

Spinnaker C

1.23.0.27

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Software Licensing Information</b>	<b>3</b>
<b>3</b>	<b>Module Index</b>	<b>5</b>
3.1	Modules . . . . .	5
<b>4</b>	<b>Data Structure Index</b>	<b>7</b>
4.1	Data Structures . . . . .	7
<b>5</b>	<b>File Index</b>	<b>9</b>
5.1	File List . . . . .	9
<b>6</b>	<b>Module Documentation</b>	<b>11</b>
6.1	Spinnaker C Definitions . . . . .	11
6.1.1	Detailed Description . . . . .	12
6.1.2	Typedef Documentation . . . . .	12
6.1.2.1	bool8_t . . . . .	12
6.1.3	Variable Documentation . . . . .	12
6.1.3.1	False . . . . .	12
6.1.3.2	True . . . . .	12
6.2	Camera Enumerations . . . . .	13
6.2.1	Detailed Description . . . . .	45
6.2.2	Enumeration Type Documentation . . . . .	45
6.2.2.1	spinAcquisitionModeEnums . . . . .	45
6.2.2.2	spinAcquisitionStatusSelectorEnums . . . . .	45

6.2.2.3	<a href="#">spinActionUnconditionalModeEnums</a>	46
6.2.2.4	<a href="#">spinAdcBitDepthEnums</a>	46
6.2.2.5	<a href="#">spinAutoAlgorithmSelectorEnums</a>	46
6.2.2.6	<a href="#">spinAutoExposureControlPriorityEnums</a>	47
6.2.2.7	<a href="#">spinAutoExposureLightingModeEnums</a>	47
6.2.2.8	<a href="#">spinAutoExposureMeteringModeEnums</a>	47
6.2.2.9	<a href="#">spinAutoExposureTargetGreyValueAutoEnums</a>	48
6.2.2.10	<a href="#">spinBalanceRatioSelectorEnums</a>	48
6.2.2.11	<a href="#">spinBalanceWhiteAutoEnums</a>	49
6.2.2.12	<a href="#">spinBalanceWhiteAutoProfileEnums</a>	49
6.2.2.13	<a href="#">spinBinningHorizontalModeEnums</a>	49
6.2.2.14	<a href="#">spinBinningSelectorEnums</a>	50
6.2.2.15	<a href="#">spinBinningVerticalModeEnums</a>	50
6.2.2.16	<a href="#">spinBlackLevelAutoBalanceEnums</a>	50
6.2.2.17	<a href="#">spinBlackLevelAutoEnums</a>	51
6.2.2.18	<a href="#">spinBlackLevelSelectorEnums</a>	51
6.2.2.19	<a href="#">spinChunkBlackLevelSelectorEnums</a>	51
6.2.2.20	<a href="#">spinChunkCounterSelectorEnums</a>	52
6.2.2.21	<a href="#">spinChunkEncoderSelectorEnums</a>	52
6.2.2.22	<a href="#">spinChunkEncoderStatusEnums</a>	52
6.2.2.23	<a href="#">spinChunkExposureTimeSelectorEnums</a>	52
6.2.2.24	<a href="#">spinChunkGainSelectorEnums</a>	53
6.2.2.25	<a href="#">spinChunkImageComponentEnums</a>	53
6.2.2.26	<a href="#">spinChunkPixelFormatEnums</a>	54
6.2.2.27	<a href="#">spinChunkRegionIDEnums</a>	54
6.2.2.28	<a href="#">spinChunkScan3dCoordinateReferenceSelectorEnums</a>	55
6.2.2.29	<a href="#">spinChunkScan3dCoordinateSelectorEnums</a>	55
6.2.2.30	<a href="#">spinChunkScan3dCoordinateSystemEnums</a>	55
6.2.2.31	<a href="#">spinChunkScan3dCoordinateSystemReferenceEnums</a>	56
6.2.2.32	<a href="#">spinChunkScan3dCoordinateTransformSelectorEnums</a>	56

6.2.2.33	<a href="#">spinChunkScan3dDistanceUnitEnums</a>	56
6.2.2.34	<a href="#">spinChunkScan3dOutputModeEnums</a>	57
6.2.2.35	<a href="#">spinChunkSelectorEnums</a>	58
6.2.2.36	<a href="#">spinChunkSourceIDEnums</a>	58
6.2.2.37	<a href="#">spinChunkTimerSelectorEnums</a>	58
6.2.2.38	<a href="#">spinChunkTransferStreamIDEnums</a>	59
6.2.2.39	<a href="#">spinCIConfigurationEnums</a>	59
6.2.2.40	<a href="#">spinCITimeSlotsCountEnums</a>	60
6.2.2.41	<a href="#">spinColorTransformationSelectorEnums</a>	60
6.2.2.42	<a href="#">spinColorTransformationValueSelectorEnums</a>	60
6.2.2.43	<a href="#">spinCounterEventActivationEnums</a>	61
6.2.2.44	<a href="#">spinCounterEventSourceEnums</a>	61
6.2.2.45	<a href="#">spinCounterResetActivationEnums</a>	62
6.2.2.46	<a href="#">spinCounterResetSourceEnums</a>	62
6.2.2.47	<a href="#">spinCounterSelectorEnums</a>	63
6.2.2.48	<a href="#">spinCounterStatusEnums</a>	63
6.2.2.49	<a href="#">spinCounterTriggerActivationEnums</a>	63
6.2.2.50	<a href="#">spinCounterTriggerSourceEnums</a>	64
6.2.2.51	<a href="#">spinCxpConnectionTestModeEnums</a>	64
6.2.2.52	<a href="#">spinCxpLinkConfigurationEnums</a>	64
6.2.2.53	<a href="#">spinCxpLinkConfigurationPreferredEnums</a>	65
6.2.2.54	<a href="#">spinCxpLinkConfigurationStatusEnums</a>	66
6.2.2.55	<a href="#">spinCxpPoCxpStatusEnums</a>	67
6.2.2.56	<a href="#">spinDecimationHorizontalModeEnums</a>	68
6.2.2.57	<a href="#">spinDecimationSelectorEnums</a>	68
6.2.2.58	<a href="#">spinDecimationVerticalModeEnums</a>	68
6.2.2.59	<a href="#">spinDefectCorrectionModeEnums</a>	69
6.2.2.60	<a href="#">spinDeinterlacingEnums</a>	69
6.2.2.61	<a href="#">spinDeviceCharacterSetEnums</a>	69
6.2.2.62	<a href="#">spinDeviceClockSelectorEnums</a>	70

6.2.2.63	<a href="#">spinDeviceConnectionStatusEnums</a>	70
6.2.2.64	<a href="#">spinDeviceIndicatorModeEnums</a>	70
6.2.2.65	<a href="#">spinDeviceLinkHeartbeatModeEnums</a>	70
6.2.2.66	<a href="#">spinDeviceLinkThroughputLimitModeEnums</a>	72
6.2.2.67	<a href="#">spinDevicePowerSupplySelectorEnums</a>	72
6.2.2.68	<a href="#">spinDeviceRegistersEndiannessEnums</a>	72
6.2.2.69	<a href="#">spinDeviceScanTypeEnums</a>	73
6.2.2.70	<a href="#">spinDeviceSerialPortBaudRateEnums</a>	73
6.2.2.71	<a href="#">spinDeviceSerialPortSelectorEnums</a>	73
6.2.2.72	<a href="#">spinDeviceStreamChannelEndiannessEnums</a>	74
6.2.2.73	<a href="#">spinDeviceStreamChannelTypeEnums</a>	74
6.2.2.74	<a href="#">spinDeviceTapGeometryEnums</a>	74
6.2.2.75	<a href="#">spinDeviceTemperatureSelectorEnums</a>	75
6.2.2.76	<a href="#">spinDeviceTLTypeEnums</a>	76
6.2.2.77	<a href="#">spinDeviceTypeEnums</a>	76
6.2.2.78	<a href="#">spinEncoderModeEnums</a>	76
6.2.2.79	<a href="#">spinEncoderOutputModeEnums</a>	77
6.2.2.80	<a href="#">spinEncoderResetActivationEnums</a>	77
6.2.2.81	<a href="#">spinEncoderResetSourceEnums</a>	78
6.2.2.82	<a href="#">spinEncoderSelectorEnums</a>	79
6.2.2.83	<a href="#">spinEncoderSourceAEnums</a>	79
6.2.2.84	<a href="#">spinEncoderSourceBEnums</a>	79
6.2.2.85	<a href="#">spinEncoderStatusEnums</a>	80
6.2.2.86	<a href="#">spinEventNotificationEnums</a>	80
6.2.2.87	<a href="#">spinEventSelectorEnums</a>	80
6.2.2.88	<a href="#">spinExposureActiveModeEnums</a>	81
6.2.2.89	<a href="#">spinExposureAutoEnums</a>	81
6.2.2.90	<a href="#">spinExposureModeEnums</a>	81
6.2.2.91	<a href="#">spinExposureTimeModeEnums</a>	82
6.2.2.92	<a href="#">spinExposureTimeSelectorEnums</a>	82

6.2.2.93	<a href="#">spinFileOpenModeEnums</a>	83
6.2.2.94	<a href="#">spinFileOperationSelectorEnums</a>	83
6.2.2.95	<a href="#">spinFileOperationStatusEnums</a>	83
6.2.2.96	<a href="#">spinFileSelectorEnums</a>	84
6.2.2.97	<a href="#">spinGainAutoBalanceEnums</a>	84
6.2.2.98	<a href="#">spinGainAutoEnums</a>	84
6.2.2.99	<a href="#">spinGainSelectorEnums</a>	85
6.2.2.100	<a href="#">spinGevCCPEnums</a>	85
6.2.2.101	<a href="#">spinGevCurrentPhysicalLinkConfigurationEnums</a>	85
6.2.2.102	<a href="#">spinGevGVCPExtendedStatusCodesSelectorEnums</a>	86
6.2.2.103	<a href="#">spinGevGVSPExtendedIDModeEnums</a>	86
6.2.2.104	<a href="#">spinGevIEEE1588ClockAccuracyEnums</a>	86
6.2.2.105	<a href="#">spinGevIEEE1588ModeEnums</a>	86
6.2.2.106	<a href="#">spinGevIEEE1588StatusEnums</a>	87
6.2.2.107	<a href="#">spinGevIPConfigurationStatusEnums</a>	87
6.2.2.108	<a href="#">spinGevPhysicalLinkConfigurationEnums</a>	88
6.2.2.109	<a href="#">spinGevSupportedOptionSelectorEnums</a>	88
6.2.2.110	<a href="#">spinImageComponentSelectorEnums</a>	89
6.2.2.111	<a href="#">spinImageCompressionJPEGFormatOptionEnums</a>	89
6.2.2.112	<a href="#">spinImageCompressionModeEnums</a>	90
6.2.2.113	<a href="#">spinImageCompressionRateOptionEnums</a>	90
6.2.2.114	<a href="#">spinLineFormatEnums</a>	90
6.2.2.115	<a href="#">spinLineInputFilterSelectorEnums</a>	91
6.2.2.116	<a href="#">spinLineModeEnums</a>	91
6.2.2.117	<a href="#">spinLineSelectorEnums</a>	91
6.2.2.118	<a href="#">spinLineSourceEnums</a>	92
6.2.2.119	<a href="#">spinLogicBlockLUTInputActivationEnums</a>	92
6.2.2.120	<a href="#">spinLogicBlockLUTInputSelectorEnums</a>	93
6.2.2.121	<a href="#">spinLogicBlockLUTInputSourceEnums</a>	93
6.2.2.122	<a href="#">spinLogicBlockLUTSelectorEnums</a>	94

6.2.2.123 spinLogicBlockSelectorEnums . . . . .	94
6.2.2.124 spinLUTSelectorEnums . . . . .	94
6.2.2.125 spinPixelColorFilterEnums . . . . .	95
6.2.2.126 spinPixelFormatEnums . . . . .	95
6.2.2.127 spinPixelFormatInfoSelectorEnums . . . . .	101
6.2.2.128 spinPixelSizeEnums . . . . .	106
6.2.2.129 spinRegionDestinationEnums . . . . .	107
6.2.2.130 spinRegionModeEnums . . . . .	107
6.2.2.131 spinRegionSelectorEnums . . . . .	108
6.2.2.132 spinRgbTransformLightSourceEnums . . . . .	108
6.2.2.133 spinScan3dCoordinateReferenceSelectorEnums . . . . .	109
6.2.2.134 spinScan3dCoordinateSelectorEnums . . . . .	109
6.2.2.135 spinScan3dCoordinateSystemEnums . . . . .	109
6.2.2.136 spinScan3dCoordinateSystemReferenceEnums . . . . .	110
6.2.2.137 spinScan3dCoordinateTransformSelectorEnums . . . . .	110
6.2.2.138 spinScan3dDistanceUnitEnums . . . . .	110
6.2.2.139 spinScan3dOutputModeEnums . . . . .	111
6.2.2.140 spinSensorDigitizationTapsEnums . . . . .	111
6.2.2.141 spinSensorShutterModeEnums . . . . .	112
6.2.2.142 spinSensorTapsEnums . . . . .	112
6.2.2.143 spinSequencerConfigurationModeEnums . . . . .	113
6.2.2.144 spinSequencerConfigurationValidEnums . . . . .	113
6.2.2.145 spinSequencerModeEnums . . . . .	113
6.2.2.146 spinSequencerSetValidEnums . . . . .	113
6.2.2.147 spinSequencerTriggerActivationEnums . . . . .	114
6.2.2.148 spinSequencerTriggerSourceEnums . . . . .	114
6.2.2.149 spinSerialPortBaudRateEnums . . . . .	114
6.2.2.150 spinSerialPortParityEnums . . . . .	115
6.2.2.151 spinSerialPortSelectorEnums . . . . .	115
6.2.2.152 spinSerialPortSourceEnums . . . . .	116



6.2.2.153 spinSerialPortStopBitsEnums . . . . .	116
6.2.2.154 spinSoftwareSignalSelectorEnums . . . . .	116
6.2.2.155 spinSourceSelectorEnums . . . . .	117
6.2.2.156 spinTestPatternEnums . . . . .	117
6.2.2.157 spinTestPatternGeneratorSelectorEnums . . . . .	117
6.2.2.158 spinTimerSelectorEnums . . . . .	118
6.2.2.159 spinTimerStatusEnums . . . . .	118
6.2.2.160 spinTimerTriggerActivationEnums . . . . .	118
6.2.2.161 spinTimerTriggerSourceEnums . . . . .	119
6.2.2.162 spinTransferComponentSelectorEnums . . . . .	120
6.2.2.163 spinTransferControlModeEnums . . . . .	120
6.2.2.164 spinTransferOperationModeEnums . . . . .	121
6.2.2.165 spinTransferQueueModeEnums . . . . .	121
6.2.2.166 spinTransferSelectorEnums . . . . .	121
6.2.2.167 spinTransferStatusSelectorEnums . . . . .	122
6.2.2.168 spinTransferTriggerActivationEnums . . . . .	122
6.2.2.169 spinTransferTriggerModeEnums . . . . .	122
6.2.2.170 spinTransferTriggerSelectorEnums . . . . .	123
6.2.2.171 spinTransferTriggerSourceEnums . . . . .	123
6.2.2.172 spinTriggerActivationEnums . . . . .	124
6.2.2.173 spinTriggerModeEnums . . . . .	125
6.2.2.174 spinTriggerOverlapEnums . . . . .	125
6.2.2.175 spinTriggerSelectorEnums . . . . .	125
6.2.2.176 spinTriggerSourceEnums . . . . .	126
6.2.2.177 spinUserOutputSelectorEnums . . . . .	126
6.2.2.178 spinUserSetDefaultEnums . . . . .	126
6.2.2.179 spinUserSetSelectorEnums . . . . .	127
6.2.2.180 spinWhiteClipSelectorEnums . . . . .	127
6.3 Chunk Data Structures . . . . .	128
6.3.1 Detailed Description . . . . .	128

6.4	Spinnaker C QuickSpin API	129
6.4.1	Detailed Description	129
6.5	QuickSpin Access	130
6.5.1	Detailed Description	130
6.5.2	Function Documentation	130
6.5.2.1	quickSpinInit()	130
6.5.2.2	quickSpinInitEx()	131
6.5.2.3	quickSpinTLDeviceInit()	131
6.5.2.4	quickSpinTLInterfaceInit()	131
6.5.2.5	quickSpinTLStreamInit()	131
6.5.2.6	quickSpinTLSystemInit()	131
6.6	Spinnaker C API	132
6.6.1	Detailed Description	133
6.6.2	Function Documentation	133
6.6.2.1	spinCameraDiscoverMaxPacketSize()	133
6.7	Error Handling	134
6.7.1	Detailed Description	134
6.7.2	Function Documentation	134
6.7.2.1	spinErrorGetLast()	134
6.7.2.2	spinErrorGetLastBuildDate()	135
6.7.2.3	spinErrorGetLastBuildTime()	135
6.7.2.4	spinErrorGetLastFileName()	136
6.7.2.5	spinErrorGetLastFullMessage()	136
6.7.2.6	spinErrorGetLastFunctionName()	137
6.7.2.7	spinErrorGetLastLineNumber()	137
6.7.2.8	spinErrorGetLastMessage()	138
6.8	System Access	139
6.8.1	Detailed Description	140
6.8.2	Function Documentation	140
6.8.2.1	spinSystemGetCameras()	140

6.8.2.2	<a href="#">spinSystemGetCamerasEx()</a>	141
6.8.2.3	<a href="#">spinSystemGetInstance()</a>	141
6.8.2.4	<a href="#">spinSystemGetInterfaces()</a>	143
6.8.2.5	<a href="#">spinSystemGetLibraryVersion()</a>	143
6.8.2.6	<a href="#">spinSystemGetLoggingLevel()</a>	144
6.8.2.7	<a href="#">spinSystemGetTLNodeMap()</a>	144
6.8.2.8	<a href="#">spinSystemIsInUse()</a>	145
6.8.2.9	<a href="#">spinSystemRegisterArrivalEvent()</a>	145
6.8.2.10	<a href="#">spinSystemRegisterInterfaceEvent()</a>	146
6.8.2.11	<a href="#">spinSystemRegisterLogEvent()</a>	146
6.8.2.12	<a href="#">spinSystemRegisterRemovalEvent()</a>	147
6.8.2.13	<a href="#">spinSystemReleaseInstance()</a>	147
6.8.2.14	<a href="#">spinSystemSendActionCommand()</a>	148
6.8.2.15	<a href="#">spinSystemSetLoggingLevel()</a>	148
6.8.2.16	<a href="#">spinSystemUnregisterAllLogEvents()</a>	149
6.8.2.17	<a href="#">spinSystemUnregisterArrivalEvent()</a>	149
6.8.2.18	<a href="#">spinSystemUnregisterInterfaceEvent()</a>	150
6.8.2.19	<a href="#">spinSystemUnregisterLogEvent()</a>	150
6.8.2.20	<a href="#">spinSystemUnregisterRemovalEvent()</a>	151
6.8.2.21	<a href="#">spinSystemUpdateCameras()</a>	151
6.8.2.22	<a href="#">spinSystemUpdateCamerasEx()</a>	152
6.9	<a href="#">InterfaceList Access</a>	153
6.9.1	<a href="#">Detailed Description</a>	153
6.9.2	<a href="#">Function Documentation</a>	153
6.9.2.1	<a href="#">spinInterfaceListClear()</a>	153
6.9.2.2	<a href="#">spinInterfaceListCreateEmpty()</a>	154
6.9.2.3	<a href="#">spinInterfaceListDestroy()</a>	154
6.9.2.4	<a href="#">spinInterfaceListGet()</a>	155
6.9.2.5	<a href="#">spinInterfaceListGetSize()</a>	155
6.10	<a href="#">CameraList Access</a>	157

6.10.1 Detailed Description . . . . .	157
6.10.2 Function Documentation . . . . .	157
6.10.2.1 spinCameraListAppend() . . . . .	158
6.10.2.2 spinCameraListClear() . . . . .	158
6.10.2.3 spinCameraListCreateEmpty() . . . . .	158
6.10.2.4 spinCameraListDestroy() . . . . .	159
6.10.2.5 spinCameraListGet() . . . . .	159
6.10.2.6 spinCameraListGetBySerial() . . . . .	160
6.10.2.7 spinCameraListGetSize() . . . . .	160
6.10.2.8 spinCameraListRemove() . . . . .	161
6.10.2.9 spinCameraListRemoveBySerial() . . . . .	161
6.11 Interface Access . . . . .	163
6.11.1 Detailed Description . . . . .	164
6.11.2 Function Documentation . . . . .	164
6.11.2.1 spinInterfaceGetCameras() . . . . .	164
6.11.2.2 spinInterfaceGetCamerasEx() . . . . .	164
6.11.2.3 spinInterfaceGetTLNodeMap() . . . . .	165
6.11.2.4 spinInterfaceIsInUse() . . . . .	165
6.11.2.5 spinInterfaceRegisterArrivalEvent() . . . . .	166
6.11.2.6 spinInterfaceRegisterInterfaceEvent() . . . . .	166
6.11.2.7 spinInterfaceRegisterRemovalEvent() . . . . .	167
6.11.2.8 spinInterfaceRelease() . . . . .	167
6.11.2.9 spinInterfaceSendActionCommand() . . . . .	168
6.11.2.10 spinInterfaceUnregisterArrivalEvent() . . . . .	168
6.11.2.11 spinInterfaceUnregisterInterfaceEvent() . . . . .	169
6.11.2.12 spinInterfaceUnregisterRemovalEvent() . . . . .	169
6.11.2.13 spinInterfaceUpdateCameras() . . . . .	170
6.12 Camera Access . . . . .	171
6.12.1 Detailed Description . . . . .	172
6.12.2 Function Documentation . . . . .	172

6.12.2.1	<a href="#">spinCameraBeginAcquisition()</a>	172
6.12.2.2	<a href="#">spinCameraDeInit()</a>	173
6.12.2.3	<a href="#">spinCameraEndAcquisition()</a>	173
6.12.2.4	<a href="#">spinCameraGetAccessMode()</a>	174
6.12.2.5	<a href="#">spinCameraGetGuiXml()</a>	174
6.12.2.6	<a href="#">spinCameraGetNextImage()</a>	175
6.12.2.7	<a href="#">spinCameraGetNextImageEx()</a>	175
6.12.2.8	<a href="#">spinCameraGetNodeMap()</a>	176
6.12.2.9	<a href="#">spinCameraGetTLDeviceNodeMap()</a>	176
6.12.2.10	<a href="#">spinCameraGetTLStreamNodeMap()</a>	177
6.12.2.11	<a href="#">spinCameraGetUniqueID()</a>	177
6.12.2.12	<a href="#">spinCameraInit()</a>	178
6.12.2.13	<a href="#">spinCameraIsInitialized()</a>	178
6.12.2.14	<a href="#">spinCameraIsStreaming()</a>	178
6.12.2.15	<a href="#">spinCameraIsValid()</a>	179
6.12.2.16	<a href="#">spinCameraReadPort()</a>	179
6.12.2.17	<a href="#">spinCameraRegisterDeviceEvent()</a>	180
6.12.2.18	<a href="#">spinCameraRegisterDeviceEventEx()</a>	180
6.12.2.19	<a href="#">spinCameraRegisterImageEvent()</a>	181
6.12.2.20	<a href="#">spinCameraRelease()</a>	181
6.12.2.21	<a href="#">spinCameraUnregisterDeviceEvent()</a>	181
6.12.2.22	<a href="#">spinCameraUnregisterImageEvent()</a>	182
6.12.2.23	<a href="#">spinCameraWritePort()</a>	182
6.13	<a href="#">Image Access</a>	183
6.13.1	<a href="#">Detailed Description</a>	185
6.13.2	<a href="#">Function Documentation</a>	185
6.13.2.1	<a href="#">spinImageCalculateStatistics()</a>	185
6.13.2.2	<a href="#">spinImageCheckCRC()</a>	186
6.13.2.3	<a href="#">spinImageConvert()</a>	186
6.13.2.4	<a href="#">spinImageConvertEx()</a>	187

