActivation_FallingEdge
 - CameraDefsC.h, [289](#)
- TriggerActivation_LevelHigh
 - CameraDefsC.h, [289](#)
- TriggerActivation_LevelLow
 - CameraDefsC.h, [289](#)
- TriggerActivation_RisingEdge
 - CameraDefsC.h, [289](#)
- TriggerDelay
 - quickSpin, [129](#)
- TriggerDivider
 - quickSpin, [129](#)
- TriggerEventTest
 - quickSpin, [130](#)
- TriggerMode
 - quickSpin, [130](#)
- TriggerMode_Off
 - CameraDefsC.h, [290](#)
- TriggerMode_On
 - CameraDefsC.h, [290](#)
- TriggerMultiplier
 - quickSpin, [130](#)
- TriggerOverlap
 - quickSpin, [130](#)
- TriggerOverlap_Off
 - CameraDefsC.h, [290](#)
- TriggerOverlap_PreviousFrame
 - CameraDefsC.h, [290](#)
- TriggerOverlap_ReadOut
 - CameraDefsC.h, [290](#)
- TriggerSelector
 - quickSpin, [130](#)
- TriggerSelector_AcquisitionStart
 - CameraDefsC.h, [290](#)
- TriggerSelector_FrameBurstStart
 - CameraDefsC.h, [290](#)
- TriggerSelector_FrameStart
 - CameraDefsC.h, [290](#)
- TriggerSoftware
 - quickSpin, [130](#)
- TriggerSource
 - quickSpin, [130](#)
- TriggerSource_Action0
 - CameraDefsC.h, [291](#)
- TriggerSource_Counter0End
 - CameraDefsC.h, [291](#)
- TriggerSource_Counter0Start
 - CameraDefsC.h, [291](#)
- TriggerSource_Counter1End
 - CameraDefsC.h, [291](#)
- TriggerSource_Counter1Start
 - CameraDefsC.h, [291](#)
- TriggerSource_Line0
 - CameraDefsC.h, [291](#)
- TriggerSource_Line1
 - CameraDefsC.h, [291](#)
- TriggerSource_Line2
 - CameraDefsC.h, [291](#)
- TriggerSource_Line3
 - CameraDefsC.h, [291](#)
- TriggerSource_LogicBlock0
 - CameraDefsC.h, [291](#)
- TriggerSource_LogicBlock1
 - CameraDefsC.h, [291](#)
- TriggerSource_Software
 - CameraDefsC.h, [291](#)
- TriggerSource_UserOutput0
 - CameraDefsC.h, [291](#)
- TriggerSource_UserOutput1
 - CameraDefsC.h, [291](#)
- TriggerSource_UserOutput2
 - CameraDefsC.h, [291](#)
- TriggerSource_UserOutput3
 - CameraDefsC.h, [291](#)
- True
 - SpinnakerDefsC.h, [418](#)
- type
 - spinLibraryVersion, [168](#)
- UNKNOWN_PIXELFORMAT
 - CameraDefsC.h, [264](#)
- UnknownNode
 - SpinnakerGenApiDefsC.h, [468](#)
- Unsigned
 - SpinnakerGenApiDefsC.h, [468](#)
- USB
 - SpinnakerGenApiDefsC.h, [469](#)

- UserOutputSelector
 - quickSpin, [130](#)
- UserOutputSelector_UserOutput0
 - CameraDefsC.h, [291](#)
- UserOutputSelector_UserOutput1
 - CameraDefsC.h, [291](#)
- UserOutputSelector_UserOutput2
 - CameraDefsC.h, [291](#)
- UserOutputSelector_UserOutput3
 - CameraDefsC.h, [291](#)
- UserOutputValue
 - quickSpin, [131](#)
- UserOutputValueAll
 - quickSpin, [131](#)
- UserOutputValueAllMask
 - quickSpin, [131](#)
- UserSetDefault
 - quickSpin, [131](#)
- UserSetDefault_Default
 - CameraDefsC.h, [292](#)
- UserSetDefault_UserSet0
 - CameraDefsC.h, [292](#)
- UserSetDefault_UserSet1
 - CameraDefsC.h, [292](#)
- UserSetFeatureEnable
 - quickSpin, [131](#)
- UserSetLoad
 - quickSpin, [131](#)
- UserSetSave
 - quickSpin, [131](#)
- UserSetSelector
 - quickSpin, [131](#)
- UserSetSelector_Default
 - CameraDefsC.h, [292](#)
- UserSetSelector_UserSet0
 - CameraDefsC.h, [292](#)
- UserSetSelector_UserSet1
 - CameraDefsC.h, [292](#)
- V3_3Enable
 - quickSpin, [132](#)
- ValueNode
 - SpinnakerGenApiDefsC.h, [467](#)
- Varying
 - SpinnakerGenApiDefsC.h, [469](#)
- WhiteClip
 - quickSpin, [132](#)
- WhiteClipSelector
 - quickSpin, [132](#)
- WhiteClipSelector_All
 - CameraDefsC.h, [292](#)
- WhiteClipSelector_Blue
 - CameraDefsC.h, [292](#)
- WhiteClipSelector_Green
 - CameraDefsC.h, [292](#)
- WhiteClipSelector_Red
 - CameraDefsC.h, [292](#)
- WhiteClipSelector_Tap1
 - CameraDefsC.h, [292](#)
- WhiteClipSelector_Tap2
 - CameraDefsC.h, [292](#)
- WhiteClipSelector_U
 - CameraDefsC.h, [292](#)
- WhiteClipSelector_V
 - CameraDefsC.h, [292](#)
- WhiteClipSelector_Y
 - CameraDefsC.h, [292](#)
- Width
 - quickSpin, [132](#)
- width
 - spinAVIOption, [157](#)
 - spinH264Option, [165](#)
 - spinMJPEGOption, [170](#)
- WidthMax
 - quickSpin, [132](#)
- WO
 - SpinnakerGenApiDefsC.h, [464](#)
- WriteAround
 - SpinnakerGenApiDefsC.h, [464](#)
- WriteThrough
 - SpinnakerGenApiDefsC.h, [464](#)
- xvAll
 - SpinnakerGenApiDefsC.h, [470](#)
- xvCycles
 - SpinnakerGenApiDefsC.h, [470](#)
- xvDefault
 - SpinnakerGenApiDefsC.h, [470](#)
- xvLoad
 - SpinnakerGenApiDefsC.h, [470](#)
- xvSFNC
 - SpinnakerGenApiDefsC.h, [470](#)
- Yes
 - SpinnakerGenApiDefsC.h, [470](#)