6.13.2.5 spinImageCreate()	187
6.13.2.6 spinImageCreateEmpty()	188
6.13.2.7 spinImageCreateEx()	188
6.13.2.8 spinImageDeepCopy()	189
6.13.2.9 spinImageDestroy()	189
6.13.2.10 spinImageGetBitsPerPixel()	190
6.13.2.11 spinImageGetBufferSize()	190
6.13.2.12 spinImageGetChunkLayoutID()	191
6.13.2.13 spinImageGetColorProcessing()	191
6.13.2.14 spinImageGetData()	192
6.13.2.15 spinImageGetDefaultColorProcessing()	192
6.13.2.16 spinImageGetFrameID()	193
6.13.2.17 spinImageGetHeight()	193
6.13.2.18 spinImageGetID()	194
6.13.2.19 spinImageGetOffsetX()	194
6.13.2.20 spinImageGetOffsetY()	195
6.13.2.21 spinImageGetPaddingX()	195
6.13.2.22 spinImageGetPaddingY()	196
6.13.2.23 spinImageGetPayloadType()	196
6.13.2.24 spinImageGetPixelFormat()	197
6.13.2.25 spinImageGetPixelFormatName()	197
6.13.2.26 spinImageGetPrivateData()	198
6.13.2.27 spinImageGetSize()	198
6.13.2.28 spinImageGetStatus()	199
6.13.2.29 spinImageGetStatusDescription()	199
6.13.2.30 spinImageGetStride()	200
6.13.2.31 spinImageGetTimeStamp()	200
6.13.2.32 spinImageGetTLPayloadType()	201
6.13.2.33 spinImageGetTLPixelFormat()	201
6.13.2.34 spinImageGetTLPixelFormatNamespace()	202

6.13.2.35 spinImageGetValidPayloadSize()	202
6.13.2.36 spinImageGetWidth()	203
6.13.2.37 spinImageHasCRC()	203
6.13.2.38 spinImageIsIncomplete()	204
6.13.2.39 spinImageRelease()	204
6.13.2.40 spinImageReset()	204
6.13.2.41 spinImageResetEx()	205
6.13.2.42 spinImageSave()	206
6.13.2.43 spinImageSaveBmp()	206
6.13.2.44 spinImageSaveFromExt()	207
6.13.2.45 spinImageSaveJpeg()	207
6.13.2.46 spinImageSaveJpg2()	208
6.13.2.47 spinImageSavePgm()	208
6.13.2.48 spinImageSavePng()	209
6.13.2.49 spinImageSavePpm()	209
6.13.2.50 spinImageSaveTiff()	210
6.13.2.51 spinImageSetDefaultColorProcessing()	210
6.14 Event Access	212
6.14.1 Detailed Description	212
6.14.2 Function Documentation	212
6.14.2.1 spinArrivalEventCreate()	213
6.14.2.2 spinArrivalEventDestroy()	213
6.14.2.3 spinDeviceEventCreate()	214
6.14.2.4 spinDeviceEventDestroy()	214
6.14.2.5 spinImageEventCreate()	215
6.14.2.6 spinImageEventDestroy()	215
6.14.2.7 spinInterfaceEventCreate()	216
6.14.2.8 spinInterfaceEventDestroy()	216
6.14.2.9 spinLogEventCreate()	217
6.14.2.10 spinLogEventDestroy()	217

6.14.2.11 spinRemovalEventCreate()	218
6.14.2.12 spinRemovalEventDestroy()	218
6.15 ImageStatistics Access	219
6.15.1 Detailed Description	219
6.15.2 Function Documentation	220
6.15.2.1 spinImageStatisticsCreate()	220
6.15.2.2 spinImageStatisticsDestroy()	220
6.15.2.3 spinImageStatisticsDisableAll()	220
6.15.2.4 spinImageStatisticsEnableAll()	221
6.15.2.5 spinImageStatisticsEnableGreyOnly()	221
6.15.2.6 spinImageStatisticsEnableHslOnly()	222
6.15.2.7 spinImageStatisticsEnableRgbOnly()	222
6.15.2.8 spinImageStatisticsGetAll()	223
6.15.2.9 spinImageStatisticsGetChannelStatus()	223
6.15.2.10 spinImageStatisticsGetHistogram()	224
6.15.2.11 spinImageStatisticsGetMean()	224
6.15.2.12 spinImageStatisticsGetNumPixelValues()	225
6.15.2.13 spinImageStatisticsGetPixelValueRange()	225
6.15.2.14 spinImageStatisticsGetRange()	226
6.15.2.15 spinImageStatisticsSetChannelStatus()	226
6.16 Logging Event Data Access	228
6.16.1 Detailed Description	228
6.16.2 Function Documentation	228
6.16.2.1 spinLogDataGetCategoryName()	228
6.16.2.2 spinLogDataGetLogMessage()	229
6.16.2.3 spinLogDataGetNDC()	229
6.16.2.4 spinLogDataGetPriority()	230
6.16.2.5 spinLogDataGetPriorityName()	230
6.16.2.6 spinLogDataGetThreadName()	231
6.16.2.7 spinLogDataGetTimestamp()	231



6.17 Device Event Data Access . . . . .	233
6.17.1 Detailed Description . . . . .	233
6.17.2 Function Documentation . . . . .	233
6.17.2.1 spinDeviceEventGetId() . . . . .	233
6.17.2.2 spinDeviceEventGetName() . . . . .	234
6.17.2.3 spinDeviceEventGetPayloadData() . . . . .	234
6.17.2.4 spinDeviceEventGetPayloadDataSize() . . . . .	235
6.18 AVIRecorder Access . . . . .	236
6.18.1 Detailed Description . . . . .	236
6.18.2 Function Documentation . . . . .	236
6.18.2.1 SPINNAKERC_API_DEPRECATED() [1/6] . . . . .	236
6.18.2.2 SPINNAKERC_API_DEPRECATED() [2/6] . . . . .	237
6.18.2.3 SPINNAKERC_API_DEPRECATED() [3/6] . . . . .	237
6.18.2.4 SPINNAKERC_API_DEPRECATED() [4/6] . . . . .	237
6.18.2.5 SPINNAKERC_API_DEPRECATED() [5/6] . . . . .	237
6.18.2.6 SPINNAKERC_API_DEPRECATED() [6/6] . . . . .	238
6.19 Chunk data access . . . . .	239
6.19.1 Detailed Description . . . . .	239
6.19.2 Function Documentation . . . . .	239
6.19.2.1 spinImageChunkDataGetFloatValue() . . . . .	239
6.19.2.2 spinImageChunkDataGetIntValue() . . . . .	239
6.20 Spinnaker C Handles . . . . .	240
6.20.1 Detailed Description . . . . .	241
6.20.2 Typedef Documentation . . . . .	241
6.20.2.1 spinArrivalEvent . . . . .	241
6.20.2.2 spinAVIRecorder . . . . .	241
6.20.2.3 spinCamera . . . . .	241
6.20.2.4 spinCameraList . . . . .	241
6.20.2.5 spinDeviceEvent . . . . .	242
6.20.2.6 spinDeviceEventData . . . . .	242

6.20.2.7	<a href="#">spinImage</a>	242
6.20.2.8	<a href="#">spinImageEvent</a>	242
6.20.2.9	<a href="#">spinImageStatistics</a>	242
6.20.2.10	<a href="#">spinInterface</a>	242
6.20.2.11	<a href="#">spinInterfaceEvent</a>	243
6.20.2.12	<a href="#">spinInterfaceList</a>	243
6.20.2.13	<a href="#">spinLogEvent</a>	243
6.20.2.14	<a href="#">spinLogEventData</a>	243
6.20.2.15	<a href="#">spinRemovalEvent</a>	243
6.20.2.16	<a href="#">spinSystem</a>	243
6.20.2.17	<a href="#">spinVideo</a>	243
6.21	<a href="#">Spinnaker C Function Signatures</a>	244
6.21.1	<a href="#">Detailed Description</a>	244
6.21.2	<a href="#">Typedef Documentation</a>	244
6.21.2.1	<a href="#">spinArrivalEventFunction</a>	244
6.21.2.2	<a href="#">spinDeviceEventFunction</a>	244
6.21.2.3	<a href="#">spinImageEventFunction</a>	245
6.21.2.4	<a href="#">spinLogEventFunction</a>	245
6.21.2.5	<a href="#">spinRemovalEventFunction</a>	245
6.22	<a href="#">Spinnaker C Enumerations</a>	246
6.22.1	<a href="#">Detailed Description</a>	248
6.22.2	<a href="#">Enumeration Type Documentation</a>	248
6.22.2.1	<a href="#">spinColorProcessingAlgorithm</a>	248
6.22.2.2	<a href="#">spinError</a>	249
6.22.2.3	<a href="#">spinImageFileFormat</a>	250
6.22.2.4	<a href="#">spinImageStatus</a>	251
6.22.2.5	<a href="#">spinnakerLogLevel</a>	251
6.22.2.6	<a href="#">spinPayloadTypeInfoIDs</a>	252
6.22.2.7	<a href="#">spinPixelFormatNamespaceID</a>	252
6.22.2.8	<a href="#">spinStatisticsChannel</a>	253

6.23 Spinnaker C Structures . . . . .	254
6.23.1 Detailed Description . . . . .	255
6.23.2 Enumeration Type Documentation . . . . .	255
6.23.2.1 actionCommandStatus . . . . .	255
6.23.2.2 spinCompressionMethod . . . . .	255
6.24 Spinnaker C GenICam API . . . . .	256
6.24.1 Detailed Description . . . . .	257
6.25 Node Map Access . . . . .	258
6.25.1 Detailed Description . . . . .	258
6.25.2 Function Documentation . . . . .	258
6.25.2.1 spinNodeMapGetNode() . . . . .	258
6.25.2.2 spinNodeMapGetNodeByIndex() . . . . .	259
6.25.2.3 spinNodeMapGetNumNodes() . . . . .	259
6.25.2.4 spinNodeMapPoll() . . . . .	260
6.26 Node Access . . . . .	261
6.26.1 Detailed Description . . . . .	262
6.26.2 Function Documentation . . . . .	262
6.26.2.1 spinNodeDeregisterCallback() . . . . .	262
6.26.2.2 spinNodeGetAccessMode() . . . . .	263
6.26.2.3 spinNodeGetCachingMode() . . . . .	263
6.26.2.4 spinNodeGetDescription() . . . . .	264
6.26.2.5 spinNodeGetDisplayName() . . . . .	264
6.26.2.6 spinNodeGetImposedAccessMode() . . . . .	265
6.26.2.7 spinNodeGetImposedVisibility() . . . . .	265
6.26.2.8 spinNodeGetName() . . . . .	266
6.26.2.9 spinNodeGetNameSpace() . . . . .	266
6.26.2.10 spinNodeGetPollingTime() . . . . .	267
6.26.2.11 spinNodeGetToolTip() . . . . .	267
6.26.2.12 spinNodeGetType() . . . . .	268
6.26.2.13 spinNodeGetVisibility() . . . . .	268

6.26.2.14	<code>spinNodeInvalidateNode()</code>	269
6.26.2.15	<code>spinNodesAvailable()</code>	269
6.26.2.16	<code>spinNodesEqual()</code>	269
6.26.2.17	<code>spinNodesImplemented()</code>	270
6.26.2.18	<code>spinNodesReadable()</code>	270
6.26.2.19	<code>spinNodesWritable()</code>	271
6.26.2.20	<code>spinNodeRegisterCallback()</code>	271
6.27	IValue Access	273
6.27.1	Detailed Description	273
6.27.2	Function Documentation	273
6.27.2.1	<code>spinNodeFromString()</code>	273
6.27.2.2	<code>spinNodeFromStringEx()</code>	274
6.27.2.3	<code>spinNodeToString()</code>	274
6.27.2.4	<code>spinNodeToStringEx()</code>	275
6.28	String Access	276
6.28.1	Detailed Description	276
6.28.2	Function Documentation	276
6.28.2.1	<code>spinStringGetMaxLength()</code>	276
6.28.2.2	<code>spinStringGetValue()</code>	277
6.28.2.3	<code>spinStringGetValueEx()</code>	277
6.28.2.4	<code>spinStringSetValue()</code>	278
6.28.2.5	<code>spinStringSetValueEx()</code>	278
6.29	Integer Access	280
6.29.1	Detailed Description	280
6.29.2	Function Documentation	280
6.29.2.1	<code>spinIntegerGetInc()</code>	280
6.29.2.2	<code>spinIntegerGetMax()</code>	281
6.29.2.3	<code>spinIntegerGetMin()</code>	281
6.29.2.4	<code>spinIntegerGetRepresentation()</code>	282
6.29.2.5	<code>spinIntegerGetValue()</code>	282

6.29.2.6	<a href="#">spinIntegerGetValueEx()</a>	283
6.29.2.7	<a href="#">spinIntegerSetValue()</a>	283
6.29.2.8	<a href="#">spinIntegerSetValueEx()</a>	284
6.30	<a href="#">IFloat Access</a>	285
6.30.1	<a href="#">Detailed Description</a>	285
6.30.2	<a href="#">Function Documentation</a>	285
6.30.2.1	<a href="#">spinFloatGetMax()</a>	285
6.30.2.2	<a href="#">spinFloatGetMin()</a>	286
6.30.2.3	<a href="#">spinFloatGetRepresentation()</a>	286
6.30.2.4	<a href="#">spinFloatGetUnit()</a>	287
6.30.2.5	<a href="#">spinFloatGetValue()</a>	287
6.30.2.6	<a href="#">spinFloatGetValueEx()</a>	288
6.30.2.7	<a href="#">spinFloatSetValue()</a>	288
6.30.2.8	<a href="#">spinFloatSetValueEx()</a>	289
6.31	<a href="#">IEnumeration Access</a>	290
6.31.1	<a href="#">Detailed Description</a>	290
6.31.2	<a href="#">Function Documentation</a>	290
6.31.2.1	<a href="#">spinEnumerationGetCurrentEntry()</a>	290
6.31.2.2	<a href="#">spinEnumerationGetEntryByIndex()</a>	291
6.31.2.3	<a href="#">spinEnumerationGetEntryByName()</a>	291
6.31.2.4	<a href="#">spinEnumerationGetNumEntries()</a>	292
6.31.2.5	<a href="#">spinEnumerationSetEnumValue()</a>	292
6.31.2.6	<a href="#">spinEnumerationSetIntValue()</a>	293
6.32	<a href="#">IEnumEntry Access</a>	294
6.32.1	<a href="#">Detailed Description</a>	294
6.32.2	<a href="#">Function Documentation</a>	294
6.32.2.1	<a href="#">spinEnumerationEntryGetEnumValue()</a>	294
6.32.2.2	<a href="#">spinEnumerationEntryGetIntValue()</a>	295
6.32.2.3	<a href="#">spinEnumerationEntryGetSymbolic()</a>	295
6.33	<a href="#">IBoolean Access</a>	297

6.33.1 Detailed Description . . . . .	297
6.33.2 Function Documentation . . . . .	297
6.33.2.1 spinBooleanGetValue() . . . . .	297
6.33.2.2 spinBooleanSetValue() . . . . .	298
6.34 ICommand Access . . . . .	299
6.34.1 Detailed Description . . . . .	299
6.34.2 Function Documentation . . . . .	299
6.34.2.1 spinCommandExecute() . . . . .	299
6.34.2.2 spinCommandIsDone() . . . . .	300
6.35 ICategory Access . . . . .	301
6.35.1 Detailed Description . . . . .	301
6.35.2 Function Documentation . . . . .	301
6.35.2.1 spinCategoryGetFeatureByIndex() . . . . .	301
6.35.2.2 spinCategoryGetNumFeatures() . . . . .	302
6.36 IRegister Access . . . . .	303
6.36.1 Detailed Description . . . . .	303
6.36.2 Function Documentation . . . . .	303
6.36.2.1 spinRegisterGet() . . . . .	304
6.36.2.2 spinRegisterGetAddress() . . . . .	304
6.36.2.3 spinRegisterGetEx() . . . . .	305
6.36.2.4 spinRegisterGetLength() . . . . .	305
6.36.2.5 spinRegisterSet() . . . . .	306
6.36.2.6 spinRegisterSetEx() . . . . .	306
6.36.2.7 spinRegisterSetReference() . . . . .	307
6.37 Spinnaker C GenICam Handles . . . . .	308
6.37.1 Detailed Description . . . . .	308
6.37.2 Typedef Documentation . . . . .	308
6.37.2.1 spinNodeCallbackFunction . . . . .	308
6.37.2.2 spinNodeCallbackHandle . . . . .	308
6.37.2.3 spinNodeHandle . . . . .	309

6.37.2.4	<a href="#">spinNodeMapHandle</a>	309
6.38	<a href="#">Spinnaker C GenICam Enumerations</a>	310
6.38.1	<a href="#">Detailed Description</a>	312
6.38.2	<a href="#">Enumeration Type Documentation</a>	312
6.38.2.1	<a href="#">spinAccessMode</a>	312
6.38.2.2	<a href="#">spinCachingMode</a>	313
6.38.2.3	<a href="#">spinDisplayNotation</a>	313
6.38.2.4	<a href="#">spinEndianness</a>	313
6.38.2.5	<a href="#">spinIncMode</a>	314
6.38.2.6	<a href="#">spinInputDirection</a>	314
6.38.2.7	<a href="#">spinInterfaceType</a>	314
6.38.2.8	<a href="#">spinLinkType</a>	315
6.38.2.9	<a href="#">spinNameSpace</a>	316
6.38.2.10	<a href="#">spinNodeType</a>	316
6.38.2.11	<a href="#">spinRepresentation</a>	317
6.38.2.12	<a href="#">spinSign</a>	317
6.38.2.13	<a href="#">spinSlope</a>	317
6.38.2.14	<a href="#">spinStandardNameSpace</a>	318
6.38.2.15	<a href="#">spinVisibility</a>	318
6.38.2.16	<a href="#">spinXMLValidation</a>	318
6.38.2.17	<a href="#">spinYesNo</a>	320
6.39	<a href="#">SpinVideo Recording Access</a>	321
6.39.1	<a href="#">Detailed Description</a>	321
6.39.2	<a href="#">Function Documentation</a>	321
6.39.2.1	<a href="#">spinVideoAppend()</a>	321
6.39.2.2	<a href="#">spinVideoClose()</a>	322
6.39.2.3	<a href="#">spinVideoOpenH264()</a>	322
6.39.2.4	<a href="#">spinVideoOpenMJPEG()</a>	322
6.39.2.5	<a href="#">spinVideoOpenUncompressed()</a>	322
6.39.2.6	<a href="#">spinVideoSetMaximumFileSize()</a>	322

6.40	Transport Layer Enumerations	324
6.40.1	Detailed Description	325
6.40.2	Enumeration Type Documentation	325
6.40.2.1	spinTLDeviceAccessStatusEnums	325
6.40.2.2	spinTLDeviceCurrentSpeedEnums	326
6.40.2.3	spinTLDeviceEndianessMechanismEnums	326
6.40.2.4	spinTLDeviceTypeEnums	326
6.40.2.5	spinTLFilterDriverStatusEnums	327
6.40.2.6	spinTLGenICamXMLLocationEnums	327
6.40.2.7	spinTLGevCCPEnums	328
6.40.2.8	spinTLGUIXMLLocationEnums	328
6.40.2.9	spinTLPOEStatusEnums	328
6.40.2.10	spinTLStreamBufferCountModeEnums	328
6.40.2.11	spinTLStreamBufferHandlingModeEnums	329
6.40.2.12	spinTLStreamDefaultBufferCountModeEnums	329
6.40.2.13	spinTLStreamTypeEnums	330
6.41	TLDevice Structures	331
6.41.1	Detailed Description	331
6.42	TLInterface Structures	332
6.42.1	Detailed Description	332
6.43	TLStream Structures	333
6.43.1	Detailed Description	333
6.44	TLSystem Structures	334
6.44.1	Detailed Description	334



<b>7</b>	<b>Data Structure Documentation</b>	<b>335</b>
7.1	actionCommandResult Struct Reference	335
7.1.1	Detailed Description	335
7.1.2	Field Documentation	335
7.1.2.1	DeviceAddress	335
7.1.2.2	Status	335
7.2	quickSpin Struct Reference	336
7.2.1	Field Documentation	348
7.2.1.1	AasRoiEnable	348
7.2.1.2	AasRoiHeight	348
7.2.1.3	AasRoiOffsetX	348
7.2.1.4	AasRoiOffsetY	348
7.2.1.5	AasRoiWidth	348
7.2.1.6	AcquisitionAbort	349
7.2.1.7	AcquisitionArm	349
7.2.1.8	AcquisitionBurstFrameCount	349
7.2.1.9	AcquisitionFrameCount	349
7.2.1.10	AcquisitionFrameRate	349
7.2.1.11	AcquisitionFrameRateEnable	349
7.2.1.12	AcquisitionLineRate	349
7.2.1.13	AcquisitionMode	349
7.2.1.14	AcquisitionResultingFrameRate	350
7.2.1.15	AcquisitionStart	350
7.2.1.16	AcquisitionStatus	350
7.2.1.17	AcquisitionStatusSelector	350
7.2.1.18	AcquisitionStop	350
7.2.1.19	ActionDeviceKey	350
7.2.1.20	ActionGroupKey	350
7.2.1.21	ActionGroupMask	350
7.2.1.22	ActionQueueSize	351

7.2.1.23	ActionSelector	351
7.2.1.24	ActionUnconditionalMode	351
7.2.1.25	AdaptiveCompressionEnable	351
7.2.1.26	AdcBitDepth	351
7.2.1.27	aPAUSEMACCtrlFramesReceived	351
7.2.1.28	aPAUSEMACCtrlFramesTransmitted	351
7.2.1.29	AutoAlgorithmSelector	351
7.2.1.30	AutoExposureControlLoopDamping	352
7.2.1.31	AutoExposureControlPriority	352
7.2.1.32	AutoExposureEVCompensation	352
7.2.1.33	AutoExposureExposureTimeLowerLimit	352
7.2.1.34	AutoExposureExposureTimeUpperLimit	352
7.2.1.35	AutoExposureGainLowerLimit	352
7.2.1.36	AutoExposureGainUpperLimit	352
7.2.1.37	AutoExposureGreyValueLowerLimit	352
7.2.1.38	AutoExposureGreyValueUpperLimit	353
7.2.1.39	AutoExposureLightingMode	353
7.2.1.40	AutoExposureMeteringMode	353
7.2.1.41	AutoExposureTargetGreyValue	353
7.2.1.42	AutoExposureTargetGreyValueAuto	353
7.2.1.43	BalanceRatio	353
7.2.1.44	BalanceRatioSelector	353
7.2.1.45	BalanceWhiteAuto	353
7.2.1.46	BalanceWhiteAutoDamping	354
7.2.1.47	BalanceWhiteAutoLowerLimit	354
7.2.1.48	BalanceWhiteAutoProfile	354
7.2.1.49	BalanceWhiteAutoUpperLimit	354
7.2.1.50	BinningHorizontal	354
7.2.1.51	BinningHorizontalMode	354
7.2.1.52	BinningSelector	354

7.2.1.53	BinningVertical	354
7.2.1.54	BinningVerticalMode	355
7.2.1.55	BlackLevel	355
7.2.1.56	BlackLevelAuto	355
7.2.1.57	BlackLevelAutoBalance	355
7.2.1.58	BlackLevelClampingEnable	355
7.2.1.59	BlackLevelRaw	355
7.2.1.60	BlackLevelSelector	355
7.2.1.61	ChunkBlackLevel	355
7.2.1.62	ChunkBlackLevelSelector	356
7.2.1.63	ChunkCounterSelector	356
7.2.1.64	ChunkCounterValue	356
7.2.1.65	ChunkCRC	356
7.2.1.66	ChunkEnable	356
7.2.1.67	ChunkEncoderSelector	356
7.2.1.68	ChunkEncoderStatus	356
7.2.1.69	ChunkEncoderValue	356
7.2.1.70	ChunkExposureEndLineStatusAll	357
7.2.1.71	ChunkExposureTime	357
7.2.1.72	ChunkExposureTimeSelector	357
7.2.1.73	ChunkFrameID	357
7.2.1.74	ChunkGain	357
7.2.1.75	ChunkGainSelector	357
7.2.1.76	ChunkHeight	357
7.2.1.77	ChunkImage	357
7.2.1.78	ChunkImageComponent	358
7.2.1.79	ChunkInferenceConfidence	358
7.2.1.80	ChunkInferenceResult	358
7.2.1.81	ChunkLinePitch	358
7.2.1.82	ChunkLineStatusAll	358

7.2.1.83	ChunkModeActive	358
7.2.1.84	ChunkOffsetX	358
7.2.1.85	ChunkOffsetY	358
7.2.1.86	ChunkPartSelector	359
7.2.1.87	ChunkPixelDynamicRangeMax	359
7.2.1.88	ChunkPixelDynamicRangeMin	359
7.2.1.89	ChunkPixelFormat	359
7.2.1.90	ChunkRegionID	359
7.2.1.91	ChunkScan3dAxisMax	359
7.2.1.92	ChunkScan3dAxisMin	359
7.2.1.93	ChunkScan3dCoordinateOffset	359
7.2.1.94	ChunkScan3dCoordinateReferenceSelector	360
7.2.1.95	ChunkScan3dCoordinateReferenceValue	360
7.2.1.96	ChunkScan3dCoordinateScale	360
7.2.1.97	ChunkScan3dCoordinateSelector	360
7.2.1.98	ChunkScan3dCoordinateSystem	360
7.2.1.99	ChunkScan3dCoordinateSystemReference	360
7.2.1.100	ChunkScan3dCoordinateTransformSelector	360
7.2.1.101	ChunkScan3dDistanceUnit	360
7.2.1.102	ChunkScan3dInvalidDataFlag	361
7.2.1.103	ChunkScan3dInvalidDataValue	361
7.2.1.104	ChunkScan3dOutputMode	361
7.2.1.105	ChunkScan3dTransformValue	361
7.2.1.106	ChunkScanLineSelector	361
7.2.1.107	ChunkSelector	361
7.2.1.108	ChunkSequencerSetActive	361
7.2.1.109	ChunkSerialData	361
7.2.1.110	ChunkSerialDataLength	362
7.2.1.111	ChunkSerialReceiveOverflow	362
7.2.1.112	ChunkSourceID	362

7.2.1.113 ChunkStreamChannelID . . . . .	362
7.2.1.114 ChunkTimerSelector . . . . .	362
7.2.1.115 ChunkTimerValue . . . . .	362
7.2.1.116 ChunkTimestamp . . . . .	362
7.2.1.117 ChunkTimestampLatchValue . . . . .	362
7.2.1.118 ChunkTransferBlockID . . . . .	363
7.2.1.119 ChunkTransferQueueCurrentBlockCount . . . . .	363
7.2.1.120 ChunkTransferStreamID . . . . .	363
7.2.1.121 ChunkWidth . . . . .	363
7.2.1.122 CIConfiguration . . . . .	363
7.2.1.123 CITimeSlotsCount . . . . .	363
7.2.1.124 ColorTransformationEnable . . . . .	363
7.2.1.125 ColorTransformationSelector . . . . .	363
7.2.1.126 ColorTransformationValue . . . . .	364
7.2.1.127 ColorTransformationValueSelector . . . . .	364
7.2.1.128 CompressionRatio . . . . .	364
7.2.1.129 CounterDelay . . . . .	364
7.2.1.130 CounterDuration . . . . .	364
7.2.1.131 CounterEventActivation . . . . .	364
7.2.1.132 CounterEventSource . . . . .	364
7.2.1.133 CounterReset . . . . .	364
7.2.1.134 CounterResetActivation . . . . .	365
7.2.1.135 CounterResetSource . . . . .	365
7.2.1.136 CounterSelector . . . . .	365
7.2.1.137 CounterStatus . . . . .	365
7.2.1.138 CounterTriggerActivation . . . . .	365
7.2.1.139 CounterTriggerSource . . . . .	365
7.2.1.140 CounterValue . . . . .	365
7.2.1.141 CounterValueAtReset . . . . .	365
7.2.1.142 CxpConnectionSelector . . . . .	366

7.2.1.143 CxpConnectionTestErrorCount . . . . .	366
7.2.1.144 CxpConnectionTestMode . . . . .	366
7.2.1.145 CxpConnectionTestPacketCount . . . . .	366
7.2.1.146 CxpLinkConfiguration . . . . .	366
7.2.1.147 CxpLinkConfigurationPreferred . . . . .	366
7.2.1.148 CxpLinkConfigurationStatus . . . . .	366
7.2.1.149 CxpPoCxpAuto . . . . .	366
7.2.1.150 CxpPoCxpStatus . . . . .	367
7.2.1.151 CxpPoCxpTripReset . . . . .	367
7.2.1.152 CxpPoCxpTurnOff . . . . .	367
7.2.1.153 DecimationHorizontal . . . . .	367
7.2.1.154 DecimationHorizontalMode . . . . .	367
7.2.1.155 DecimationSelector . . . . .	367
7.2.1.156 DecimationVertical . . . . .	367
7.2.1.157 DecimationVerticalMode . . . . .	367
7.2.1.158 DefectCorrectionMode . . . . .	368
7.2.1.159 DefectCorrectStaticEnable . . . . .	368
7.2.1.160 DefectTableApply . . . . .	368
7.2.1.161 DefectTableCoordinateX . . . . .	368
7.2.1.162 DefectTableCoordinateY . . . . .	368
7.2.1.163 DefectTableFactoryRestore . . . . .	368
7.2.1.164 DefectTableIndex . . . . .	368
7.2.1.165 DefectTablePixelCount . . . . .	368
7.2.1.166 DefectTableSave . . . . .	369
7.2.1.167 Deinterlacing . . . . .	369
7.2.1.168 DeviceCharacterSet . . . . .	369
7.2.1.169 DeviceClockFrequency . . . . .	369
7.2.1.170 DeviceClockSelector . . . . .	369
7.2.1.171 DeviceConnectionSelector . . . . .	369
7.2.1.172 DeviceConnectionSpeed . . . . .	369

7.2.1.173 DeviceConnectionStatus . . . . .	369
7.2.1.174 DeviceEventChannelCount . . . . .	370
7.2.1.175 DeviceFamilyName . . . . .	370
7.2.1.176 DeviceFeaturePersistenceEnd . . . . .	370
7.2.1.177 DeviceFeaturePersistenceStart . . . . .	370
7.2.1.178 DeviceFirmwareVersion . . . . .	370
7.2.1.179 DeviceGenCPVersionMajor . . . . .	370
7.2.1.180 DeviceGenCPVersionMinor . . . . .	370
7.2.1.181 DeviceID . . . . .	370
7.2.1.182 DeviceIndicatorMode . . . . .	371
7.2.1.183 DeviceLinkBandwidthReserve . . . . .	371
7.2.1.184 DeviceLinkCommandTimeout . . . . .	371
7.2.1.185 DeviceLinkConnectionCount . . . . .	371
7.2.1.186 DeviceLinkCurrentThroughput . . . . .	371
7.2.1.187 DeviceLinkHeartbeatMode . . . . .	371
7.2.1.188 DeviceLinkHeartbeatTimeout . . . . .	371
7.2.1.189 DeviceLinkSelector . . . . .	371
7.2.1.190 DeviceLinkSpeed . . . . .	372
7.2.1.191 DeviceLinkThroughputLimit . . . . .	372
7.2.1.192 DeviceLinkThroughputLimitMode . . . . .	372
7.2.1.193 DeviceManifestEntrySelector . . . . .	372
7.2.1.194 DeviceManifestPrimaryURL . . . . .	372
7.2.1.195 DeviceManifestSchemaMajorVersion . . . . .	372
7.2.1.196 DeviceManifestSchemaMinorVersion . . . . .	372
7.2.1.197 DeviceManifestSecondaryURL . . . . .	372
7.2.1.198 DeviceManifestXMLMajorVersion . . . . .	373
7.2.1.199 DeviceManifestXMLMinorVersion . . . . .	373
7.2.1.200 DeviceManifestXMLSubMinorVersion . . . . .	373
7.2.1.201 DeviceManufacturerInfo . . . . .	373
7.2.1.202 DeviceMaxThroughput . . . . .	373

7.2.1.203 DeviceModelName . . . . .	373
7.2.1.204 DevicePowerSupplySelector . . . . .	373
7.2.1.205 DeviceRegistersCheck . . . . .	373
7.2.1.206 DeviceRegistersEndianness . . . . .	374
7.2.1.207 DeviceRegistersStreamingEnd . . . . .	374
7.2.1.208 DeviceRegistersStreamingStart . . . . .	374
7.2.1.209 DeviceRegistersValid . . . . .	374
7.2.1.210 DeviceReset . . . . .	374
7.2.1.211 DeviceScanType . . . . .	374
7.2.1.212 DeviceSerialNumber . . . . .	374
7.2.1.213 DeviceSerialPortBaudRate . . . . .	374
7.2.1.214 DeviceSerialPortSelector . . . . .	375
7.2.1.215 DeviceSFNCVersionMajor . . . . .	375
7.2.1.216 DeviceSFNCVersionMinor . . . . .	375
7.2.1.217 DeviceSFNCVersionSubMinor . . . . .	375
7.2.1.218 DeviceStreamChannelCount . . . . .	375
7.2.1.219 DeviceStreamChannelEndianness . . . . .	375
7.2.1.220 DeviceStreamChannelLink . . . . .	375
7.2.1.221 DeviceStreamChannelPacketSize . . . . .	375
7.2.1.222 DeviceStreamChannelSelector . . . . .	376
7.2.1.223 DeviceStreamChannelType . . . . .	376
7.2.1.224 DeviceTapGeometry . . . . .	376
7.2.1.225 DeviceTemperature . . . . .	376
7.2.1.226 DeviceTemperatureSelector . . . . .	376
7.2.1.227 DeviceTLType . . . . .	376
7.2.1.228 DeviceTLVersionMajor . . . . .	376
7.2.1.229 DeviceTLVersionMinor . . . . .	376
7.2.1.230 DeviceTLVersionSubMinor . . . . .	377
7.2.1.231 DeviceType . . . . .	377
7.2.1.232 DeviceUptime . . . . .	377



7.2.1.233 DeviceUserID . . . . .	377
7.2.1.234 DeviceVendorName . . . . .	377
7.2.1.235 DeviceVersion . . . . .	377
7.2.1.236 EncoderDivider . . . . .	377
7.2.1.237 EncoderMode . . . . .	377
7.2.1.238 EncoderOutputMode . . . . .	378
7.2.1.239 EncoderReset . . . . .	378
7.2.1.240 EncoderResetActivation . . . . .	378
7.2.1.241 EncoderResetSource . . . . .	378
7.2.1.242 EncoderSelector . . . . .	378
7.2.1.243 EncoderSourceA . . . . .	378
7.2.1.244 EncoderSourceB . . . . .	378
7.2.1.245 EncoderStatus . . . . .	378
7.2.1.246 EncoderTimeout . . . . .	379
7.2.1.247 EncoderValue . . . . .	379
7.2.1.248 EncoderValueAtReset . . . . .	379
7.2.1.249 EnumerationCount . . . . .	379
7.2.1.250 EventAcquisitionEnd . . . . .	379
7.2.1.251 EventAcquisitionEndFrameID . . . . .	379
7.2.1.252 EventAcquisitionEndTimestamp . . . . .	379
7.2.1.253 EventAcquisitionError . . . . .	379
7.2.1.254 EventAcquisitionErrorFrameID . . . . .	380
7.2.1.255 EventAcquisitionErrorTimestamp . . . . .	380
7.2.1.256 EventAcquisitionStart . . . . .	380
7.2.1.257 EventAcquisitionStartFrameID . . . . .	380
7.2.1.258 EventAcquisitionStartTimestamp . . . . .	380
7.2.1.259 EventAcquisitionTransferEnd . . . . .	380
7.2.1.260 EventAcquisitionTransferEndFrameID . . . . .	380
7.2.1.261 EventAcquisitionTransferEndTimestamp . . . . .	380
7.2.1.262 EventAcquisitionTransferStart . . . . .	381

7.2.1.263 EventAcquisitionTransferStartFrameID . . . . .	381
7.2.1.264 EventAcquisitionTransferStartTimestamp . . . . .	381
7.2.1.265 EventAcquisitionTrigger . . . . .	381
7.2.1.266 EventAcquisitionTriggerFrameID . . . . .	381
7.2.1.267 EventAcquisitionTriggerTimestamp . . . . .	381
7.2.1.268 EventActionLate . . . . .	381
7.2.1.269 EventActionLateFrameID . . . . .	381
7.2.1.270 EventActionLateTimestamp . . . . .	382
7.2.1.271 EventCounter0End . . . . .	382
7.2.1.272 EventCounter0EndFrameID . . . . .	382
7.2.1.273 EventCounter0EndTimestamp . . . . .	382
7.2.1.274 EventCounter0Start . . . . .	382
7.2.1.275 EventCounter0StartFrameID . . . . .	382
7.2.1.276 EventCounter0StartTimestamp . . . . .	382
7.2.1.277 EventCounter1End . . . . .	382
7.2.1.278 EventCounter1EndFrameID . . . . .	383
7.2.1.279 EventCounter1EndTimestamp . . . . .	383
7.2.1.280 EventCounter1Start . . . . .	383
7.2.1.281 EventCounter1StartFrameID . . . . .	383
7.2.1.282 EventCounter1StartTimestamp . . . . .	383
7.2.1.283 EventEncoder0Restarted . . . . .	383
7.2.1.284 EventEncoder0RestartedFrameID . . . . .	383
7.2.1.285 EventEncoder0RestartedTimestamp . . . . .	383
7.2.1.286 EventEncoder0Stopped . . . . .	384
7.2.1.287 EventEncoder0StoppedFrameID . . . . .	384
7.2.1.288 EventEncoder0StoppedTimestamp . . . . .	384
7.2.1.289 EventEncoder1Restarted . . . . .	384
7.2.1.290 EventEncoder1RestartedFrameID . . . . .	384
7.2.1.291 EventEncoder1RestartedTimestamp . . . . .	384
7.2.1.292 EventEncoder1Stopped . . . . .	384

7.2.1.293 EventEncoder1StoppedFrameID . . . . .	384
7.2.1.294 EventEncoder1StoppedTimestamp . . . . .	385
7.2.1.295 EventError . . . . .	385
7.2.1.296 EventErrorCode . . . . .	385
7.2.1.297 EventErrorFrameID . . . . .	385
7.2.1.298 EventErrorTimestamp . . . . .	385
7.2.1.299 EventExposureEnd . . . . .	385
7.2.1.300 EventExposureEndFrameID . . . . .	385
7.2.1.301 EventExposureEndTimestamp . . . . .	385
7.2.1.302 EventExposureStart . . . . .	386
7.2.1.303 EventExposureStartFrameID . . . . .	386
7.2.1.304 EventExposureStartTimestamp . . . . .	386
7.2.1.305 EventFrameBurstEnd . . . . .	386
7.2.1.306 EventFrameBurstEndFrameID . . . . .	386
7.2.1.307 EventFrameBurstEndTimestamp . . . . .	386
7.2.1.308 EventFrameBurstStart . . . . .	386
7.2.1.309 EventFrameBurstStartFrameID . . . . .	386
7.2.1.310 EventFrameBurstStartTimestamp . . . . .	387
7.2.1.311 EventFrameEnd . . . . .	387
7.2.1.312 EventFrameEndFrameID . . . . .	387
7.2.1.313 EventFrameEndTimestamp . . . . .	387
7.2.1.314 EventFrameStart . . . . .	387
7.2.1.315 EventFrameStartFrameID . . . . .	387
7.2.1.316 EventFrameStartTimestamp . . . . .	387
7.2.1.317 EventFrameTransferEnd . . . . .	387
7.2.1.318 EventFrameTransferEndFrameID . . . . .	388
7.2.1.319 EventFrameTransferEndTimestamp . . . . .	388
7.2.1.320 EventFrameTransferStart . . . . .	388
7.2.1.321 EventFrameTransferStartFrameID . . . . .	388
7.2.1.322 EventFrameTransferStartTimestamp . . . . .	388

7.2.1.323 EventFrameTrigger . . . . .	388
7.2.1.324 EventFrameTriggerFrameID . . . . .	388
7.2.1.325 EventFrameTriggerTimestamp . . . . .	388
7.2.1.326 EventLine0AnyEdge . . . . .	389
7.2.1.327 EventLine0AnyEdgeFrameID . . . . .	389
7.2.1.328 EventLine0AnyEdgeTimestamp . . . . .	389
7.2.1.329 EventLine0FallingEdge . . . . .	389
7.2.1.330 EventLine0FallingEdgeFrameID . . . . .	389
7.2.1.331 EventLine0FallingEdgeTimestamp . . . . .	389
7.2.1.332 EventLine0RisingEdge . . . . .	389
7.2.1.333 EventLine0RisingEdgeFrameID . . . . .	389
7.2.1.334 EventLine0RisingEdgeTimestamp . . . . .	390
7.2.1.335 EventLine1AnyEdge . . . . .	390
7.2.1.336 EventLine1AnyEdgeFrameID . . . . .	390
7.2.1.337 EventLine1AnyEdgeTimestamp . . . . .	390
7.2.1.338 EventLine1FallingEdge . . . . .	390
7.2.1.339 EventLine1FallingEdgeFrameID . . . . .	390
7.2.1.340 EventLine1FallingEdgeTimestamp . . . . .	390
7.2.1.341 EventLine1RisingEdge . . . . .	390
7.2.1.342 EventLine1RisingEdgeFrameID . . . . .	391
7.2.1.343 EventLine1RisingEdgeTimestamp . . . . .	391
7.2.1.344 EventLinkSpeedChange . . . . .	391
7.2.1.345 EventLinkSpeedChangeFrameID . . . . .	391
7.2.1.346 EventLinkSpeedChangeTimestamp . . . . .	391
7.2.1.347 EventLinkTrigger0 . . . . .	391
7.2.1.348 EventLinkTrigger0FrameID . . . . .	391
7.2.1.349 EventLinkTrigger0Timestamp . . . . .	391
7.2.1.350 EventLinkTrigger1 . . . . .	392
7.2.1.351 EventLinkTrigger1FrameID . . . . .	392
7.2.1.352 EventLinkTrigger1Timestamp . . . . .	392

7.2.1.353 EventNotification . . . . .	392
7.2.1.354 EventSelector . . . . .	392
7.2.1.355 EventSequencerSetChange . . . . .	392
7.2.1.356 EventSequencerSetChangeFrameID . . . . .	392
7.2.1.357 EventSequencerSetChangeTimestamp . . . . .	392
7.2.1.358 EventSerialData . . . . .	393
7.2.1.359 EventSerialDataLength . . . . .	393
7.2.1.360 EventSerialPortReceive . . . . .	393
7.2.1.361 EventSerialPortReceiveTimestamp . . . . .	393
7.2.1.362 EventSerialReceiveOverflow . . . . .	393
7.2.1.363 EventStream0TransferBlockEnd . . . . .	393
7.2.1.364 EventStream0TransferBlockEndFrameID . . . . .	393
7.2.1.365 EventStream0TransferBlockEndTimestamp . . . . .	393
7.2.1.366 EventStream0TransferBlockStart . . . . .	394
7.2.1.367 EventStream0TransferBlockStartFrameID . . . . .	394
7.2.1.368 EventStream0TransferBlockStartTimestamp . . . . .	394
7.2.1.369 EventStream0TransferBlockTrigger . . . . .	394
7.2.1.370 EventStream0TransferBlockTriggerFrameID . . . . .	394
7.2.1.371 EventStream0TransferBlockTriggerTimestamp . . . . .	394
7.2.1.372 EventStream0TransferBurstEnd . . . . .	394
7.2.1.373 EventStream0TransferBurstEndFrameID . . . . .	394
7.2.1.374 EventStream0TransferBurstEndTimestamp . . . . .	395
7.2.1.375 EventStream0TransferBurstStart . . . . .	395
7.2.1.376 EventStream0TransferBurstStartFrameID . . . . .	395
7.2.1.377 EventStream0TransferBurstStartTimestamp . . . . .	395
7.2.1.378 EventStream0TransferEnd . . . . .	395
7.2.1.379 EventStream0TransferEndFrameID . . . . .	395
7.2.1.380 EventStream0TransferEndTimestamp . . . . .	395
7.2.1.381 EventStream0TransferOverflow . . . . .	395
7.2.1.382 EventStream0TransferOverflowFrameID . . . . .	396

7.2.1.383 EventStream0TransferOverflowTimestamp . . . . .	396
7.2.1.384 EventStream0TransferPause . . . . .	396
7.2.1.385 EventStream0TransferPauseFrameID . . . . .	396
7.2.1.386 EventStream0TransferPauseTimestamp . . . . .	396
7.2.1.387 EventStream0TransferResume . . . . .	396
7.2.1.388 EventStream0TransferResumeFrameID . . . . .	396
7.2.1.389 EventStream0TransferResumeTimestamp . . . . .	396
7.2.1.390 EventStream0TransferStart . . . . .	397
7.2.1.391 EventStream0TransferStartFrameID . . . . .	397
7.2.1.392 EventStream0TransferStartTimestamp . . . . .	397
7.2.1.393 EventTest . . . . .	397
7.2.1.394 EventTestTimestamp . . . . .	397
7.2.1.395 EventTimer0End . . . . .	397
7.2.1.396 EventTimer0EndFrameID . . . . .	397
7.2.1.397 EventTimer0EndTimestamp . . . . .	397
7.2.1.398 EventTimer0Start . . . . .	398
7.2.1.399 EventTimer0StartFrameID . . . . .	398
7.2.1.400 EventTimer0StartTimestamp . . . . .	398
7.2.1.401 EventTimer1End . . . . .	398
7.2.1.402 EventTimer1EndFrameID . . . . .	398
7.2.1.403 EventTimer1EndTimestamp . . . . .	398
7.2.1.404 EventTimer1Start . . . . .	398
7.2.1.405 EventTimer1StartFrameID . . . . .	398
7.2.1.406 EventTimer1StartTimestamp . . . . .	399
7.2.1.407 ExposureActiveMode . . . . .	399
7.2.1.408 ExposureAuto . . . . .	399
7.2.1.409 ExposureMode . . . . .	399
7.2.1.410 ExposureTime . . . . .	399
7.2.1.411 ExposureTimeMode . . . . .	399
7.2.1.412 ExposureTimeSelector . . . . .	399

7.2.1.413 FactoryReset . . . . .	399
7.2.1.414 FileAccessBuffer . . . . .	400
7.2.1.415 FileAccessLength . . . . .	400
7.2.1.416 FileAccessOffset . . . . .	400
7.2.1.417 FileOpenMode . . . . .	400
7.2.1.418 FileOperationExecute . . . . .	400
7.2.1.419 FileOperationResult . . . . .	400
7.2.1.420 FileOperationSelector . . . . .	400
7.2.1.421 FileOperationStatus . . . . .	400
7.2.1.422 FileSelector . . . . .	401
7.2.1.423 FileSize . . . . .	401
7.2.1.424 Gain . . . . .	401
7.2.1.425 GainAuto . . . . .	401
7.2.1.426 GainAutoBalance . . . . .	401
7.2.1.427 GainSelector . . . . .	401
7.2.1.428 Gamma . . . . .	401
7.2.1.429 GammaEnable . . . . .	401
7.2.1.430 GevActiveLinkCount . . . . .	402
7.2.1.431 GevCCP . . . . .	402
7.2.1.432 GevCurrentDefaultGateway . . . . .	402
7.2.1.433 GevCurrentIPAddress . . . . .	402
7.2.1.434 GevCurrentIPConfigurationDHCP . . . . .	402
7.2.1.435 GevCurrentIPConfigurationLLA . . . . .	402
7.2.1.436 GevCurrentIPConfigurationPersistentIP . . . . .	402
7.2.1.437 GevCurrentPhysicalLinkConfiguration . . . . .	402
7.2.1.438 GevCurrentSubnetMask . . . . .	403
7.2.1.439 GevDiscoveryAckDelay . . . . .	403
7.2.1.440 GevFirstURL . . . . .	403
7.2.1.441 GevGVCPEExtendedStatusCodes . . . . .	403
7.2.1.442 GevGVCPEExtendedStatusCodesSelector . . . . .	403

7.2.1.443	GevGVCPHeartbeatDisable	403
7.2.1.444	GevGVCPPendingAck	403
7.2.1.445	GevGVCPPendingTimeout	403
7.2.1.446	GevGVSPExtendedIDMode	404
7.2.1.447	GevHeartbeatTimeout	404
7.2.1.448	GevIEEE1588	404
7.2.1.449	GevIEEE1588ClockAccuracy	404
7.2.1.450	GevIEEE1588Mode	404
7.2.1.451	GevIEEE1588Status	404
7.2.1.452	GevInterfaceSelector	404
7.2.1.453	GevIPConfigurationStatus	404
7.2.1.454	GevMACAddress	405
7.2.1.455	GevMCDA	405
7.2.1.456	GevMCPHostPort	405
7.2.1.457	GevMCRC	405
7.2.1.458	GevMCSP	405
7.2.1.459	GevMCTT	405
7.2.1.460	GevNumberOfInterfaces	405
7.2.1.461	GevPAUSEFrameReception	405
7.2.1.462	GevPAUSEFrameTransmission	406
7.2.1.463	GevPersistentDefaultGateway	406
7.2.1.464	GevPersistentIPAddress	406
7.2.1.465	GevPersistentSubnetMask	406
7.2.1.466	GevPhysicalLinkConfiguration	406
7.2.1.467	GevPrimaryApplicationIPAddress	406
7.2.1.468	GevPrimaryApplicationSocket	406
7.2.1.469	GevPrimaryApplicationSwitchoverKey	406
7.2.1.470	GevSCCFGAllInTransmission	407
7.2.1.471	GevSCCFGExtendedChunkData	407
7.2.1.472	GevSCCFGPacketResendDestination	407



7.2.1.473	GevSCCFGUnconditionalStreaming	407
7.2.1.474	GevSCDA	407
7.2.1.475	GevSCPD	407
7.2.1.476	GevSCPDirection	407
7.2.1.477	GevSCPHostPort	407
7.2.1.478	GevSCPInterfaceIndex	408
7.2.1.479	GevSCPSBigEndian	408
7.2.1.480	GevSCPSDoNotFragment	408
7.2.1.481	GevSCPSFireTestPacket	408
7.2.1.482	GevSCPSPacketSize	408
7.2.1.483	GevSCSP	408
7.2.1.484	GevSCZoneConfigurationLock	408
7.2.1.485	GevSCZoneCount	408
7.2.1.486	GevSCZoneDirectionAll	409
7.2.1.487	GevSecondURL	409
7.2.1.488	GevStreamChannelSelector	409
7.2.1.489	GevSupportedOption	409
7.2.1.490	GevSupportedOptionSelector	409
7.2.1.491	GevTimestampTickFrequency	409
7.2.1.492	GuiXmlManifestAddress	409
7.2.1.493	Height	409
7.2.1.494	HeightMax	410
7.2.1.495	ImageComponentEnable	410
7.2.1.496	ImageComponentSelector	410
7.2.1.497	ImageCompressionBitrate	410
7.2.1.498	ImageCompressionJPEGFormatOption	410
7.2.1.499	ImageCompressionMode	410
7.2.1.500	ImageCompressionQuality	410
7.2.1.501	ImageCompressionRateOption	410
7.2.1.502	IspEnable	411

7.2.1.503 LineFilterWidth . . . . .	411
7.2.1.504 LineFormat . . . . .	411
7.2.1.505 LineInputFilterSelector . . . . .	411
7.2.1.506 LineInverter . . . . .	411
7.2.1.507 LineMode . . . . .	411
7.2.1.508 LinePitch . . . . .	411
7.2.1.509 LineSelector . . . . .	411
7.2.1.510 LineSource . . . . .	412
7.2.1.511 LineStatus . . . . .	412
7.2.1.512 LineStatusAll . . . . .	412
7.2.1.513 LinkErrorCount . . . . .	412
7.2.1.514 LinkUptime . . . . .	412
7.2.1.515 LogicBlockLUTInputActivation . . . . .	412
7.2.1.516 LogicBlockLUTInputSelector . . . . .	412
7.2.1.517 LogicBlockLUTInputSource . . . . .	412
7.2.1.518 LogicBlockLUTOutputValue . . . . .	413
7.2.1.519 LogicBlockLUTOutputValueAll . . . . .	413
7.2.1.520 LogicBlockLUTRowIndex . . . . .	413
7.2.1.521 LogicBlockLUTSelector . . . . .	413
7.2.1.522 LogicBlockSelector . . . . .	413
7.2.1.523 LUTEnable . . . . .	413
7.2.1.524 LUTIndex . . . . .	413
7.2.1.525 LUTSelector . . . . .	413
7.2.1.526 LUTValue . . . . .	414
7.2.1.527 LUTValueAll . . . . .	414
7.2.1.528 MaxDeviceResetTime . . . . .	414
7.2.1.529 OffsetX . . . . .	414
7.2.1.530 OffsetY . . . . .	414
7.2.1.531 PacketResendRequestCount . . . . .	414
7.2.1.532 PayloadSize . . . . .	414

7.2.1.533 PixelColorFilter . . . . .	414
7.2.1.534 PixelDynamicRangeMax . . . . .	415
7.2.1.535 PixelDynamicRangeMin . . . . .	415
7.2.1.536 PixelFormat . . . . .	415
7.2.1.537 PixelFormatInfoID . . . . .	415
7.2.1.538 PixelFormatInfoSelector . . . . .	415
7.2.1.539 PixelSize . . . . .	415
7.2.1.540 PowerSupplyCurrent . . . . .	415
7.2.1.541 PowerSupplyVoltage . . . . .	415
7.2.1.542 RegionDestination . . . . .	416
7.2.1.543 RegionMode . . . . .	416
7.2.1.544 RegionSelector . . . . .	416
7.2.1.545 ReverseX . . . . .	416
7.2.1.546 ReverseY . . . . .	416
7.2.1.547 RgbTransformLightSource . . . . .	416
7.2.1.548 Saturation . . . . .	416
7.2.1.549 SaturationEnable . . . . .	416
7.2.1.550 Scan3dAxisMax . . . . .	417
7.2.1.551 Scan3dAxisMin . . . . .	417
7.2.1.552 Scan3dCoordinateOffset . . . . .	417
7.2.1.553 Scan3dCoordinateReferenceSelector . . . . .	417
7.2.1.554 Scan3dCoordinateReferenceValue . . . . .	417
7.2.1.555 Scan3dCoordinateScale . . . . .	417
7.2.1.556 Scan3dCoordinateSelector . . . . .	417
7.2.1.557 Scan3dCoordinateSystem . . . . .	417
7.2.1.558 Scan3dCoordinateSystemReference . . . . .	418
7.2.1.559 Scan3dCoordinateTransformSelector . . . . .	418
7.2.1.560 Scan3dDistanceUnit . . . . .	418
7.2.1.561 Scan3dInvalidDataFlag . . . . .	418
7.2.1.562 Scan3dInvalidDataValue . . . . .	418

7.2.1.563 Scan3dOutputMode . . . . .	418
7.2.1.564 Scan3dTransformValue . . . . .	418
7.2.1.565 SensorDescription . . . . .	418
7.2.1.566 SensorDigitizationTaps . . . . .	419
7.2.1.567 SensorHeight . . . . .	419
7.2.1.568 SensorShutterMode . . . . .	419
7.2.1.569 SensorTaps . . . . .	419
7.2.1.570 SensorWidth . . . . .	419
7.2.1.571 SequencerConfigurationMode . . . . .	419
7.2.1.572 SequencerConfigurationValid . . . . .	419
7.2.1.573 SequencerFeatureEnable . . . . .	419
7.2.1.574 SequencerMode . . . . .	420
7.2.1.575 SequencerPathSelector . . . . .	420
7.2.1.576 SequencerSetActive . . . . .	420
7.2.1.577 SequencerSetLoad . . . . .	420
7.2.1.578 SequencerSetNext . . . . .	420
7.2.1.579 SequencerSetSave . . . . .	420
7.2.1.580 SequencerSetSelector . . . . .	420
7.2.1.581 SequencerSetStart . . . . .	420
7.2.1.582 SequencerSetValid . . . . .	421
7.2.1.583 SequencerTriggerActivation . . . . .	421
7.2.1.584 SequencerTriggerSource . . . . .	421
7.2.1.585 SerialPortBaudRate . . . . .	421
7.2.1.586 SerialPortDataBits . . . . .	421
7.2.1.587 SerialPortParity . . . . .	421
7.2.1.588 SerialPortSelector . . . . .	421
7.2.1.589 SerialPortSource . . . . .	421
7.2.1.590 SerialPortStopBits . . . . .	422
7.2.1.591 SerialReceiveFramingErrorCount . . . . .	422
7.2.1.592 SerialReceiveParityErrorCount . . . . .	422

7.2.1.593 SerialReceiveQueueClear . . . . .	422
7.2.1.594 SerialReceiveQueueCurrentCharacterCount . . . . .	422
7.2.1.595 SerialReceiveQueueMaxCharacterCount . . . . .	422
7.2.1.596 SerialTransmitQueueCurrentCharacterCount . . . . .	422
7.2.1.597 SerialTransmitQueueMaxCharacterCount . . . . .	422
7.2.1.598 Sharpening . . . . .	423
7.2.1.599 SharpeningAuto . . . . .	423
7.2.1.600 SharpeningEnable . . . . .	423
7.2.1.601 SharpeningThreshold . . . . .	423
7.2.1.602 SoftwareSignalPulse . . . . .	423
7.2.1.603 SoftwareSignalSelector . . . . .	423
7.2.1.604 SourceCount . . . . .	423
7.2.1.605 SourceSelector . . . . .	423
7.2.1.606 Test0001 . . . . .	424
7.2.1.607 TestEventGenerate . . . . .	424
7.2.1.608 TestPattern . . . . .	424
7.2.1.609 TestPatternGeneratorSelector . . . . .	424
7.2.1.610 TestPendingAck . . . . .	424
7.2.1.611 TimerDelay . . . . .	424
7.2.1.612 TimerDuration . . . . .	424
7.2.1.613 TimerReset . . . . .	424
7.2.1.614 TimerSelector . . . . .	425
7.2.1.615 TimerStatus . . . . .	425
7.2.1.616 TimerTriggerActivation . . . . .	425
7.2.1.617 TimerTriggerSource . . . . .	425
7.2.1.618 TimerValue . . . . .	425
7.2.1.619 Timestamp . . . . .	425
7.2.1.620 TimestampLatch . . . . .	425
7.2.1.621 TimestampLatchValue . . . . .	425
7.2.1.622 TimestampReset . . . . .	426

7.2.1.623 TLParamsLocked . . . . .	426
7.2.1.624 TransferAbort . . . . .	426
7.2.1.625 TransferBlockCount . . . . .	426
7.2.1.626 TransferBurstCount . . . . .	426
7.2.1.627 TransferComponentSelector . . . . .	426
7.2.1.628 TransferControlMode . . . . .	426
7.2.1.629 TransferOperationMode . . . . .	426
7.2.1.630 TransferPause . . . . .	427
7.2.1.631 TransferQueueCurrentBlockCount . . . . .	427
7.2.1.632 TransferQueueMaxBlockCount . . . . .	427
7.2.1.633 TransferQueueMode . . . . .	427
7.2.1.634 TransferQueueOverflowCount . . . . .	427
7.2.1.635 TransferResume . . . . .	427
7.2.1.636 TransferSelector . . . . .	427
7.2.1.637 TransferStart . . . . .	427
7.2.1.638 TransferStatus . . . . .	428
7.2.1.639 TransferStatusSelector . . . . .	428
7.2.1.640 TransferStop . . . . .	428
7.2.1.641 TransferStreamChannel . . . . .	428
7.2.1.642 TransferTriggerActivation . . . . .	428
7.2.1.643 TransferTriggerMode . . . . .	428
7.2.1.644 TransferTriggerSelector . . . . .	428
7.2.1.645 TransferTriggerSource . . . . .	428
7.2.1.646 TriggerActivation . . . . .	429
7.2.1.647 TriggerDelay . . . . .	429
7.2.1.648 TriggerDivider . . . . .	429
7.2.1.649 TriggerEventTest . . . . .	429
7.2.1.650 TriggerMode . . . . .	429
7.2.1.651 TriggerMultiplier . . . . .	429
7.2.1.652 TriggerOverlap . . . . .	429

7.2.1.653	TriggerSelector	429
7.2.1.654	TriggerSoftware	430
7.2.1.655	TriggerSource	430
7.2.1.656	UserOutputSelector	430
7.2.1.657	UserOutputValue	430
7.2.1.658	UserOutputValueAll	430
7.2.1.659	UserOutputValueAllMask	430
7.2.1.660	UserSetDefault	430
7.2.1.661	UserSetFeatureEnable	430
7.2.1.662	UserSetLoad	431
7.2.1.663	UserSetSave	431
7.2.1.664	UserSetSelector	431
7.2.1.665	V3_3Enable	431
7.2.1.666	WhiteClip	431
7.2.1.667	WhiteClipSelector	431
7.2.1.668	Width	431
7.2.1.669	WidthMax	432
7.3	quickSpinTLDevice Struct Reference	432
7.3.1	Field Documentation	433
7.3.1.1	DeviceAccessStatus	433
7.3.1.2	DeviceCurrentSpeed	433
7.3.1.3	DeviceDisplayName	433
7.3.1.4	DeviceDriverVersion	433
7.3.1.5	DeviceEndianessMechanism	433
7.3.1.6	DeviceID	433
7.3.1.7	DeviceInstanceld	433
7.3.1.8	DeviceIsUpdater	434
7.3.1.9	DeviceLinkSpeed	434
7.3.1.10	DeviceLocation	434
7.3.1.11	DeviceModelName	434

7.3.1.12	DeviceMulticastMonitorMode	434
7.3.1.13	DeviceSerialNumber	434
7.3.1.14	DeviceType	434
7.3.1.15	DeviceU3VProtocol	434
7.3.1.16	DeviceUserID	435
7.3.1.17	DeviceVendorName	435
7.3.1.18	DeviceVersion	435
7.3.1.19	GenICamXMLLocation	435
7.3.1.20	GenICamXMLPath	435
7.3.1.21	GevCCP	435
7.3.1.22	GevDeviceDiscoverMaximumPacketSize	435
7.3.1.23	GevDeviceForceIP	435
7.3.1.24	GevDeviceGateway	436
7.3.1.25	GevDeviceIPAddress	436
7.3.1.26	GevDeviceIsWrongSubnet	436
7.3.1.27	GevDeviceMACAddress	436
7.3.1.28	GevDeviceMaximumPacketSize	436
7.3.1.29	GevDeviceMaximumRetryCount	436
7.3.1.30	GevDeviceModelsBigEndian	436
7.3.1.31	GevDevicePort	436
7.3.1.32	GevDeviceReadAndWriteTimeout	437
7.3.1.33	GevDeviceSubnetMask	437
7.3.1.34	GevVersionMajor	437
7.3.1.35	GevVersionMinor	437
7.3.1.36	GUIXMLLocation	437
7.3.1.37	GUIXMLPath	437
7.4	quickSpinTLInterface Struct Reference	438
7.4.1	Field Documentation	438
7.4.1.1	ActionCommand	439
7.4.1.2	AutoForceIP	439



7.4.1.3	DeviceAccessStatus	439
7.4.1.4	DeviceCount	439
7.4.1.5	DeviceID	439
7.4.1.6	DeviceModelName	439
7.4.1.7	DeviceSelector	439
7.4.1.8	DeviceUnlock	439
7.4.1.9	DeviceUpdateList	440
7.4.1.10	DeviceVendorName	440
7.4.1.11	FilterDriverStatus	440
7.4.1.12	GevActionDeviceKey	440
7.4.1.13	GevActionGroupKey	440
7.4.1.14	GevActionGroupMask	440
7.4.1.15	GevActionTime	440
7.4.1.16	GevDeviceIPAddress	440
7.4.1.17	GevDeviceMACAddress	441
7.4.1.18	GevDeviceSubnetMask	441
7.4.1.19	GevInterfaceGateway	441
7.4.1.20	GevInterfaceIPAddress	441
7.4.1.21	GevInterfaceMACAddress	441
7.4.1.22	GevInterfaceMTU	441
7.4.1.23	GevInterfaceReceiveLinkSpeed	441
7.4.1.24	GevInterfaceSubnetMask	441
7.4.1.25	GevInterfaceTransmitLinkSpeed	442
7.4.1.26	HostAdapterDriverVersion	442
7.4.1.27	HostAdapterName	442
7.4.1.28	HostAdapterVendor	442
7.4.1.29	IncompatibleDeviceCount	442
7.4.1.30	IncompatibleDeviceID	442
7.4.1.31	IncompatibleDeviceModelName	442
7.4.1.32	IncompatibleDeviceSelector	442

7.4.1.33	IncompatibleDeviceVendorName	443
7.4.1.34	IncompatibleGevDeviceIPAddress	443
7.4.1.35	IncompatibleGevDeviceMACAddress	443
7.4.1.36	IncompatibleGevDeviceSubnetMask	443
7.4.1.37	InterfaceDisplayName	443
7.4.1.38	InterfaceID	443
7.4.1.39	InterfaceType	443
7.4.1.40	POEStatus	444
7.5	quickSpinTLStream Struct Reference	444
7.5.1	Field Documentation	444
7.5.1.1	GevFailedPacketCount	444
7.5.1.2	GevMaximumNumberResendBuffers	445
7.5.1.3	GevMaximumNumberResendRequests	445
7.5.1.4	GevPacketResendMode	445
7.5.1.5	GevPacketResendTimeout	445
7.5.1.6	GevResendPacketCount	445
7.5.1.7	GevResendRequestCount	445
7.5.1.8	GevTotalPacketCount	445
7.5.1.9	StreamBlockTransferSize	445
7.5.1.10	StreamBufferCountManual	446
7.5.1.11	StreamBufferCountMax	446
7.5.1.12	StreamBufferCountMode	446
7.5.1.13	StreamBufferCountResult	446
7.5.1.14	StreamBufferHandlingMode	446
7.5.1.15	StreamBufferUnderrunCount	446
7.5.1.16	StreamCRCCheckEnable	446
7.5.1.17	StreamDefaultBufferCount	446
7.5.1.18	StreamDefaultBufferCountMax	447
7.5.1.19	StreamDefaultBufferCountMode	447
7.5.1.20	StreamFailedBufferCount	447

7.5.1.21	StreamID	447
7.5.1.22	StreamTotalBufferCount	447
7.5.1.23	StreamType	447
7.6	quickSpinTLSystem Struct Reference	447
7.6.1	Field Documentation	448
7.6.1.1	EnumerateGEVInterfaces	448
7.7	spinAVIOption Struct Reference	448
7.7.1	Detailed Description	448
7.7.2	Field Documentation	448
7.7.2.1	frameRate	448
7.7.2.2	reserved	449
7.8	spinBMPOption Struct Reference	449
7.8.1	Detailed Description	449
7.8.2	Field Documentation	449
7.8.2.1	indexedColor_8bit	449
7.8.2.2	reserved	449
7.9	spinChunkData Struct Reference	450
7.9.1	Detailed Description	450
7.9.2	Field Documentation	450
7.9.2.1	m_blackLevel	451
7.9.2.2	m_counterValue	451
7.9.2.3	m_cRC	451
7.9.2.4	m_encoderValue	451
7.9.2.5	m_exposureEndLineStatusAll	451
7.9.2.6	m_exposureTime	451
7.9.2.7	m_frameID	451
7.9.2.8	m_gain	451
7.9.2.9	m_height	452
7.9.2.10	m_image	452
7.9.2.11	m_inferenceConfidence	452

7.9.2.12	<a href="#">m_inferenceResult</a>	452
7.9.2.13	<a href="#">m_linePitch</a>	452
7.9.2.14	<a href="#">m_lineStatusAll</a>	452
7.9.2.15	<a href="#">m_offsetX</a>	452
7.9.2.16	<a href="#">m_offsetY</a>	452
7.9.2.17	<a href="#">m_partSelector</a>	453
7.9.2.18	<a href="#">m_pixelDynamicRangeMax</a>	453
7.9.2.19	<a href="#">m_pixelDynamicRangeMin</a>	453
7.9.2.20	<a href="#">m_scan3dAxisMax</a>	453
7.9.2.21	<a href="#">m_scan3dAxisMin</a>	453
7.9.2.22	<a href="#">m_scan3dCoordinateOffset</a>	453
7.9.2.23	<a href="#">m_scan3dCoordinateReferenceValue</a>	453
7.9.2.24	<a href="#">m_scan3dCoordinateScale</a>	453
7.9.2.25	<a href="#">m_scan3dInvalidDataValue</a>	454
7.9.2.26	<a href="#">m_scan3dTransformValue</a>	454
7.9.2.27	<a href="#">m_scanLineSelector</a>	454
7.9.2.28	<a href="#">m_sequencerSetActive</a>	454
7.9.2.29	<a href="#">m_serialDataLength</a>	454
7.9.2.30	<a href="#">m_streamChannelID</a>	454
7.9.2.31	<a href="#">m_timerValue</a>	454
7.9.2.32	<a href="#">m_timestamp</a>	454
7.9.2.33	<a href="#">m_timestampLatchValue</a>	455
7.9.2.34	<a href="#">m_transferBlockID</a>	455
7.9.2.35	<a href="#">m_transferQueueCurrentBlockCount</a>	455
7.9.2.36	<a href="#">m_width</a>	455
7.10	<a href="#">spinH264Option Struct Reference</a>	455
7.10.1	<a href="#">Detailed Description</a>	456
7.10.2	<a href="#">Field Documentation</a>	456
7.10.2.1	<a href="#">bitrate</a>	456
7.10.2.2	<a href="#">frameRate</a>	456

7.10.2.3	height	456
7.10.2.4	reserved	456
7.10.2.5	width	456
7.11	spinJPEGOption Struct Reference	457
7.11.1	Detailed Description	457
7.11.2	Field Documentation	457
7.11.2.1	progressive	457
7.11.2.2	quality	457
7.11.2.3	reserved	458
7.12	spinJPG2Option Struct Reference	458
7.12.1	Detailed Description	458
7.12.2	Field Documentation	458
7.12.2.1	quality	458
7.12.2.2	reserved	458
7.13	spinLibraryVersion Struct Reference	459
7.13.1	Detailed Description	459
7.13.2	Field Documentation	459
7.13.2.1	build	459
7.13.2.2	major	459
7.13.2.3	minor	459
7.13.2.4	type	460
7.14	spinMJPGOption Struct Reference	460
7.14.1	Detailed Description	460
7.14.2	Field Documentation	460
7.14.2.1	frameRate	460
7.14.2.2	quality	460
7.14.2.3	reserved	461
7.15	spinPGMOption Struct Reference	461
7.15.1	Detailed Description	461
7.15.2	Field Documentation	461

7.15.2.1	binaryFile	461
7.15.2.2	reserved	461
7.16	spinPNGOption Struct Reference	462
7.16.1	Detailed Description	462
7.16.2	Field Documentation	462
7.16.2.1	compressionLevel	462
7.16.2.2	interlaced	462
7.16.2.3	reserved	462
7.17	spinPPMOption Struct Reference	463
7.17.1	Detailed Description	463
7.17.2	Field Documentation	463
7.17.2.1	binaryFile	463
7.17.2.2	reserved	463
7.18	spinTIFFOption Struct Reference	463
7.18.1	Detailed Description	464
7.18.2	Field Documentation	464
7.18.2.1	compression	464
7.18.2.2	reserved	464
<b>8</b>	<b>File Documentation</b>	<b>465</b>
8.1	doc/Doxygen/spindocs/C/Licensing.dox File Reference	465
8.2	doc/Doxygen/spindocs/C/MainPage.dox File Reference	465
8.3	include/spinc/CameraDefsC.h File Reference	465
8.4	include/spinc/ChunkDataDefC.h File Reference	498
8.5	include/spinc/QuickSpinC.h File Reference	499
8.6	include/spinc/QuickSpinDefsC.h File Reference	499
8.6.1	Typedef Documentation	500
8.6.1.1	quickSpinBooleanNode	500
8.6.1.2	quickSpinCommandNode	500
8.6.1.3	quickSpinEnumerationNode	500
8.6.1.4	quickSpinFloatNode	501

8.6.1.5	<a href="#">quickSpinIntegerNode</a>	501
8.6.1.6	<a href="#">quickSpinRegisterNode</a>	501
8.6.1.7	<a href="#">quickSpinStringNode</a>	501
8.7	<a href="#">include/spinc/SpinnakerC.h File Reference</a>	501
8.7.1	<a href="#">Function Documentation</a>	510
8.7.1.1	<a href="#">spinCameraForceIP()</a>	510
8.8	<a href="#">include/spinc/SpinnakerDefsC.h File Reference</a>	511
8.9	<a href="#">include/spinc/SpinnakerGenApiC.h File Reference</a>	516
8.10	<a href="#">include/spinc/SpinnakerGenApiDefsC.h File Reference</a>	520
8.11	<a href="#">include/spinc/SpinnakerPlatformC.h File Reference</a>	523
8.11.1	<a href="#">Macro Definition Documentation</a>	523
8.11.1.1	<a href="#">SPINNAKERC_API</a>	523
8.12	<a href="#">include/spinc/SpinVideoC.h File Reference</a>	524
8.13	<a href="#">include/spinc/TransportLayerDefsC.h File Reference</a>	524
8.14	<a href="#">include/spinc/TransportLayerDeviceC.h File Reference</a>	526
8.15	<a href="#">include/spinc/TransportLayerInterfaceC.h File Reference</a>	527
8.16	<a href="#">include/spinc/TransportLayerStreamC.h File Reference</a>	527
8.17	<a href="#">include/spinc/TransportLayerSystemC.h File Reference</a>	528
<b>Index</b>		<b>529</b>





# **Chapter 1**

## **Introduction**

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



## Chapter 2

# Software Licensing Information

Table 2.1 License table

Component	License
Spinnaker	Copyright © 2017 FLIR Integrated Imaging Solutions, Inc. All Rights Reserved. This software is the confidential and proprietary information of FLIR Integrated Imaging Solutions, Inc. ("Confidential Information"). You shall not disclose such Confidential Information and shall use it only in accordance with the terms of the license agreement you entered into with FLIR Integrated Imaging Solutions, Inc. (FLIR). FLIR MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. FLIR SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES.
GenICam	GenICam License
AdapterManager	The Code Project Open License (CPO-OL)
Make ListView.ScrollIntoView Scroll the Item into the Center of the ListView	WP:CC-BY-SA License
Work with Bitmaps Faster in C#	The Code Project Open License (CPO-OL) 1.02
FreeImage	FreeImage public license
Boost	Boost Software License
Libusb	GPLv2.1 License
Libraw1394	GPLv2.0 License
FFMPEG	GPLv2.1 License
log4Net	Apache license 2.0
log4Cpp	GPL License

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



## Chapter 3

# Module Index

### 3.1 Modules

Here is a list of all modules:

Spinnaker C QuickSpin API . . . . .	129
QuickSpin Access . . . . .	130
Transport Layer Enumerations . . . . .	324
TLDevice Structures . . . . .	331
TLInterface Structures . . . . .	332
TLStream Structures . . . . .	333
TLSystem Structures . . . . .	334
Spinnaker C API . . . . .	132
Spinnaker C Definitions . . . . .	11
Camera Enumerations . . . . .	13
Chunk Data Structures . . . . .	128
Spinnaker C Handles . . . . .	240
Spinnaker C Function Signatures . . . . .	244
Spinnaker C Enumerations . . . . .	246
Spinnaker C Structures . . . . .	254
Error Handling . . . . .	134
System Access . . . . .	139
InterfaceList Access . . . . .	153
CameraList Access . . . . .	157
Interface Access . . . . .	163
Camera Access . . . . .	171
SpinVideo Recording Access . . . . .	321
Image Access . . . . .	183
Event Access . . . . .	212
ImageStatistics Access . . . . .	219
Logging Event Data Access . . . . .	228
Device Event Data Access . . . . .	233
AVIRecorder Access . . . . .	236
Chunk data access . . . . .	239
Spinnaker C GenICam API . . . . .	256
Node Map Access . . . . .	258
Node Access . . . . .	261
IValue Access . . . . .	273
String Access . . . . .	276

Integer Access . . . . .	280
IFloat Access . . . . .	285
IEnumeration Access . . . . .	290
IEnumEntry Access . . . . .	294
IBoolean Access . . . . .	297
ICommand Access . . . . .	299
ICategory Access . . . . .	301
IRegister Access . . . . .	303
Spinnaker C GenICam Handles . . . . .	308
Spinnaker C GenICam Enumerations . . . . .	310

## Chapter 4

# Data Structure Index

### 4.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">actionCommandResult</a>	
Action Command Result . . . . .	335
<a href="#">quickSpin</a> . . . . .	336
<a href="#">quickSpinTLDevice</a> . . . . .	432
<a href="#">quickSpinTLInterface</a> . . . . .	438
<a href="#">quickSpinTLStream</a> . . . . .	444
<a href="#">quickSpinTLSystem</a> . . . . .	447
<a href="#">spinAVIOption</a>	
Options for saving uncompressed videos . . . . .	448
<a href="#">spinBMPOption</a>	
Options for saving BMP images . . . . .	449
<a href="#">spinChunkData</a>	
The type of information that can be obtained from image chunk data . . . . .	450
<a href="#">spinH264Option</a>	
Options for saving H264 videos . . . . .	455
<a href="#">spinJPEGOption</a>	
Options for saving JPEG images . . . . .	457
<a href="#">spinJPG2Option</a>	
Options for saving JPEG 2000 images . . . . .	458
<a href="#">spinLibraryVersion</a>	
Provides easier access to the current version of Spinnaker . . . . .	459
<a href="#">spinMJPGOption</a>	
Options for saving MJPG videos . . . . .	460
<a href="#">spinPGMOption</a>	
Options for saving PGM images . . . . .	461
<a href="#">spinPNGOption</a>	
Options for saving PNG images . . . . .	462
<a href="#">spinPPMOption</a>	
Options for saving PPM images . . . . .	463
<a href="#">spinTIFFOption</a>	
Options for saving TIFF images . . . . .	463





## Chapter 5

# File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

include/spinc/ <a href="#">CameraDefsC.h</a>	465
include/spinc/ <a href="#">ChunkDataDefC.h</a>	498
include/spinc/ <a href="#">QuickSpinC.h</a>	499
include/spinc/ <a href="#">QuickSpinDefsC.h</a>	499
include/spinc/ <a href="#">SpinnakerC.h</a>	501
include/spinc/ <a href="#">SpinnakerDefsC.h</a>	511
include/spinc/ <a href="#">SpinnakerGenApiC.h</a>	516
include/spinc/ <a href="#">SpinnakerGenApiDefsC.h</a>	520
include/spinc/ <a href="#">SpinnakerPlatformC.h</a>	523
include/spinc/ <a href="#">SpinVideoC.h</a>	524
include/spinc/ <a href="#">TransportLayerDefsC.h</a>	524
include/spinc/ <a href="#">TransportLayerDeviceC.h</a>	526
include/spinc/ <a href="#">TransportLayerInterfaceC.h</a>	527
include/spinc/ <a href="#">TransportLayerStreamC.h</a>	527
include/spinc/ <a href="#">TransportLayerSystemC.h</a>	528



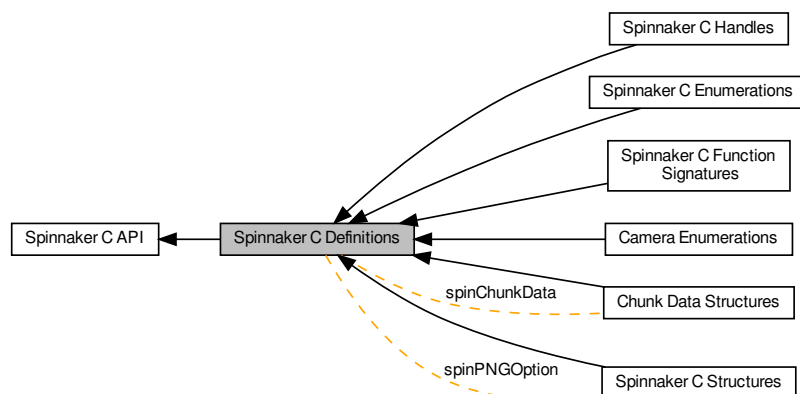
## Chapter 6

# Module Documentation

### 6.1 Spinnaker C Definitions

Definitions for Spinnaker C.

Collaboration diagram for Spinnaker C Definitions:



### Modules

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

## Data Structures

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

## Typedefs

- typedef uint8\_t [bool8\\_t](#)

## Variables

- static const [bool8\\_t](#) [False](#) = 0
- static const [bool8\\_t](#) [True](#) = 1

### 6.1.1 Detailed Description

Definitions for Spinnaker C.

Definitions for Spinnaker C API.

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

### 6.1.2 Typedef Documentation

#### 6.1.2.1 bool8\_t

```
typedef uint8_t bool8_t
```

### 6.1.3 Variable Documentation

#### 6.1.3.1 False

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

#### 6.1.3.2 True

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

## 6.2 Camera Enumerations

Collaboration diagram for Camera Enumerations:



### Enumerations

- enum `spinLUTSelectorEnums` {  
`LUTSelector_LUT1`,  
`NUM_LUTSELECTOR` }
- The enum definitions for camera nodes.*
- enum `spinExposureModeEnums` {  
`ExposureMode_Timed`,  
`ExposureMode_TriggerWidth`,  
`NUM_EXPOSUREMODE` }
- enum `spinAcquisitionModeEnums` {  
`AcquisitionMode_Continuous`,  
`AcquisitionMode_SingleFrame`,  
`AcquisitionMode_MultiFrame`,  
`NUM_ACQUISITIONMODE` }
- enum `spinTriggerSourceEnums` {  
`TriggerSource_Software`,  
`TriggerSource_Line0`,  
`TriggerSource_Line1`,  
`TriggerSource_Line2`,  
`TriggerSource_Line3`,  
`TriggerSource_UserOutput0`,  
`TriggerSource_UserOutput1`,  
`TriggerSource_UserOutput2`,  
`TriggerSource_UserOutput3`,  
`TriggerSource_Counter0Start`,  
`TriggerSource_Counter1Start`,  
`TriggerSource_Counter0End`,  
`TriggerSource_Counter1End`,  
`TriggerSource_LogicBlock0`,  
`TriggerSource_LogicBlock1`,  
`TriggerSource_Action0`,  
`NUM_TRIGGERSOURCE` }
- enum `spinTriggerActivationEnums` {  
`TriggerActivation_LevelLow`,  
`TriggerActivation_LevelHigh`,  
`TriggerActivation_FallingEdge`,  
`TriggerActivation_RisingEdge`,  
`TriggerActivation_AnyEdge`,  
`NUM_TRIGGERACTIVATION` }

- enum `spinSensorShutterModeEnums` {  
`SensorShutterMode_Global`,  
`SensorShutterMode_Rolling`,  
`SensorShutterMode_GlobalReset`,  
`NUM_SENSORSHUTTERMODE` }
- enum `spinTriggerModeEnums` {  
`TriggerMode_Off`,  
`TriggerMode_On`,  
`NUM_TRIGGERMODE` }
- enum `spinTriggerOverlapEnums` {  
`TriggerOverlap_Off`,  
`TriggerOverlap_ReadOut`,  
`TriggerOverlap_PreviousFrame`,  
`NUM_TRIGGEROVERLAP` }
- enum `spinTriggerSelectorEnums` {  
`TriggerSelector_AcquisitionStart`,  
`TriggerSelector_FrameStart`,  
`TriggerSelector_FrameBurstStart`,  
`NUM_TRIGGERSELECTOR` }
- enum `spinExposureAutoEnums` {  
`ExposureAuto_Off`,  
`ExposureAuto_Once`,  
`ExposureAuto_Continuous`,  
`NUM_EXPOSUREAUTO` }
- enum `spinEventSelectorEnums` {  
`EventSelector_Error`,  
`EventSelector_ExposureEnd`,  
`EventSelector_SerialPortReceive`,  
`NUM_EVENTSELECTOR` }
- enum `spinEventNotificationEnums` {  
`EventNotification_On`,  
`EventNotification_Off`,  
`NUM_EVENTNOTIFICATION` }
- enum `spinLogicBlockSelectorEnums` {  
`LogicBlockSelector_LogicBlock0`,  
`LogicBlockSelector_LogicBlock1`,  
`NUM_LOGICBLOCKSELECTOR` }
- enum `spinLogicBlockLUTInputActivationEnums` {  
`LogicBlockLUTInputActivation_LevelLow`,  
`LogicBlockLUTInputActivation_LevelHigh`,  
`LogicBlockLUTInputActivation_FallingEdge`,  
`LogicBlockLUTInputActivation_RisingEdge`,  
`LogicBlockLUTInputActivation_AnyEdge`,  
`NUM_LOGICBLOCKLUTINPUTACTIVATION` }
- enum `spinLogicBlockLUTInputSelectorEnums` {  
`LogicBlockLUTInputSelector_Input0`,  
`LogicBlockLUTInputSelector_Input1`,  
`LogicBlockLUTInputSelector_Input2`,  
`LogicBlockLUTInputSelector_Input3`,  
`NUM_LOGICBLOCKLUTINPUTSELECTOR` }
- enum `spinLogicBlockLUTInputSourceEnums` {  
`LogicBlockLUTInputSource_Zero`,  
`LogicBlockLUTInputSource_Line0`,  
`LogicBlockLUTInputSource_Line1`,  
`LogicBlockLUTInputSource_Line2`,  
`LogicBlockLUTInputSource_Line3`,  
`LogicBlockLUTInputSource_UserOutput0`,  
`LogicBlockLUTInputSource_UserOutput1`,

```

LogicBlockLUTInputSource_UserOutput2,
LogicBlockLUTInputSource_UserOutput3,
LogicBlockLUTInputSource_Counter0Start,
LogicBlockLUTInputSource_Counter1Start,
LogicBlockLUTInputSource_Counter0End,
LogicBlockLUTInputSource_Counter1End,
LogicBlockLUTInputSource_LogicBlock0,
LogicBlockLUTInputSource_LogicBlock1,
LogicBlockLUTInputSource_ExposureStart,
LogicBlockLUTInputSource_ExposureEnd,
LogicBlockLUTInputSource_FrameTriggerWait,
LogicBlockLUTInputSource_AcquisitionActive,
NUM_LOGICBLOCKLUTINPUTSOURCE }

• enum spinLogicBlockLUTSelectorEnums {
    LogicBlockLUTSelector_Value,
    LogicBlockLUTSelector_Enable,
    NUM_LOGICBLOCKLUTSELECTOR }

• enum spinColorTransformationSelectorEnums {
    ColorTransformationSelector_RGBtoRGB,
    ColorTransformationSelector_RGBtoYUV,
    NUM_COLORTRANSFORMATIONSELECTOR }

• enum spinRgbTransformLightSourceEnums {
    RgbTransformLightSource_General,
    RgbTransformLightSource_Tungsten2800K,
    RgbTransformLightSource_WarmFluorescent3000K,
    RgbTransformLightSource_CoolFluorescent4000K,
    RgbTransformLightSource_Daylight5000K,
    RgbTransformLightSource_Cloudy6500K,
    RgbTransformLightSource_Shade8000K,
    RgbTransformLightSource_Custom,
    NUM_RGBTRANSFORMLIGHTSOURCE }

• enum spinColorTransformationValueSelectorEnums {
    ColorTransformationValueSelector_Gain00,
    ColorTransformationValueSelector_Gain01,
    ColorTransformationValueSelector_Gain02,
    ColorTransformationValueSelector_Gain10,
    ColorTransformationValueSelector_Gain11,
    ColorTransformationValueSelector_Gain12,
    ColorTransformationValueSelector_Gain20,
    ColorTransformationValueSelector_Gain21,
    ColorTransformationValueSelector_Gain22,
    ColorTransformationValueSelector_Offset0,
    ColorTransformationValueSelector_Offset1,
    ColorTransformationValueSelector_Offset2,
    NUM_COLORTRANSFORMATIONVALUESELECTOR }

• enum spinDeviceRegistersEndiannessEnums {
    DeviceRegistersEndianness_Little,
    DeviceRegistersEndianness_Big,
    NUM_DEVICEREGISTERSENDIANNES }

• enum spinDeviceScanTypeEnums {
    DeviceScanType_Areascan,
    NUM_DEVICESCANTYPE }

• enum spinDeviceCharacterSetEnums {
    DeviceCharacterSet_UTF8,
    DeviceCharacterSet_ASCII,
    NUM_DEVICECHARACTERSET }

• enum spinDeviceTLTypeEnums {
    DeviceTLType_GigEVision,

```

```

DeviceTLType_CameraLink,
DeviceTLType_CameraLinkHS,
DeviceTLType_CoaXPress,
DeviceTLType_USB3Vision,
DeviceTLType_Custom,
NUM_DEVICE TLTYPE }

• enum spinDevicePowerSupplySelectorEnums {
DevicePowerSupplySelector_External,
NUM_DEVICEPOWERSUPPLYSELECTOR }

• enum spinDeviceTemperatureSelectorEnums {
DeviceTemperatureSelector_Sensor,
NUM_DEVICE TEMPERATURESELECTOR }

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

• enum spinAutoExposureControlPriorityEnums {
AutoExposureControlPriority_Gain,
AutoExposureControlPriority_ExposureTime,
NUM_AUTOEXPOSURECONTROLPRIORITY }

• enum spinAutoExposureMeteringModeEnums {
AutoExposureMeteringMode_Average,
AutoExposureMeteringMode_Spot,
AutoExposureMeteringMode_Partial,
AutoExposureMeteringMode_CenterWeighted,
AutoExposureMeteringMode_HistogramPeak,
NUM_AUTOEXPOSUREMETERINGMODE }

• enum spinBalanceWhiteAutoProfileEnums {
BalanceWhiteAutoProfile_Indoor,
BalanceWhiteAutoProfile_Outdoor,
NUM_BALANCEWHITEAUTOPROFILE }

• enum spinAutoAlgorithmSelectorEnums {
AutoAlgorithmSelector_Awb,
AutoAlgorithmSelector_Ae,
NUM_AUTOALGORITHMSELECTOR }

• enum spinAutoExposureTargetGreyValueAutoEnums {
AutoExposureTargetGreyValueAuto_Off,
AutoExposureTargetGreyValueAuto_Continuous,
NUM_AUTOEXPOSURETARGETGREYVALUEAUTO }

• enum spinAutoExposureLightingModeEnums {
AutoExposureLightingMode_AutoDetect,
AutoExposureLightingMode_Backlight,
AutoExposureLightingMode_Frontlight,
AutoExposureLightingMode_Normal,
NUM_AUTOEXPOSURELIGHTINGMODE }

• enum spinGevIEEE1588StatusEnums {
GevIEEE1588Status_Initializing,
GevIEEE1588Status_Faulty,
GevIEEE1588Status_Disabled,
GevIEEE1588Status_Listening,
GevIEEE1588Status_PreMaster,
GevIEEE1588Status_Master,
GevIEEE1588Status_Passive,
GevIEEE1588Status_Uncalibrated,
GevIEEE1588Status_Slave,
NUM_GEVIEEE1588STATUS }

```



- enum spinGevIEEE1588ModeEnums {  
GevIEEE1588Mode\_Auto,  
GevIEEE1588Mode\_SlaveOnly,  
NUM\_GEVIEEE1588MODE }
- enum spinGevIEEE1588ClockAccuracyEnums {  
GevIEEE1588ClockAccuracy\_Unknown,  
NUM\_GEVIEEE1588CLOCKACCURACY }
- enum spinGevCCPEnums {  
GevCCP\_OpenAccess,  
GevCCP\_ExclusiveAccess,  
GevCCP\_ControlAccess,  
NUM\_GEVCCP }
- enum spinGevSupportedOptionSelectorEnums {  
GevSupportedOptionSelector\_UserDefinedName,  
GevSupportedOptionSelector\_SerialNumber,  
GevSupportedOptionSelector\_HeartbeatDisable,  
GevSupportedOptionSelector\_LinkSpeed,  
GevSupportedOptionSelector\_CCPApplicationSocket,  
GevSupportedOptionSelector\_ManifestTable,  
GevSupportedOptionSelector\_TestData,  
GevSupportedOptionSelector\_DiscoveryAckDelay,  
GevSupportedOptionSelector\_DiscoveryAckDelayWritable,  
GevSupportedOptionSelector\_ExtendedStatusCodes,  
GevSupportedOptionSelector\_Action,  
GevSupportedOptionSelector\_PendingAck,  
GevSupportedOptionSelector\_EventData,  
GevSupportedOptionSelector\_Event,  
GevSupportedOptionSelector\_PacketResend,  
GevSupportedOptionSelector\_WriteMem,  
GevSupportedOptionSelector\_CommandsConcatenation,  
GevSupportedOptionSelector\_IPConfigurationLLA,  
GevSupportedOptionSelector\_IPConfigurationDHCP,  
GevSupportedOptionSelector\_IPConfigurationPersistentIP,  
GevSupportedOptionSelector\_StreamChannelSourceSocket,  
GevSupportedOptionSelector\_MessageChannelSourceSocket,  
NUM\_GEVSUPPORTEDOPTIONSELECTOR }
- enum spinBlackLevelSelectorEnums {  
BlackLevelSelector\_All,  
BlackLevelSelector\_Analog,  
BlackLevelSelector\_Digital,  
NUM\_BLACKLEVELSELECTOR }
- enum spinBalanceWhiteAutoEnums {  
BalanceWhiteAuto\_Off,  
BalanceWhiteAuto\_Once,  
BalanceWhiteAuto\_Continuous,  
NUM\_BALANCEWHITEAUTO }
- enum spinGainAutoEnums {  
GainAuto\_Off,  
GainAuto\_Once,  
GainAuto\_Continuous,  
NUM\_GAINAUTO }
- enum spinBalanceRatioSelectorEnums {  
BalanceRatioSelector\_Red,  
BalanceRatioSelector\_Blue,  
NUM\_BALANCERATIOSELECTOR }
- enum spinGainSelectorEnums {  
GainSelector\_All,  
NUM\_GAINSELECTOR }

- enum [spinDefectCorrectionModeEnums](#) {  
    [DefectCorrectionMode\\_Average](#),  
    [DefectCorrectionMode\\_Highlight](#),  
    [DefectCorrectionMode\\_Zero](#),  
    [NUM\\_DEFECTCORRECTIONMODE](#) }
- enum [spinUserSetSelectorEnums](#) {  
    [UserSetSelector\\_Default](#),  
    [UserSetSelector\\_UserSet0](#),  
    [UserSetSelector\\_UserSet1](#),  
    [NUM\\_USERSETSELECTOR](#) }
- enum [spinUserSetDefaultEnums](#) {  
    [UserSetDefault\\_Default](#),  
    [UserSetDefault\\_UserSet0](#),  
    [UserSetDefault\\_UserSet1](#),  
    [NUM\\_USERSETDEFAULT](#) }
- enum [spinSerialPortBaudRateEnums](#) {  
    [SerialPortBaudRate\\_Baud300](#),  
    [SerialPortBaudRate\\_Baud600](#),  
    [SerialPortBaudRate\\_Baud1200](#),  
    [SerialPortBaudRate\\_Baud2400](#),  
    [SerialPortBaudRate\\_Baud4800](#),  
    [SerialPortBaudRate\\_Baud9600](#),  
    [SerialPortBaudRate\\_Baud14400](#),  
    [SerialPortBaudRate\\_Baud19200](#),  
    [SerialPortBaudRate\\_Baud38400](#),  
    [SerialPortBaudRate\\_Baud57600](#),  
    [SerialPortBaudRate\\_Baud115200](#),  
    [SerialPortBaudRate\\_Baud230400](#),  
    [SerialPortBaudRate\\_Baud460800](#),  
    [SerialPortBaudRate\\_Baud921600](#),  
    [NUM\\_SERIALPORTBAUDRATE](#) }
- enum [spinSerialPortParityEnums](#) {  
    [SerialPortParity\\_None](#),  
    [SerialPortParity\\_Odd](#),  
    [SerialPortParity\\_Even](#),  
    [SerialPortParity\\_Mark](#),  
    [SerialPortParity\\_Space](#),  
    [NUM\\_SERIALPORTPARITY](#) }
- enum [spinSerialPortSelectorEnums](#) {  
    [SerialPortSelector\\_SerialPort0](#),  
    [NUM\\_SERIALPORTSELECTOR](#) }
- enum [spinSerialPortStopBitsEnums](#) {  
    [SerialPortStopBits\\_Bits1](#),  
    [SerialPortStopBits\\_Bits1AndAHalf](#),  
    [SerialPortStopBits\\_Bits2](#),  
    [NUM\\_SERIALPORTSTOPBITS](#) }
- enum [spinSerialPortSourceEnums](#) {  
    [SerialPortSource\\_Line0](#),  
    [SerialPortSource\\_Line1](#),  
    [SerialPortSource\\_Line2](#),  
    [SerialPortSource\\_Line3](#),  
    [SerialPortSource\\_Off](#),  
    [NUM\\_SERIALPORTSOURCE](#) }
- enum [spinSequencerModeEnums](#) {  
    [SequencerMode\\_Off](#),  
    [SequencerMode\\_On](#),  
    [NUM\\_SEQUENCERMODE](#) }

- enum spinSequencerConfigurationValidEnums {  
SequencerConfigurationValid\_No,  
SequencerConfigurationValid\_Yes,  
NUM\_SEQUENCERCONFIGURATIONVALID }
- enum spinSequencerSetValidEnums {  
SequencerSetValid\_No,  
SequencerSetValid\_Yes,  
NUM\_SEQUENCERSETVALID }
- enum spinSequencerTriggerActivationEnums {  
SequencerTriggerActivation\_RisingEdge,  
SequencerTriggerActivation\_FallingEdge,  
SequencerTriggerActivation\_AnyEdge,  
SequencerTriggerActivation\_LevelHigh,  
SequencerTriggerActivation\_LevelLow,  
NUM\_SEQUENCERTRIGGERACTIVATION }
- enum spinSequencerConfigurationModeEnums {  
SequencerConfigurationMode\_Off,  
SequencerConfigurationMode\_On,  
NUM\_SEQUENCERCONFIGURATIONMODE }
- enum spinSequencerTriggerSourceEnums {  
SequencerTriggerSource\_Off,  
SequencerTriggerSource\_FrameStart,  
NUM\_SEQUENCERTRIGGERSOURCE }
- enum spinTransferQueueModeEnums {  
TransferQueueMode\_FirstInFirstOut,  
NUM\_TRANSFERQUEUEMODE }
- enum spinTransferOperationModeEnums {  
TransferOperationMode\_Continuous,  
TransferOperationMode\_MultiBlock,  
NUM\_TRANSFEROPERATIONMODE }
- enum spinTransferControlModeEnums {  
TransferControlMode\_Basic,  
TransferControlMode\_Automatic,  
TransferControlMode\_UserControlled,  
NUM\_TRANSFERCONTROLMODE }
- enum spinChunkGainSelectorEnums {  
ChunkGainSelector\_All,  
ChunkGainSelector\_Red,  
ChunkGainSelector\_Green,  
ChunkGainSelector\_Blue,  
NUM\_CHUNKGAINSELECTOR }
- enum spinChunkSelectorEnums {  
ChunkSelector\_Image,  
ChunkSelector\_CRC,  
ChunkSelector\_FrameID,  
ChunkSelector\_OffsetX,  
ChunkSelector\_OffsetY,  
ChunkSelector\_Width,  
ChunkSelector\_Height,  
ChunkSelector\_ExposureTime,  
ChunkSelector\_Gain,  
ChunkSelector\_BlackLevel,  
ChunkSelector\_PixelFormat,  
ChunkSelector\_Timestamp,  
ChunkSelector\_SequencerSetActive,  
ChunkSelector\_SerialData,  
ChunkSelector\_ExposureEndLineStatusAll,  
NUM\_CHUNKSELECTOR }

- enum `spinChunkBlackLevelSelectorEnums` {  
`ChunkBlackLevelSelector_All`,  
`NUM_CHUNKBLACKLEVELSELECTOR` }
- enum `spinChunkPixelFormatEnums` {  
`ChunkPixelFormat_Mono8`,  
`ChunkPixelFormat_Mono12Packed`,  
`ChunkPixelFormat_Mono16`,  
`ChunkPixelFormat_RGB8Packed`,  
`ChunkPixelFormat_YUV422Packed`,  
`ChunkPixelFormat_BayerGR8`,  
`ChunkPixelFormat_BayerRG8`,  
`ChunkPixelFormat_BayerGB8`,  
`ChunkPixelFormat_BayerBG8`,  
`ChunkPixelFormat_YCbCr601_422_8_CbYCrY`,  
`NUM_CHUNKPIXELFORMAT` }
- enum `spinFileOperationStatusEnums` {  
`FileOperationStatus_Success`,  
`FileOperationStatus_Failure`,  
`FileOperationStatus_Overflow`,  
`NUM_FILEOPERATIONSTATUS` }
- enum `spinFileOpenModeEnums` {  
`FileOpenMode_Read`,  
`FileOpenMode_Write`,  
`FileOpenMode_ReadWrite`,  
`NUM_FILEOPENMODE` }
- enum `spinFileOperationSelectorEnums` {  
`FileOperationSelector_Open`,  
`FileOperationSelector_Close`,  
`FileOperationSelector_Read`,  
`FileOperationSelector_Write`,  
`FileOperationSelector_Delete`,  
`NUM_FILEOPERATIONSELECTOR` }
- enum `spinFileSelectorEnums` {  
`FileSelector_UserSetDefault`,  
`FileSelector_UserSet0`,  
`FileSelector_UserSet1`,  
`FileSelector_UserFile1`,  
`FileSelector_SerialPort0`,  
`NUM_FILESELECTOR` }
- enum `spinBinningSelectorEnums` {  
`BinningSelector_All`,  
`BinningSelector_Sensor`,  
`BinningSelector_ISP`,  
`NUM_BINNINGSELECTOR` }
- enum `spinTestPatternGeneratorSelectorEnums` {  
`TestPatternGeneratorSelector_Sensor`,  
`TestPatternGeneratorSelector_PipelineStart`,  
`NUM_TESTPATTERNGENERATORSELECTOR` }
- enum `spinTestPatternEnums` {  
`TestPattern_Off`,  
`TestPattern_Increment`,  
`TestPattern_SensorTestPattern`,  
`NUM_TESTPATTERN` }
- enum `spinPixelColorFilterEnums` {  
`PixelColorFilter_None`,  
`PixelColorFilter_BayerRG`,  
`PixelColorFilter_BayerGB`,  
`PixelColorFilter_BayerGR`,

```
PixelColorFilter_BayerBG,  
NUM_PIXELCOLORFILTER }  
• enum spinAdcBitDepthEnums {  
    AdcBitDepth_Bit8,  
    AdcBitDepth_Bit10,  
    AdcBitDepth_Bit12,  
    AdcBitDepth_Bit14,  
    NUM_ADCBITDEPTH }  
• enum spinDecimationHorizontalModeEnums {  
    DecimationHorizontalMode_Discard,  
    NUM_DECIMATIONHORIZONTALMODE }  
• enum spinBinningVerticalModeEnums {  
    BinningVerticalMode_Sum,  
    BinningVerticalMode_Average,  
    NUM_BINNINGVERTICALMODE }  
• enum spinPixelSizeEnums {  
    PixelSize_Bpp1,  
    PixelSize_Bpp2,  
    PixelSize_Bpp4,  
    PixelSize_Bpp8,  
    PixelSize_Bpp10,  
    PixelSize_Bpp12,  
    PixelSize_Bpp14,  
    PixelSize_Bpp16,  
    PixelSize_Bpp20,  
    PixelSize_Bpp24,  
    PixelSize_Bpp30,  
    PixelSize_Bpp32,  
    PixelSize_Bpp36,  
    PixelSize_Bpp48,  
    PixelSize_Bpp64,  
    PixelSize_Bpp96,  
    NUM_PIXELSIZE }  
• enum spinDecimationSelectorEnums {  
    DecimationSelector_All,  
    DecimationSelector_Sensor,  
    NUM_DECIMATIONSELECTOR }  
• enum spinImageCompressionModeEnums {  
    ImageCompressionMode_Off,  
    ImageCompressionMode_Lossless,  
    NUM_IMAGECOMPRESSIONMODE }  
• enum spinBinningHorizontalModeEnums {  
    BinningHorizontalMode_Sum,  
    BinningHorizontalMode_Average,  
    NUM_BINNINGHORIZONTALMODE }  
• enum spinPixelFormatEnums {  
    PixelFormat_Mono8,  
    PixelFormat_Mono16,  
    PixelFormat_RGB8Packed,  
    PixelFormat_BayerGR8,  
    PixelFormat_BayerRG8,  
    PixelFormat_BayerGB8,  
    PixelFormat_BayerBG8,  
    PixelFormat_BayerGR16,  
    PixelFormat_BayerRG16,  
    PixelFormat_BayerGB16,  
    PixelFormat_BayerBG16,  
    PixelFormat_Mono12Packed,
```

[PixelFormat\\_BayerGR12Packed,](#)  
[PixelFormat\\_BayerRG12Packed,](#)  
[PixelFormat\\_BayerGB12Packed,](#)  
[PixelFormat\\_BayerBG12Packed,](#)  
[PixelFormat\\_YUV411Packed,](#)  
[PixelFormat\\_YUV422Packed,](#)  
[PixelFormat\\_YUV444Packed,](#)  
[PixelFormat\\_Mono12p,](#)  
[PixelFormat\\_BayerGR12p,](#)  
[PixelFormat\\_BayerRG12p,](#)  
[PixelFormat\\_BayerGB12p,](#)  
[PixelFormat\\_BayerBG12p,](#)  
[PixelFormat\\_YCbCr8,](#)  
[PixelFormat\\_YCbCr422\\_8,](#)  
[PixelFormat\\_YCbCr411\\_8,](#)  
[PixelFormat\\_BGR8,](#)  
[PixelFormat\\_BGRa8,](#)  
[PixelFormat\\_Mono10Packed,](#)  
[PixelFormat\\_BayerGR10Packed,](#)  
[PixelFormat\\_BayerRG10Packed,](#)  
[PixelFormat\\_BayerGB10Packed,](#)  
[PixelFormat\\_BayerBG10Packed,](#)  
[PixelFormat\\_Mono10p,](#)  
[PixelFormat\\_BayerGR10p,](#)  
[PixelFormat\\_BayerRG10p,](#)  
[PixelFormat\\_BayerGB10p,](#)  
[PixelFormat\\_BayerBG10p,](#)  
[PixelFormat\\_Mono1p,](#)  
[PixelFormat\\_Mono2p,](#)  
[PixelFormat\\_Mono4p,](#)  
[PixelFormat\\_Mono8s,](#)  
[PixelFormat\\_Mono10,](#)  
[PixelFormat\\_Mono12,](#)  
[PixelFormat\\_Mono14,](#)  
[PixelFormat\\_Mono16s,](#)  
[PixelFormat\\_Mono32f,](#)  
[PixelFormat\\_BayerBG10,](#)  
[PixelFormat\\_BayerBG12,](#)  
[PixelFormat\\_BayerGB10,](#)  
[PixelFormat\\_BayerGB12,](#)  
[PixelFormat\\_BayerGR10,](#)  
[PixelFormat\\_BayerGR12,](#)  
[PixelFormat\\_BayerRG10,](#)  
[PixelFormat\\_BayerRG12,](#)  
[PixelFormat\\_RGBa8,](#)  
[PixelFormat\\_RGBa10,](#)  
[PixelFormat\\_RGBa10p,](#)  
[PixelFormat\\_RGBa12,](#)  
[PixelFormat\\_RGBa12p,](#)  
[PixelFormat\\_RGBa14,](#)  
[PixelFormat\\_RGBa16,](#)  
[PixelFormat\\_RGB8,](#)  
[PixelFormat\\_RGB8\\_Planar,](#)  
[PixelFormat\\_RGB10,](#)  
[PixelFormat\\_RGB10\\_Planar,](#)  
[PixelFormat\\_RGB10p,](#)  
[PixelFormat\\_RGB10p32,](#)  
[PixelFormat\\_RGB12,](#)

PixelFormat\_RGB12\_Planar,  
PixelFormat\_RGB12p,  
PixelFormat\_RGB14,  
PixelFormat\_RGB16,  
PixelFormat\_RGB16s,  
PixelFormat\_RGB32f,  
PixelFormat\_RGB16\_Planar,  
PixelFormat\_RGB565p,  
PixelFormat\_BGRa10,  
PixelFormat\_BGRa10p,  
PixelFormat\_BGRa12,  
PixelFormat\_BGRa12p,  
PixelFormat\_BGRa14,  
PixelFormat\_BGRa16,  
PixelFormat\_RGBa32f,  
PixelFormat\_BGR10,  
PixelFormat\_BGR10p,  
PixelFormat\_BGR12,  
PixelFormat\_BGR12p,  
PixelFormat\_BGR14,  
PixelFormat\_BGR16,  
PixelFormat\_BGR565p,  
PixelFormat\_R8,  
PixelFormat\_R10,  
PixelFormat\_R12,  
PixelFormat\_R16,  
PixelFormat\_G8,  
PixelFormat\_G10,  
PixelFormat\_G12,  
PixelFormat\_G16,  
PixelFormat\_B8,  
PixelFormat\_B10,  
PixelFormat\_B12,  
PixelFormat\_B16,  
PixelFormat\_Coord3D\_ABC8,  
PixelFormat\_Coord3D\_ABC8\_Planar,  
PixelFormat\_Coord3D\_ABC10p,  
PixelFormat\_Coord3D\_ABC10p\_Planar,  
PixelFormat\_Coord3D\_ABC12p,  
PixelFormat\_Coord3D\_ABC12p\_Planar,  
PixelFormat\_Coord3D\_ABC16,  
PixelFormat\_Coord3D\_ABC16\_Planar,  
PixelFormat\_Coord3D\_ABC32f,  
PixelFormat\_Coord3D\_ABC32f\_Planar,  
PixelFormat\_Coord3D\_AC8,  
PixelFormat\_Coord3D\_AC8\_Planar,  
PixelFormat\_Coord3D\_AC10p,  
PixelFormat\_Coord3D\_AC10p\_Planar,  
PixelFormat\_Coord3D\_AC12p,  
PixelFormat\_Coord3D\_AC12p\_Planar,  
PixelFormat\_Coord3D\_AC16,  
PixelFormat\_Coord3D\_AC16\_Planar,  
PixelFormat\_Coord3D\_AC32f,  
PixelFormat\_Coord3D\_AC32f\_Planar,  
PixelFormat\_Coord3D\_A8,  
PixelFormat\_Coord3D\_A10p,  
PixelFormat\_Coord3D\_A12p,  
PixelFormat\_Coord3D\_A16,

PixelFormat\_Coord3D\_A32f,  
PixelFormat\_Coord3D\_B8,  
PixelFormat\_Coord3D\_B10p,  
PixelFormat\_Coord3D\_B12p,  
PixelFormat\_Coord3D\_B16,  
PixelFormat\_Coord3D\_B32f,  
PixelFormat\_Coord3D\_C8,  
PixelFormat\_Coord3D\_C10p,  
PixelFormat\_Coord3D\_C12p,  
PixelFormat\_Coord3D\_C16,  
PixelFormat\_Coord3D\_C32f,  
PixelFormat\_Confidence1,  
PixelFormat\_Confidence1p,  
PixelFormat\_Confidence8,  
PixelFormat\_Confidence16,  
PixelFormat\_Confidence32f,  
PixelFormat\_BiColorBGRG8,  
PixelFormat\_BiColorBGRG10,  
PixelFormat\_BiColorBGRG10p,  
PixelFormat\_BiColorBGRG12,  
PixelFormat\_BiColorBGRG12p,  
PixelFormat\_BiColorRGBG8,  
PixelFormat\_BiColorRGBG10,  
PixelFormat\_BiColorRGBG10p,  
PixelFormat\_BiColorRGBG12,  
PixelFormat\_BiColorRGBG12p,  
PixelFormat\_SCF1WBWG8,  
PixelFormat\_SCF1WBWG10,  
PixelFormat\_SCF1WBWG10p,  
PixelFormat\_SCF1WBWG12,  
PixelFormat\_SCF1WBWG12p,  
PixelFormat\_SCF1WBWG14,  
PixelFormat\_SCF1WBWG16,  
PixelFormat\_SCF1WGWB8,  
PixelFormat\_SCF1WGWB10,  
PixelFormat\_SCF1WGWB10p,  
PixelFormat\_SCF1WGWB12,  
PixelFormat\_SCF1WGWB12p,  
PixelFormat\_SCF1WGWB14,  
PixelFormat\_SCF1WGWB16,  
PixelFormat\_SCF1WGWR8,  
PixelFormat\_SCF1WGWR10,  
PixelFormat\_SCF1WGWR10p,  
PixelFormat\_SCF1WGWR12,  
PixelFormat\_SCF1WGWR12p,  
PixelFormat\_SCF1WGWR14,  
PixelFormat\_SCF1WGWR16,  
PixelFormat\_SCF1WRWG8,  
PixelFormat\_SCF1WRWG10,  
PixelFormat\_SCF1WRWG10p,  
PixelFormat\_SCF1WRWG12,  
PixelFormat\_SCF1WRWG12p,  
PixelFormat\_SCF1WRWG14,  
PixelFormat\_SCF1WRWG16,  
PixelFormat\_YCbCr8\_CbYCr,  
PixelFormat\_YCbCr10\_CbYCr,  
PixelFormat\_YCbCr10p\_CbYCr,  
PixelFormat\_YCbCr12\_CbYCr,



PixelFormat\_YCbCr12p\_CbYCr,  
PixelFormat\_YCbCr411\_8\_CbYYCrYY,  
PixelFormat\_YCbCr422\_8\_CbYCrY,  
PixelFormat\_YCbCr422\_10,  
PixelFormat\_YCbCr422\_10\_CbYCrY,  
PixelFormat\_YCbCr422\_10p,  
PixelFormat\_YCbCr422\_10p\_CbYCrY,  
PixelFormat\_YCbCr422\_12,  
PixelFormat\_YCbCr422\_12\_CbYCrY,  
PixelFormat\_YCbCr422\_12p,  
PixelFormat\_YCbCr422\_12p\_CbYCrY,  
PixelFormat\_YCbCr601\_8\_CbYCr,  
PixelFormat\_YCbCr601\_10\_CbYCr,  
PixelFormat\_YCbCr601\_10p\_CbYCr,  
PixelFormat\_YCbCr601\_12\_CbYCr,  
PixelFormat\_YCbCr601\_12p\_CbYCr,  
PixelFormat\_YCbCr601\_411\_8\_CbYYCrYY,  
PixelFormat\_YCbCr601\_422\_8,  
PixelFormat\_YCbCr601\_422\_8\_CbYCrY,  
PixelFormat\_YCbCr601\_422\_10,  
PixelFormat\_YCbCr601\_422\_10\_CbYCrY,  
PixelFormat\_YCbCr601\_422\_10p,  
PixelFormat\_YCbCr601\_422\_10p\_CbYCrY,  
PixelFormat\_YCbCr601\_422\_12,  
PixelFormat\_YCbCr601\_422\_12\_CbYCrY,  
PixelFormat\_YCbCr601\_422\_12p,  
PixelFormat\_YCbCr601\_422\_12p\_CbYCrY,  
PixelFormat\_YCbCr709\_8\_CbYCr,  
PixelFormat\_YCbCr709\_10\_CbYCr,  
PixelFormat\_YCbCr709\_10p\_CbYCr,  
PixelFormat\_YCbCr709\_12\_CbYCr,  
PixelFormat\_YCbCr709\_12p\_CbYCr,  
PixelFormat\_YCbCr709\_411\_8\_CbYYCrYY,  
PixelFormat\_YCbCr709\_422\_8,  
PixelFormat\_YCbCr709\_422\_8\_CbYCrY,  
PixelFormat\_YCbCr709\_422\_10,  
PixelFormat\_YCbCr709\_422\_10\_CbYCrY,  
PixelFormat\_YCbCr709\_422\_10p,  
PixelFormat\_YCbCr709\_422\_10p\_CbYCrY,  
PixelFormat\_YCbCr709\_422\_12,  
PixelFormat\_YCbCr709\_422\_12\_CbYCrY,  
PixelFormat\_YCbCr709\_422\_12p,  
PixelFormat\_YCbCr709\_422\_12p\_CbYCrY,  
PixelFormat\_YUV8\_UYV,  
PixelFormat\_YUV411\_8\_UYYVYY,  
PixelFormat\_YUV422\_8,  
PixelFormat\_YUV422\_8\_UYVY,  
PixelFormat\_Polarized8,  
PixelFormat\_Polarized10p,  
PixelFormat\_Polarized12p,  
PixelFormat\_Polarized16,  
PixelFormat\_BayerRGPolarized8,  
PixelFormat\_BayerRGPolarized10p,  
PixelFormat\_BayerRGPolarized12p,  
PixelFormat\_BayerRGPolarized16,  
PixelFormat\_LLCMono8,  
PixelFormat\_LLCBayerRG8,  
PixelFormat\_JPEGMono8,

```

PixelFormat_JPEGColor8,
PixelFormat_Raw16,
PixelFormat_Raw8,
PixelFormat_R12_Jpeg,
PixelFormat_GR12_Jpeg,
PixelFormat_GB12_Jpeg,
PixelFormat_B12_Jpeg,
UNKNOWN_PIXELFORMAT,
NUM_PIXELFORMAT }

• enum spinDecimationVerticalModeEnums {
    DecimationVerticalMode_Discard,
    NUM_DECIMATIONVERTICALMODE }

• enum spinLineModeEnums {
    LineMode_Input,
    LineMode_Output,
    NUM_LINEMODE }

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

• enum spinLineInputFilterSelectorEnums {
    LineInputFilterSelector_Deglintch,
    LineInputFilterSelector_Debounce,
    NUM_LINEINPUTFILTERSELECTOR }

• enum spinUserOutputSelectorEnums {
    UserOutputSelector_UserOutput0,
    UserOutputSelector_UserOutput1,
    UserOutputSelector_UserOutput2,
    UserOutputSelector_UserOutput3,
    NUM_USEROUTPUTSELECTOR }

• enum spinLineFormatEnums {
    LineFormat_NoConnect,
    LineFormat_TriState,
    LineFormat_TTL,
    LineFormat_LVDS,
    LineFormat_RS422,
    LineFormat_OptoCoupled,
    LineFormat_OpenDrain,
    NUM_LINEFORMAT }

• enum spinLineSelectorEnums {
    LineSelector_Line0,

```

```

LineSelector_Line1,
LineSelector_Line2,
LineSelector_Line3,
NUM_LINESELECTOR }

• enum spinExposureActiveModeEnums {
  ExposureActiveMode_Line1,
  ExposureActiveMode_AnyPixels,
  ExposureActiveMode_AllPixels,
  NUM_EXPOSUREACTIVEMODE }

• enum spinCounterTriggerActivationEnums {
  CounterTriggerActivation_LevelLow,
  CounterTriggerActivation_LevelHigh,
  CounterTriggerActivation_FallingEdge,
  CounterTriggerActivation_RisingEdge,
  CounterTriggerActivation_AnyEdge,
  NUM_COUNTERTRIGGERACTIVATION }

• enum spinCounterSelectorEnums {
  CounterSelector_Counter0,
  CounterSelector_Counter1,
  NUM_COUNTERSELECTOR }

• enum spinCounterStatusEnums {
  CounterStatus_CounterIdle,
  CounterStatus_CounterTriggerWait,
  CounterStatus_CounterActive,
  CounterStatus_CounterCompleted,
  CounterStatus_CounterOverflow,
  NUM_COUNTERSTATUS }

• enum spinCounterTriggerSourceEnums {
  CounterTriggerSource_Off,
  CounterTriggerSource_Line0,
  CounterTriggerSource_Line1,
  CounterTriggerSource_Line2,
  CounterTriggerSource_Line3,
  CounterTriggerSource_UserOutput0,
  CounterTriggerSource_UserOutput1,
  CounterTriggerSource_UserOutput2,
  CounterTriggerSource_UserOutput3,
  CounterTriggerSource_Counter0Start,
  CounterTriggerSource_Counter1Start,
  CounterTriggerSource_Counter0End,
  CounterTriggerSource_Counter1End,
  CounterTriggerSource_LogicBlock0,
  CounterTriggerSource_LogicBlock1,
  CounterTriggerSource_ExposureStart,
  CounterTriggerSource_ExposureEnd,
  CounterTriggerSource_FrameTriggerWait,
  NUM_COUNTERTRIGGERSOURCE }

• enum spinCounterResetSourceEnums {
  CounterResetSource_Off,
  CounterResetSource_Line0,
  CounterResetSource_Line1,
  CounterResetSource_Line2,
  CounterResetSource_Line3,
  CounterResetSource_UserOutput0,
  CounterResetSource_UserOutput1,
  CounterResetSource_UserOutput2,
  CounterResetSource_UserOutput3,
  CounterResetSource_Counter0Start,

```

```

CounterResetSource_Counter1Start,
CounterResetSource_Counter0End,
CounterResetSource_Counter1End,
CounterResetSource_LogicBlock0,
CounterResetSource_LogicBlock1,
CounterResetSource_ExposureStart,
CounterResetSource_ExposureEnd,
CounterResetSource_FrameTriggerWait,
NUM_COUNTERRESETSOURCE }

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

• enum spinCounterEventActivationEnums {
CounterEventActivation_LevelLow,
CounterEventActivation_LevelHigh,
CounterEventActivation_FallingEdge,
CounterEventActivation_RisingEdge,
CounterEventActivation_AnyEdge,
NUM_COUNTEREVENTACTIVATION }

• enum spinCounterResetActivationEnums {
CounterResetActivation_LevelLow,
CounterResetActivation_LevelHigh,
CounterResetActivation_FallingEdge,
CounterResetActivation_RisingEdge,
CounterResetActivation_AnyEdge,
NUM_COUNTERRESETACTIVATION }

• enum spinDeviceTypeEnums {
DeviceType_Transmitter,
DeviceType_Receiver,
DeviceType_Transceiver,
DeviceType_Peripheral,
NUM_DEVICETYPE }

• enum spinDeviceConnectionStatusEnums {
DeviceConnectionStatus_Active,
DeviceConnectionStatus_Inactive,
NUM_DEVICECONNECTIONSTATUS }

• enum spinDeviceLinkThroughputLimitModeEnums {
DeviceLinkThroughputLimitMode_On,
DeviceLinkThroughputLimitMode_Off,
NUM_DEVICELINKTHROUGHPUTLIMITMODE }

```

- enum `spinDeviceLinkHeartbeatModeEnums` {  
    `DeviceLinkHeartbeatMode_On`,  
    `DeviceLinkHeartbeatMode_Off`,  
    `NUM_DEVICELINKHEARTBEATMODE` }
- enum `spinDeviceStreamChannelTypeEnums` {  
    `DeviceStreamChannelType_Transmitter`,  
    `DeviceStreamChannelType_Receiver`,  
    `NUM_DEVICESTREAMCHANNELTYPE` }
- enum `spinDeviceStreamChannelEndiannessEnums` {  
    `DeviceStreamChannelEndianness_Big`,  
    `DeviceStreamChannelEndianness_Little`,  
    `NUM_DEVICESTREAMCHANNELENDIANNESS` }
- enum `spinDeviceClockSelectorEnums` {  
    `DeviceClockSelector_Sensor`,  
    `DeviceClockSelector_SensorDigitization`,  
    `DeviceClockSelector_CameraLink`,  
    `NUM_DEVICECLOCKSELECTOR` }
- enum `spinDeviceSerialPortSelectorEnums` {  
    `DeviceSerialPortSelector_CameraLink`,  
    `NUM_DEVICESERIALPORTSELECTOR` }
- enum `spinDeviceSerialPortBaudRateEnums` {  
    `DeviceSerialPortBaudRate_Baud9600`,  
    `DeviceSerialPortBaudRate_Baud19200`,  
    `DeviceSerialPortBaudRate_Baud38400`,  
    `DeviceSerialPortBaudRate_Baud57600`,  
    `DeviceSerialPortBaudRate_Baud115200`,  
    `DeviceSerialPortBaudRate_Baud230400`,  
    `DeviceSerialPortBaudRate_Baud460800`,  
    `DeviceSerialPortBaudRate_Baud921600`,  
    `NUM_DEVICESERIALPORTBAUDRATE` }
- enum `spinSensorTapsEnums` {  
    `SensorTaps_One`,  
    `SensorTaps_Two`,  
    `SensorTaps_Three`,  
    `SensorTaps_Four`,  
    `SensorTaps_Eight`,  
    `SensorTaps_Ten`,  
    `NUM_SENSORTAPS` }
- enum `spinSensorDigitizationTapsEnums` {  
    `SensorDigitizationTaps_One`,  
    `SensorDigitizationTaps_Two`,  
    `SensorDigitizationTaps_Three`,  
    `SensorDigitizationTaps_Four`,  
    `SensorDigitizationTaps_Eight`,  
    `SensorDigitizationTaps_Ten`,  
    `NUM_SENSORDIGITIZATIONTAPS` }
- enum `spinRegionSelectorEnums` {  
    `RegionSelector_Region0`,  
    `RegionSelector_Region1`,  
    `RegionSelector_Region2`,  
    `RegionSelector_All`,  
    `NUM_REGIONSELECTOR` }
- enum `spinRegionModeEnums` {  
    `RegionMode_Off`,  
    `RegionMode_On`,  
    `NUM_REGIONMODE` }
- enum `spinRegionDestinationEnums` {  
    `RegionDestination_Stream0`,

```

RegionDestination_Stream1,
RegionDestination_Stream2,
NUM_REGIONDESTINATION }

```

- `enum spinImageComponentSelectorEnums {`  
`ImageComponentSelector_Intensity,`  
`ImageComponentSelector_Color,`  
`ImageComponentSelector_Infrared,`  
`ImageComponentSelector_Ultraviolet,`  
`ImageComponentSelector_Range,`  
`ImageComponentSelector_Disparity,`  
`ImageComponentSelector_Confidence,`  
`ImageComponentSelector_Scatter,`  
`NUM_IMAGECOMPONENTSELECTOR }`
- `enum spinPixelFormatInfoSelectorEnums {`  
`PixelFormatInfoSelector_Mono1p,`  
`PixelFormatInfoSelector_Mono2p,`  
`PixelFormatInfoSelector_Mono4p,`  
`PixelFormatInfoSelector_Mono8,`  
`PixelFormatInfoSelector_Mono8s,`  
`PixelFormatInfoSelector_Mono10,`  
`PixelFormatInfoSelector_Mono10p,`  
`PixelFormatInfoSelector_Mono12,`  
`PixelFormatInfoSelector_Mono12p,`  
`PixelFormatInfoSelector_Mono14,`  
`PixelFormatInfoSelector_Mono16,`  
`PixelFormatInfoSelector_Mono16s,`  
`PixelFormatInfoSelector_Mono32f,`  
`PixelFormatInfoSelector_BayerBG8,`  
`PixelFormatInfoSelector_BayerBG10,`  
`PixelFormatInfoSelector_BayerBG10p,`  
`PixelFormatInfoSelector_BayerBG12,`  
`PixelFormatInfoSelector_BayerBG12p,`  
`PixelFormatInfoSelector_BayerBG16,`  
`PixelFormatInfoSelector_BayerGB8,`  
`PixelFormatInfoSelector_BayerGB10,`  
`PixelFormatInfoSelector_BayerGB10p,`  
`PixelFormatInfoSelector_BayerGB12,`  
`PixelFormatInfoSelector_BayerGB12p,`  
`PixelFormatInfoSelector_BayerGB16,`  
`PixelFormatInfoSelector_BayerGR8,`  
`PixelFormatInfoSelector_BayerGR10,`  
`PixelFormatInfoSelector_BayerGR10p,`  
`PixelFormatInfoSelector_BayerGR12,`  
`PixelFormatInfoSelector_BayerGR12p,`  
`PixelFormatInfoSelector_BayerGR16,`  
`PixelFormatInfoSelector_BayerRG8,`  
`PixelFormatInfoSelector_BayerRG10,`  
`PixelFormatInfoSelector_BayerRG10p,`  
`PixelFormatInfoSelector_BayerRG12,`  
`PixelFormatInfoSelector_BayerRG12p,`  
`PixelFormatInfoSelector_BayerRG16,`  
`PixelFormatInfoSelector_RGBa8,`  
`PixelFormatInfoSelector_RGBa10,`  
`PixelFormatInfoSelector_RGBa10p,`  
`PixelFormatInfoSelector_RGBa12,`  
`PixelFormatInfoSelector_RGBa12p,`  
`PixelFormatInfoSelector_RGBa14,`  
`PixelFormatInfoSelector_RGBa16,`

PixelFormatInfoSelector\_RGB8,  
PixelFormatInfoSelector\_RGB8\_Planar,  
PixelFormatInfoSelector\_RGB10,  
PixelFormatInfoSelector\_RGB10\_Planar,  
PixelFormatInfoSelector\_RGB10p,  
PixelFormatInfoSelector\_RGB10p32,  
PixelFormatInfoSelector\_RGB12,  
PixelFormatInfoSelector\_RGB12\_Planar,  
PixelFormatInfoSelector\_RGB12p,  
PixelFormatInfoSelector\_RGB14,  
PixelFormatInfoSelector\_RGB16,  
PixelFormatInfoSelector\_RGB16s,  
PixelFormatInfoSelector\_RGB32f,  
PixelFormatInfoSelector\_RGB16\_Planar,  
PixelFormatInfoSelector\_RGB565p,  
PixelFormatInfoSelector\_BGRa8,  
PixelFormatInfoSelector\_BGRa10,  
PixelFormatInfoSelector\_BGRa10p,  
PixelFormatInfoSelector\_BGRa12,  
PixelFormatInfoSelector\_BGRa12p,  
PixelFormatInfoSelector\_BGRa14,  
PixelFormatInfoSelector\_BGRa16,  
PixelFormatInfoSelector\_RGBa32f,  
PixelFormatInfoSelector\_BGR8,  
PixelFormatInfoSelector\_BGR10,  
PixelFormatInfoSelector\_BGR10p,  
PixelFormatInfoSelector\_BGR12,  
PixelFormatInfoSelector\_BGR12p,  
PixelFormatInfoSelector\_BGR14,  
PixelFormatInfoSelector\_BGR16,  
PixelFormatInfoSelector\_BGR565p,  
PixelFormatInfoSelector\_R8,  
PixelFormatInfoSelector\_R10,  
PixelFormatInfoSelector\_R12,  
PixelFormatInfoSelector\_R16,  
PixelFormatInfoSelector\_G8,  
PixelFormatInfoSelector\_G10,  
PixelFormatInfoSelector\_G12,  
PixelFormatInfoSelector\_G16,  
PixelFormatInfoSelector\_B8,  
PixelFormatInfoSelector\_B10,  
PixelFormatInfoSelector\_B12,  
PixelFormatInfoSelector\_B16,  
PixelFormatInfoSelector\_Coord3D\_ABC8,  
PixelFormatInfoSelector\_Coord3D\_ABC8\_Planar,  
PixelFormatInfoSelector\_Coord3D\_ABC10p,  
PixelFormatInfoSelector\_Coord3D\_ABC10p\_Planar,  
PixelFormatInfoSelector\_Coord3D\_ABC12p,  
PixelFormatInfoSelector\_Coord3D\_ABC12p\_Planar,  
PixelFormatInfoSelector\_Coord3D\_ABC16,  
PixelFormatInfoSelector\_Coord3D\_ABC16\_Planar,  
PixelFormatInfoSelector\_Coord3D\_ABC32f,  
PixelFormatInfoSelector\_Coord3D\_ABC32f\_Planar,  
PixelFormatInfoSelector\_Coord3D\_AC8,  
PixelFormatInfoSelector\_Coord3D\_AC8\_Planar,  
PixelFormatInfoSelector\_Coord3D\_AC10p,  
PixelFormatInfoSelector\_Coord3D\_AC10p\_Planar,  
PixelFormatInfoSelector\_Coord3D\_AC12p,

[PixelFormatInfoSelector\\_Coord3D\\_AC12p\\_Planar](#),  
[PixelFormatInfoSelector\\_Coord3D\\_AC16](#),  
[PixelFormatInfoSelector\\_Coord3D\\_AC16\\_Planar](#),  
[PixelFormatInfoSelector\\_Coord3D\\_AC32f](#),  
[PixelFormatInfoSelector\\_Coord3D\\_AC32f\\_Planar](#),  
[PixelFormatInfoSelector\\_Coord3D\\_A8](#),  
[PixelFormatInfoSelector\\_Coord3D\\_A10p](#),  
[PixelFormatInfoSelector\\_Coord3D\\_A12p](#),  
[PixelFormatInfoSelector\\_Coord3D\\_A16](#),  
[PixelFormatInfoSelector\\_Coord3D\\_A32f](#),  
[PixelFormatInfoSelector\\_Coord3D\\_B8](#),  
[PixelFormatInfoSelector\\_Coord3D\\_B10p](#),  
[PixelFormatInfoSelector\\_Coord3D\\_B12p](#),  
[PixelFormatInfoSelector\\_Coord3D\\_B16](#),  
[PixelFormatInfoSelector\\_Coord3D\\_B32f](#),  
[PixelFormatInfoSelector\\_Coord3D\\_C8](#),  
[PixelFormatInfoSelector\\_Coord3D\\_C10p](#),  
[PixelFormatInfoSelector\\_Coord3D\\_C12p](#),  
[PixelFormatInfoSelector\\_Coord3D\\_C16](#),  
[PixelFormatInfoSelector\\_Coord3D\\_C32f](#),  
[PixelFormatInfoSelector\\_Confidence1](#),  
[PixelFormatInfoSelector\\_Confidence1p](#),  
[PixelFormatInfoSelector\\_Confidence8](#),  
[PixelFormatInfoSelector\\_Confidence16](#),  
[PixelFormatInfoSelector\\_Confidence32f](#),  
[PixelFormatInfoSelector\\_BiColorBGRG8](#),  
[PixelFormatInfoSelector\\_BiColorBGRG10](#),  
[PixelFormatInfoSelector\\_BiColorBGRG10p](#),  
[PixelFormatInfoSelector\\_BiColorBGRG12](#),  
[PixelFormatInfoSelector\\_BiColorBGRG12p](#),  
[PixelFormatInfoSelector\\_BiColorRGBG8](#),  
[PixelFormatInfoSelector\\_BiColorRGBG10](#),  
[PixelFormatInfoSelector\\_BiColorRGBG10p](#),  
[PixelFormatInfoSelector\\_BiColorRGBG12](#),  
[PixelFormatInfoSelector\\_BiColorRGBG12p](#),  
[PixelFormatInfoSelector\\_SCF1WBWG8](#),  
[PixelFormatInfoSelector\\_SCF1WBWG10](#),  
[PixelFormatInfoSelector\\_SCF1WBWG10p](#),  
[PixelFormatInfoSelector\\_SCF1WBWG12](#),  
[PixelFormatInfoSelector\\_SCF1WBWG12p](#),  
[PixelFormatInfoSelector\\_SCF1WBWG14](#),  
[PixelFormatInfoSelector\\_SCF1WBWG16](#),  
[PixelFormatInfoSelector\\_SCF1WGWB8](#),  
[PixelFormatInfoSelector\\_SCF1WGWB10](#),  
[PixelFormatInfoSelector\\_SCF1WGWB10p](#),  
[PixelFormatInfoSelector\\_SCF1WGWB12](#),  
[PixelFormatInfoSelector\\_SCF1WGWB12p](#),  
[PixelFormatInfoSelector\\_SCF1WGWB14](#),  
[PixelFormatInfoSelector\\_SCF1WGWB16](#),  
[PixelFormatInfoSelector\\_SCF1WGWR8](#),  
[PixelFormatInfoSelector\\_SCF1WGWR10](#),  
[PixelFormatInfoSelector\\_SCF1WGWR10p](#),  
[PixelFormatInfoSelector\\_SCF1WGWR12](#),  
[PixelFormatInfoSelector\\_SCF1WGWR12p](#),  
[PixelFormatInfoSelector\\_SCF1WGWR14](#),  
[PixelFormatInfoSelector\\_SCF1WGWR16](#),  
[PixelFormatInfoSelector\\_SCF1WRWG8](#),  
[PixelFormatInfoSelector\\_SCF1WRWG10](#),



PixelFormatInfoSelector\_SCF1WRWG10p,  
PixelFormatInfoSelector\_SCF1WRWG12,  
PixelFormatInfoSelector\_SCF1WRWG12p,  
PixelFormatInfoSelector\_SCF1WRWG14,  
PixelFormatInfoSelector\_SCF1WRWG16,  
PixelFormatInfoSelector\_YCbCr8,  
PixelFormatInfoSelector\_YCbCr8\_CbYCr,  
PixelFormatInfoSelector\_YCbCr10\_CbYCr,  
PixelFormatInfoSelector\_YCbCr10p\_CbYCr,  
PixelFormatInfoSelector\_YCbCr12\_CbYCr,  
PixelFormatInfoSelector\_YCbCr12p\_CbYCr,  
PixelFormatInfoSelector\_YCbCr411\_8,  
PixelFormatInfoSelector\_YCbCr411\_8\_CbYYCrYY,  
PixelFormatInfoSelector\_YCbCr422\_8,  
PixelFormatInfoSelector\_YCbCr422\_8\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr422\_10,  
PixelFormatInfoSelector\_YCbCr422\_10\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr422\_10p,  
PixelFormatInfoSelector\_YCbCr422\_10p\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr422\_12,  
PixelFormatInfoSelector\_YCbCr422\_12\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr422\_12p,  
PixelFormatInfoSelector\_YCbCr422\_12p\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr601\_8\_CbYCr,  
PixelFormatInfoSelector\_YCbCr601\_10\_CbYCr,  
PixelFormatInfoSelector\_YCbCr601\_10p\_CbYCr,  
PixelFormatInfoSelector\_YCbCr601\_12\_CbYCr,  
PixelFormatInfoSelector\_YCbCr601\_12p\_CbYCr,  
PixelFormatInfoSelector\_YCbCr601\_411\_8\_CbYYCrYY,  
PixelFormatInfoSelector\_YCbCr601\_422\_8,  
PixelFormatInfoSelector\_YCbCr601\_422\_8\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr601\_422\_10,  
PixelFormatInfoSelector\_YCbCr601\_422\_10\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr601\_422\_10p,  
PixelFormatInfoSelector\_YCbCr601\_422\_10p\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr601\_422\_12,  
PixelFormatInfoSelector\_YCbCr601\_422\_12\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr601\_422\_12p,  
PixelFormatInfoSelector\_YCbCr601\_422\_12p\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr709\_8\_CbYCr,  
PixelFormatInfoSelector\_YCbCr709\_10\_CbYCr,  
PixelFormatInfoSelector\_YCbCr709\_10p\_CbYCr,  
PixelFormatInfoSelector\_YCbCr709\_12\_CbYCr,  
PixelFormatInfoSelector\_YCbCr709\_12p\_CbYCr,  
PixelFormatInfoSelector\_YCbCr709\_411\_8\_CbYYCrYY,  
PixelFormatInfoSelector\_YCbCr709\_422\_8,  
PixelFormatInfoSelector\_YCbCr709\_422\_8\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr709\_422\_10,  
PixelFormatInfoSelector\_YCbCr709\_422\_10\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr709\_422\_10p,  
PixelFormatInfoSelector\_YCbCr709\_422\_10p\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr709\_422\_12,  
PixelFormatInfoSelector\_YCbCr709\_422\_12\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr709\_422\_12p,  
PixelFormatInfoSelector\_YCbCr709\_422\_12p\_CbYCrY,  
PixelFormatInfoSelector\_YUV8\_UYV,  
PixelFormatInfoSelector\_YUV411\_8\_UYYVYY,  
PixelFormatInfoSelector\_YUV422\_8,

```

PixelFormatInfoSelector_YUV422_8_UYVY,
PixelFormatInfoSelector_Polarized8,
PixelFormatInfoSelector_Polarized10p,
PixelFormatInfoSelector_Polarized12p,
PixelFormatInfoSelector_Polarized16,
PixelFormatInfoSelector_BayerRGPolarized8,
PixelFormatInfoSelector_BayerRGPolarized10p,
PixelFormatInfoSelector_BayerRGPolarized12p,
PixelFormatInfoSelector_BayerRGPolarized16,
PixelFormatInfoSelector_LLCMono8,
PixelFormatInfoSelector_LLCBayerRG8,
PixelFormatInfoSelector_JPEGMono8,
PixelFormatInfoSelector_JPEGColor8,
NUM_PIXELFORMATINFOSELECTOR }

• enum spinDeinterlacingEnums {
    Deinterlacing_Off,
    Deinterlacing_LineDuplication,
    Deinterlacing_Weave,
    NUM_DEINTERLACING }

• enum spinImageCompressionRateOptionEnums {
    ImageCompressionRateOption_FixBitrate,
    ImageCompressionRateOption_FixQuality,
    NUM_IMAGECOMPRESSIONRATEOPTION }

• enum spinImageCompressionJPEGFormatOptionEnums {
    ImageCompressionJPEGFormatOption_Lossless,
    ImageCompressionJPEGFormatOption_BaselineStandard,
    ImageCompressionJPEGFormatOption_BaselineOptimized,
    ImageCompressionJPEGFormatOption_Progressive,
    NUM_IMAGECOMPRESSIONJPEGFORMATOPTION }

• enum spinAcquisitionStatusSelectorEnums {
    AcquisitionStatusSelector_AcquisitionTriggerWait,
    AcquisitionStatusSelector_AcquisitionActive,
    AcquisitionStatusSelector_AcquisitionTransfer,
    AcquisitionStatusSelector_FrameTriggerWait,
    AcquisitionStatusSelector_FrameActive,
    AcquisitionStatusSelector_ExposureActive,
    NUM_ACQUISITIONSTATUSSELECTOR }

• enum spinExposureTimeModeEnums {
    ExposureTimeMode_Common,
    ExposureTimeMode_Individual,
    NUM_EXPOSURETIMEMODE }

• enum spinExposureTimeSelectorEnums {
    ExposureTimeSelector_Common,
    ExposureTimeSelector_Red,
    ExposureTimeSelector_Green,
    ExposureTimeSelector_Blue,
    ExposureTimeSelector_Cyan,
    ExposureTimeSelector_Magenta,
    ExposureTimeSelector_Yellow,
    ExposureTimeSelector_Infrared,
    ExposureTimeSelector_Ultraviolet,
    ExposureTimeSelector_Stage1,
    ExposureTimeSelector_Stage2,
    NUM_EXPOSURETIMESELECTOR }

• enum spinGainAutoBalanceEnums {
    GainAutoBalance_Off,
    GainAutoBalance_Once,

```

```
GainAutoBalance_Continuous,
NUM_GAINAUTOBALANCE }
• enum spinBlackLevelAutoEnums {
    BlackLevelAuto_Off,
    BlackLevelAuto_Once,
    BlackLevelAuto_Continuous,
    NUM_BLACKLEVELAUTO }
• enum spinBlackLevelAutoBalanceEnums {
    BlackLevelAutoBalance_Off,
    BlackLevelAutoBalance_Once,
    BlackLevelAutoBalance_Continuous,
    NUM_BLACKLEVELAUTOBALANCE }
• enum spinWhiteClipSelectorEnums {
    WhiteClipSelector_All,
    WhiteClipSelector_Red,
    WhiteClipSelector_Green,
    WhiteClipSelector_Blue,
    WhiteClipSelector_Y,
    WhiteClipSelector_U,
    WhiteClipSelector_V,
    WhiteClipSelector_Tap1,
    WhiteClipSelector_Tap2,
    NUM_WHITECLIPSELECTOR }
• enum spinTimerSelectorEnums {
    TimerSelector_Timer0,
    TimerSelector_Timer1,
    TimerSelector_Timer2,
    NUM_TIMERSELECTOR }
• enum spinTimerStatusEnums {
    TimerStatus_TimerIdle,
    TimerStatus_TimerTriggerWait,
    TimerStatus_TimerActive,
    TimerStatus_TimerCompleted,
    NUM_TIMERSTATUS }
• enum spinTimerTriggerSourceEnums {
    TimerTriggerSource_Off,
    TimerTriggerSource_AcquisitionTrigger,
    TimerTriggerSource_AcquisitionStart,
    TimerTriggerSource_AcquisitionEnd,
    TimerTriggerSource_FrameTrigger,
    TimerTriggerSource_FrameStart,
    TimerTriggerSource_FrameEnd,
    TimerTriggerSource_FrameBurstStart,
    TimerTriggerSource_FrameBurstEnd,
    TimerTriggerSource_LineTrigger,
    TimerTriggerSource_LineStart,
    TimerTriggerSource_LineEnd,
    TimerTriggerSource_ExposureStart,
    TimerTriggerSource_ExposureEnd,
    TimerTriggerSource_Line0,
    TimerTriggerSource_Line1,
    TimerTriggerSource_Line2,
    TimerTriggerSource_UserOutput0,
    TimerTriggerSource_UserOutput1,
    TimerTriggerSource_UserOutput2,
    TimerTriggerSource_Counter0Start,
    TimerTriggerSource_Counter1Start,
    TimerTriggerSource_Counter2Start,
```

```

TimerTriggerSource_Counter0End,
TimerTriggerSource_Counter1End,
TimerTriggerSource_Counter2End,
TimerTriggerSource_Timer0Start,
TimerTriggerSource_Timer1Start,
TimerTriggerSource_Timer2Start,
TimerTriggerSource_Timer0End,
TimerTriggerSource_Timer1End,
TimerTriggerSource_Timer2End,
TimerTriggerSource_Encoder0,
TimerTriggerSource_Encoder1,
TimerTriggerSource_Encoder2,
TimerTriggerSource_SoftwareSignal0,
TimerTriggerSource_SoftwareSignal1,
TimerTriggerSource_SoftwareSignal2,
TimerTriggerSource_Action0,
TimerTriggerSource_Action1,
TimerTriggerSource_Action2,
TimerTriggerSource_LinkTrigger0,
TimerTriggerSource_LinkTrigger1,
TimerTriggerSource_LinkTrigger2,
NUM_TIMERTRIGGERSOURCE }

• enum spinTimerTriggerActivationEnums {
    TimerTriggerActivation_RisingEdge,
    TimerTriggerActivation_FallingEdge,
    TimerTriggerActivation_AnyEdge,
    TimerTriggerActivation_LevelHigh,
    TimerTriggerActivation_LevelLow,
    NUM_TIMERTRIGGERACTIVATION }

• enum spinEncoderSelectorEnums {
    EncoderSelector_Encoder0,
    EncoderSelector_Encoder1,
    EncoderSelector_Encoder2,
    NUM_ENCODERSELECTOR }

• enum spinEncoderSourceAEnums {
    EncoderSourceA_Off,
    EncoderSourceA_Line0,
    EncoderSourceA_Line1,
    EncoderSourceA_Line2,
    NUM_ENCODERSOURCEA }

• enum spinEncoderSourceBEnums {
    EncoderSourceB_Off,
    EncoderSourceB_Line0,
    EncoderSourceB_Line1,
    EncoderSourceB_Line2,
    NUM_ENCODERSOURCEB }

• enum spinEncoderModeEnums {
    EncoderMode_FourPhase,
    EncoderMode_HighResolution,
    NUM_ENCODERMODE }

• enum spinEncoderOutputModeEnums {
    EncoderOutputMode_Off,
    EncoderOutputMode_PositionUp,
    EncoderOutputMode_PositionDown,
    EncoderOutputMode_DirectionUp,
    EncoderOutputMode_DirectionDown,
    EncoderOutputMode_Motion,
    NUM_ENCODEROUTPUTMODE }

```

- enum `spinEncoderStatusEnums` {  
    `EncoderStatus_EncoderUp`,  
    `EncoderStatus_EncoderDown`,  
    `EncoderStatus_EncoderIdle`,  
    `EncoderStatus_EncoderStatic`,  
    `NUM_ENCODERSTATUS` }
- enum `spinEncoderResetSourceEnums` {  
    `EncoderResetSource_Off`,  
    `EncoderResetSource_AcquisitionTrigger`,  
    `EncoderResetSource_AcquisitionStart`,  
    `EncoderResetSource_AcquisitionEnd`,  
    `EncoderResetSource_FrameTrigger`,  
    `EncoderResetSource_FrameStart`,  
    `EncoderResetSource_FrameEnd`,  
    `EncoderResetSource_ExposureStart`,  
    `EncoderResetSource_ExposureEnd`,  
    `EncoderResetSource_Line0`,  
    `EncoderResetSource_Line1`,  
    `EncoderResetSource_Line2`,  
    `EncoderResetSource_Counter0Start`,  
    `EncoderResetSource_Counter1Start`,  
    `EncoderResetSource_Counter2Start`,  
    `EncoderResetSource_Counter0End`,  
    `EncoderResetSource_Counter1End`,  
    `EncoderResetSource_Counter2End`,  
    `EncoderResetSource_Timer0Start`,  
    `EncoderResetSource_Timer1Start`,  
    `EncoderResetSource_Timer2Start`,  
    `EncoderResetSource_Timer0End`,  
    `EncoderResetSource_Timer1End`,  
    `EncoderResetSource_Timer2End`,  
    `EncoderResetSource_UserOutput0`,  
    `EncoderResetSource_UserOutput1`,  
    `EncoderResetSource_UserOutput2`,  
    `EncoderResetSource_SoftwareSignal0`,  
    `EncoderResetSource_SoftwareSignal1`,  
    `EncoderResetSource_SoftwareSignal2`,  
    `EncoderResetSource_Action0`,  
    `EncoderResetSource_Action1`,  
    `EncoderResetSource_Action2`,  
    `EncoderResetSource_LinkTrigger0`,  
    `EncoderResetSource_LinkTrigger1`,  
    `EncoderResetSource_LinkTrigger2`,  
    `NUM_ENCODERRESETSOURCE` }
- enum `spinEncoderResetActivationEnums` {  
    `EncoderResetActivation_RisingEdge`,  
    `EncoderResetActivation_FallingEdge`,  
    `EncoderResetActivation_AnyEdge`,  
    `EncoderResetActivation_LevelHigh`,  
    `EncoderResetActivation_LevelLow`,  
    `NUM_ENCODERRESETACTIVATION` }
- enum `spinSoftwareSignalSelectorEnums` {  
    `SoftwareSignalSelector_SoftwareSignal0`,  
    `SoftwareSignalSelector_SoftwareSignal1`,  
    `SoftwareSignalSelector_SoftwareSignal2`,  
    `NUM_SOFTWARESIGNALSELECTOR` }
- enum `spinActionUnconditionalModeEnums` {  
    `ActionUnconditionalMode_Off`,

```

    ActionUnconditionalMode_On,
    NUM_ACTIONUNCONDITIONALMODE }
• enum spinSourceSelectorEnums {
    SourceSelector_Source0,
    SourceSelector_Source1,
    SourceSelector_Source2,
    SourceSelector_All,
    NUM_SOURCESELECTOR }
• enum spinTransferSelectorEnums {
    TransferSelector_Stream0,
    TransferSelector_Stream1,
    TransferSelector_Stream2,
    TransferSelector_All,
    NUM_TRANSFERSELECTOR }
• enum spinTransferTriggerSelectorEnums {
    TransferTriggerSelector_TransferStart,
    TransferTriggerSelector_TransferStop,
    TransferTriggerSelector_TransferAbort,
    TransferTriggerSelector_TransferPause,
    TransferTriggerSelector_TransferResume,
    TransferTriggerSelector_TransferActive,
    TransferTriggerSelector_TransferBurstStart,
    TransferTriggerSelector_TransferBurstStop,
    NUM_TRANSFERTRIGGERSELECTOR }
• enum spinTransferTriggerModeEnums {
    TransferTriggerMode_Off,
    TransferTriggerMode_On,
    NUM_TRANSFERTRIGGERMODE }
• enum spinTransferTriggerSourceEnums {
    TransferTriggerSource_Line0,
    TransferTriggerSource_Line1,
    TransferTriggerSource_Line2,
    TransferTriggerSource_Counter0Start,
    TransferTriggerSource_Counter1Start,
    TransferTriggerSource_Counter2Start,
    TransferTriggerSource_Counter0End,
    TransferTriggerSource_Counter1End,
    TransferTriggerSource_Counter2End,
    TransferTriggerSource_Timer0Start,
    TransferTriggerSource_Timer1Start,
    TransferTriggerSource_Timer2Start,
    TransferTriggerSource_Timer0End,
    TransferTriggerSource_Timer1End,
    TransferTriggerSource_Timer2End,
    TransferTriggerSource_SoftwareSignal0,
    TransferTriggerSource_SoftwareSignal1,
    TransferTriggerSource_SoftwareSignal2,
    TransferTriggerSource_Action0,
    TransferTriggerSource_Action1,
    TransferTriggerSource_Action2,
    NUM_TRANSFERTRIGGERSOURCE }
• enum spinTransferTriggerActivationEnums {
    TransferTriggerActivation_RisingEdge,
    TransferTriggerActivation_FallingEdge,
    TransferTriggerActivation_AnyEdge,
    TransferTriggerActivation_LevelHigh,
    TransferTriggerActivation_LevelLow,
    NUM_TRANSFERTRIGGERACTIVATION }

```

- enum spinTransferStatusSelectorEnums {  
TransferStatusSelector\_Streaming,  
TransferStatusSelector\_Paused,  
TransferStatusSelector\_Stopping,  
TransferStatusSelector\_Stopped,  
TransferStatusSelector\_QueueOverflow,  
NUM\_TRANSFERSTATUSSELECTOR }
- enum spinTransferComponentSelectorEnums {  
TransferComponentSelector\_Red,  
TransferComponentSelector\_Green,  
TransferComponentSelector\_Blue,  
TransferComponentSelector\_All,  
NUM\_TRANSFERCOMPONENTSELECTOR }
- enum spinScan3dDistanceUnitEnums {  
Scan3dDistanceUnit\_Millimeter,  
Scan3dDistanceUnit\_Inch,  
NUM\_SCAN3DDISTANCEUNIT }
- enum spinScan3dCoordinateSystemEnums {  
Scan3dCoordinateSystem\_Cartesian,  
Scan3dCoordinateSystem\_Spherical,  
Scan3dCoordinateSystem\_Cylindrical,  
NUM\_SCAN3DCOORDINATESYSTEM }
- enum spinScan3dOutputModeEnums {  
Scan3dOutputMode\_UncalibratedC,  
Scan3dOutputMode\_CalibratedABC\_Grid,  
Scan3dOutputMode\_CalibratedABC\_PointCloud,  
Scan3dOutputMode\_CalibratedAC,  
Scan3dOutputMode\_CalibratedAC\_Linescan,  
Scan3dOutputMode\_CalibratedC,  
Scan3dOutputMode\_CalibratedC\_Linescan,  
Scan3dOutputMode\_RectifiedC,  
Scan3dOutputMode\_RectifiedC\_Linescan,  
Scan3dOutputMode\_DisparityC,  
Scan3dOutputMode\_DisparityC\_Linescan,  
NUM\_SCAN3DOUTPUTMODE }
- enum spinScan3dCoordinateSystemReferenceEnums {  
Scan3dCoordinateSystemReference\_Anchor,  
Scan3dCoordinateSystemReference\_Transformed,  
NUM\_SCAN3DCOORDINATESYSTEMREFERENCE }
- enum spinScan3dCoordinateSelectorEnums {  
Scan3dCoordinateSelector\_CoordinateA,  
Scan3dCoordinateSelector\_CoordinateB,  
Scan3dCoordinateSelector\_CoordinateC,  
NUM\_SCAN3DCOORDINATESELECTOR }
- enum spinScan3dCoordinateTransformSelectorEnums {  
Scan3dCoordinateTransformSelector\_RotationX,  
Scan3dCoordinateTransformSelector\_RotationY,  
Scan3dCoordinateTransformSelector\_RotationZ,  
Scan3dCoordinateTransformSelector\_TranslationX,  
Scan3dCoordinateTransformSelector\_TranslationY,  
Scan3dCoordinateTransformSelector\_TranslationZ,  
NUM\_SCAN3DCOORDINATETRANSFORMSELECTOR }
- enum spinScan3dCoordinateReferenceSelectorEnums {  
Scan3dCoordinateReferenceSelector\_RotationX,  
Scan3dCoordinateReferenceSelector\_RotationY,  
Scan3dCoordinateReferenceSelector\_RotationZ,  
Scan3dCoordinateReferenceSelector\_TranslationX,  
Scan3dCoordinateReferenceSelector\_TranslationY,

```

Scan3dCoordinateReferenceSelector_TranslationZ,
NUM_SCAN3DCOORDINATEREFERENCESELECTOR }
• enum spinChunkImageComponentEnums {
    ChunkImageComponent_Intensity,
    ChunkImageComponent_Color,
    ChunkImageComponent_Infrared,
    ChunkImageComponent_Ultraviolet,
    ChunkImageComponent_Range,
    ChunkImageComponent_Disparity,
    ChunkImageComponent_Confidence,
    ChunkImageComponent_Scatter,
    NUM_CHUNKIMAGECOMPONENT }
• enum spinChunkCounterSelectorEnums {
    ChunkCounterSelector_Counter0,
    ChunkCounterSelector_Counter1,
    ChunkCounterSelector_Counter2,
    NUM_CHUNKCOUNTERSELECTOR }
• enum spinChunkTimerSelectorEnums {
    ChunkTimerSelector_Timer0,
    ChunkTimerSelector_Timer1,
    ChunkTimerSelector_Timer2,
    NUM_CHUNKTIMERSELECTOR }
• enum spinChunkEncoderSelectorEnums {
    ChunkEncoderSelector_Encoder0,
    ChunkEncoderSelector_Encoder1,
    ChunkEncoderSelector_Encoder2,
    NUM_CHUNKENCODERSELECTOR }
• enum spinChunkEncoderStatusEnums {
    ChunkEncoderStatus_EncoderUp,
    ChunkEncoderStatus_EncoderDown,
    ChunkEncoderStatus_EncoderIdle,
    ChunkEncoderStatus_EncoderStatic,
    NUM_CHUNKENCODERSTATUS }
• enum spinChunkExposureTimeSelectorEnums {
    ChunkExposureTimeSelector_Common,
    ChunkExposureTimeSelector_Red,
    ChunkExposureTimeSelector_Green,
    ChunkExposureTimeSelector_Blue,
    ChunkExposureTimeSelector_Cyan,
    ChunkExposureTimeSelector_Magenta,
    ChunkExposureTimeSelector_Yellow,
    ChunkExposureTimeSelector_Infrared,
    ChunkExposureTimeSelector_Ultraviolet,
    ChunkExposureTimeSelector_Stage1,
    ChunkExposureTimeSelector_Stage2,
    NUM_CHUNKEXPOSURETIMESELECTOR }
• enum spinChunkSourceIDEnums {
    ChunkSourceID_Source0,
    ChunkSourceID_Source1,
    ChunkSourceID_Source2,
    NUM_CHUNKSOURCEID }
• enum spinChunkRegionIDEnums {
    ChunkRegionID_Region0,
    ChunkRegionID_Region1,
    ChunkRegionID_Region2,
    NUM_CHUNKREGIONID }
• enum spinChunkTransferStreamIDEnums {
    ChunkTransferStreamID_Stream0,

```



```

    ChunkTransferStreamID_Stream1,
    ChunkTransferStreamID_Stream2,
    ChunkTransferStreamID_Stream3,
    NUM_CHUNKTRANSFERSTREAMID }
• enum spinChunkScan3dDistanceUnitEnums {
    ChunkScan3dDistanceUnit_Millimeter,
    ChunkScan3dDistanceUnit_Inch,
    NUM_CHUNKSCAN3DDISTANCEUNIT }
• enum spinChunkScan3dOutputModeEnums {
    ChunkScan3dOutputMode_UncalibratedC,
    ChunkScan3dOutputMode_CalibratedABC_Grid,
    ChunkScan3dOutputMode_CalibratedABC_PointCloud,
    ChunkScan3dOutputMode_CalibratedAC,
    ChunkScan3dOutputMode_CalibratedAC_Linescan,
    ChunkScan3dOutputMode_CalibratedC,
    ChunkScan3dOutputMode_CalibratedC_Linescan,
    ChunkScan3dOutputMode_RectifiedC,
    ChunkScan3dOutputMode_RectifiedC_Linescan,
    ChunkScan3dOutputMode_DisparityC,
    ChunkScan3dOutputMode_DisparityC_Linescan,
    NUM_CHUNKSCAN3DOUTPUTMODE }
• enum spinChunkScan3dCoordinateSystemEnums {
    ChunkScan3dCoordinateSystem_Cartesian,
    ChunkScan3dCoordinateSystem_Spherical,
    ChunkScan3dCoordinateSystem_Cylindrical,
    NUM_CHUNKSCAN3DCOORDINATESYSTEM }
• enum spinChunkScan3dCoordinateSystemReferenceEnums {
    ChunkScan3dCoordinateSystemReference_Anchor,
    ChunkScan3dCoordinateSystemReference_Transformed,
    NUM_CHUNKSCAN3DCOORDINATESYSTEMREFERENCE }
• enum spinChunkScan3dCoordinateSelectorEnums {
    ChunkScan3dCoordinateSelector_CoordinateA,
    ChunkScan3dCoordinateSelector_CoordinateB,
    ChunkScan3dCoordinateSelector_CoordinateC,
    NUM_CHUNKSCAN3DCOORDINATESELECTOR }
• enum spinChunkScan3dCoordinateTransformSelectorEnums {
    ChunkScan3dCoordinateTransformSelector_RotationX,
    ChunkScan3dCoordinateTransformSelector_RotationY,
    ChunkScan3dCoordinateTransformSelector_RotationZ,
    ChunkScan3dCoordinateTransformSelector_TranslationX,
    ChunkScan3dCoordinateTransformSelector_TranslationY,
    ChunkScan3dCoordinateTransformSelector_TranslationZ,
    NUM_CHUNKSCAN3DCOORDINATETRANSFORMSELECTOR }
• enum spinChunkScan3dCoordinateReferenceSelectorEnums {
    ChunkScan3dCoordinateReferenceSelector_RotationX,
    ChunkScan3dCoordinateReferenceSelector_RotationY,
    ChunkScan3dCoordinateReferenceSelector_RotationZ,
    ChunkScan3dCoordinateReferenceSelector_TranslationX,
    ChunkScan3dCoordinateReferenceSelector_TranslationY,
    ChunkScan3dCoordinateReferenceSelector_TranslationZ,
    NUM_CHUNKSCAN3DCOORDINATEREFERENCESELECTOR }
• enum spinDeviceTapGeometryEnums {
    DeviceTapGeometry_Geometry_1X_1Y,
    DeviceTapGeometry_Geometry_1X2_1Y,
    DeviceTapGeometry_Geometry_1X2_1Y2,
    DeviceTapGeometry_Geometry_2X_1Y,
    DeviceTapGeometry_Geometry_2X_1Y2Geometry_2XE_1Y,
    DeviceTapGeometry_Geometry_2XE_1Y2,

```

```

DeviceTapGeometry_Geometry_2XM_1Y,
DeviceTapGeometry_Geometry_2XM_1Y2,
DeviceTapGeometry_Geometry_1X_1Y2,
DeviceTapGeometry_Geometry_1X_2YE,
DeviceTapGeometry_Geometry_1X3_1Y,
DeviceTapGeometry_Geometry_3X_1Y,
DeviceTapGeometry_Geometry_1X,
DeviceTapGeometry_Geometry_1X2,
DeviceTapGeometry_Geometry_2X,
DeviceTapGeometry_Geometry_2XE,
DeviceTapGeometry_Geometry_2XM,
DeviceTapGeometry_Geometry_1X3,
DeviceTapGeometry_Geometry_3X,
DeviceTapGeometry_Geometry_1X4_1Y,
DeviceTapGeometry_Geometry_4X_1Y,
DeviceTapGeometry_Geometry_2X2_1Y,
DeviceTapGeometry_Geometry_2X2E_1YGeometry_2X2M_1Y,
DeviceTapGeometry_Geometry_1X2_2YE,
DeviceTapGeometry_Geometry_2X_2YE,
DeviceTapGeometry_Geometry_2XE_2YE,
DeviceTapGeometry_Geometry_2XM_2YE,
DeviceTapGeometry_Geometry_1X4,
DeviceTapGeometry_Geometry_4X,
DeviceTapGeometry_Geometry_2X2,
DeviceTapGeometry_Geometry_2X2E,
DeviceTapGeometry_Geometry_2X2M,
DeviceTapGeometry_Geometry_1X8_1Y,
DeviceTapGeometry_Geometry_8X_1Y,
DeviceTapGeometry_Geometry_4X2_1Y,
DeviceTapGeometry_Geometry_2X2E_2YE,
DeviceTapGeometry_Geometry_1X8,
DeviceTapGeometry_Geometry_8X,
DeviceTapGeometry_Geometry_4X2,
DeviceTapGeometry_Geometry_4X2E,
DeviceTapGeometry_Geometry_4X2E_1Y,
DeviceTapGeometry_Geometry_1X10_1Y,
DeviceTapGeometry_Geometry_10X_1Y,
DeviceTapGeometry_Geometry_1X10,
DeviceTapGeometry_Geometry_10X,
NUM_DEVICETAPGEOMETRY }

• enum spinGevPhysicalLinkConfigurationEnums {
    GevPhysicalLinkConfiguration_SingleLink,
    GevPhysicalLinkConfiguration_MultiLink,
    GevPhysicalLinkConfiguration_StaticLAG,
    GevPhysicalLinkConfiguration_DynamicLAG,
    NUM_GEVPHYSICALLINKCONFIGURATION }

• enum spinGevCurrentPhysicalLinkConfigurationEnums {
    GevCurrentPhysicalLinkConfiguration_SingleLink,
    GevCurrentPhysicalLinkConfiguration_MultiLink,
    GevCurrentPhysicalLinkConfiguration_StaticLAG,
    GevCurrentPhysicalLinkConfiguration_DynamicLAG,
    NUM_GEVCURRENTPHYSICALLINKCONFIGURATION }

• enum spinGevIPConfigurationStatusEnums {
    GevIPConfigurationStatus_None,
    GevIPConfigurationStatus_PersistentIP,
    GevIPConfigurationStatus_DHCP,
    GevIPConfigurationStatus_LLA,
    GevIPConfigurationStatus_ForceIP,

```

```

NUM_GEVIPCONFIGURATIONSTATUS }

• enum spinGevGVCPExtendedStatusCodesSelectorEnums {
    GevGVCPExtendedStatusCodesSelector_Version1_1,
    GevGVCPExtendedStatusCodesSelector_Version2_0,
    NUM_GEVGVCPEXTENDEDSTATUSCODESSELECTOR }

• enum spinGevGVSPExtendedIDModeEnums {
    GevGVSPExtendedIDMode_Off,
    GevGVSPExtendedIDMode_On,
    NUM_GEVGVSPEXTENDEDIDMODE }

• enum spinCIConfigurationEnums {
    CIConfiguration_Base,
    CIConfiguration_Medium,
    CIConfiguration_Full,
    CIConfiguration_DualBase,
    CIConfiguration_EightyBit,
    NUM_CLCONFIGURATION }

• enum spinCITimeSlotsCountEnums {
    CITimeSlotsCount_One,
    CITimeSlotsCount_Two,
    CITimeSlotsCount_Three,
    NUM_CLTIMESLOTSCOUNT }

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

• enum spinCxpLinkConfigurationPreferredEnums {
    CxpLinkConfigurationPreferred_CXP1_X1,
    CxpLinkConfigurationPreferred_CXP2_X1,

```

```

CxpLinkConfigurationPreferred_CXP3_X1,
CxpLinkConfigurationPreferred_CXP5_X1,
CxpLinkConfigurationPreferred_CXP6_X1,
CxpLinkConfigurationPreferred_CXP1_X2,
CxpLinkConfigurationPreferred_CXP2_X2,
CxpLinkConfigurationPreferred_CXP3_X2,
CxpLinkConfigurationPreferred_CXP5_X2,
CxpLinkConfigurationPreferred_CXP6_X2,
CxpLinkConfigurationPreferred_CXP1_X3,
CxpLinkConfigurationPreferred_CXP2_X3,
CxpLinkConfigurationPreferred_CXP3_X3,
CxpLinkConfigurationPreferred_CXP5_X3,
CxpLinkConfigurationPreferred_CXP6_X3,
CxpLinkConfigurationPreferred_CXP1_X4,
CxpLinkConfigurationPreferred_CXP2_X4,
CxpLinkConfigurationPreferred_CXP3_X4,
CxpLinkConfigurationPreferred_CXP5_X4,
CxpLinkConfigurationPreferred_CXP6_X4,
CxpLinkConfigurationPreferred_CXP1_X5,
CxpLinkConfigurationPreferred_CXP2_X5,
CxpLinkConfigurationPreferred_CXP3_X5,
CxpLinkConfigurationPreferred_CXP5_X5,
CxpLinkConfigurationPreferred_CXP6_X5,
CxpLinkConfigurationPreferred_CXP1_X6,
CxpLinkConfigurationPreferred_CXP2_X6,
CxpLinkConfigurationPreferred_CXP3_X6,
CxpLinkConfigurationPreferred_CXP5_X6,
CxpLinkConfigurationPreferred_CXP6_X6,
NUM_CXPLINKCONFIGURATIONPREFERRED }

```

- `enum spinCxpLinkConfigurationEnums {`

```

CxpLinkConfiguration_Auto,
CxpLinkConfiguration_CXP1_X1,
CxpLinkConfiguration_CXP2_X1,
CxpLinkConfiguration_CXP3_X1,
CxpLinkConfiguration_CXP5_X1,
CxpLinkConfiguration_CXP6_X1,
CxpLinkConfiguration_CXP1_X2,
CxpLinkConfiguration_CXP2_X2,
CxpLinkConfiguration_CXP3_X2,
CxpLinkConfiguration_CXP5_X2,
CxpLinkConfiguration_CXP6_X2,
CxpLinkConfiguration_CXP1_X3,
CxpLinkConfiguration_CXP2_X3,
CxpLinkConfiguration_CXP3_X3,
CxpLinkConfiguration_CXP5_X3,
CxpLinkConfiguration_CXP6_X3,
CxpLinkConfiguration_CXP1_X4,
CxpLinkConfiguration_CXP2_X4,
CxpLinkConfiguration_CXP3_X4,
CxpLinkConfiguration_CXP5_X4,
CxpLinkConfiguration_CXP6_X4,
CxpLinkConfiguration_CXP1_X5,
CxpLinkConfiguration_CXP2_X5,
CxpLinkConfiguration_CXP3_X5,
CxpLinkConfiguration_CXP5_X5,
CxpLinkConfiguration_CXP6_X5,
CxpLinkConfiguration_CXP1_X6,
CxpLinkConfiguration_CXP2_X6,

```

```

CxpLinkConfiguration_CXP3_X6,
CxpLinkConfiguration_CXP5_X6,
CxpLinkConfiguration_CXP6_X6,
NUM_CXPLINKCONFIGURATION }
• enum spinCxpConnectionTestModeEnums {
  CxpConnectionTestMode_Off,
  CxpConnectionTestMode_Mode1,
  NUM_CXPCONNECTIONTESTMODE }
• enum spinCxpPoCxpStatusEnums {
  CxpPoCxpStatus_Auto,
  CxpPoCxpStatus_Off,
  CxpPoCxpStatus_Tripped,
  NUM_CXPPOCXPSTATUS }

```

### 6.2.1 Detailed Description

### 6.2.2 Enumeration Type Documentation

#### 6.2.2.1 spinAcquisitionModeEnums

```
enum spinAcquisitionModeEnums
```

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

##### Enumerator

AcquisitionMode_Continuous	
AcquisitionMode_SingleFrame	
AcquisitionMode_MultiFrame	
NUM_ACQUISITIONMODE	

#### 6.2.2.2 spinAcquisitionStatusSelectorEnums

```
enum spinAcquisitionStatusSelectorEnums
```

< Selects the internal acquisition signal to read using AcquisitionStatus.

##### Enumerator

AcquisitionStatusSelector_AcquisitionTriggerWait	Device is currently waiting for a trigger for the capture of one or many frames.
AcquisitionStatusSelector_AcquisitionActive	Device is currently doing an acquisition of one or many frames.

## Enumerator

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	

## 6.2.2.3 spinActionUnconditionalModeEnums

enum [spinActionUnconditionalModeEnums](#)

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

## Enumerator

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

## 6.2.2.4 spinAdcBitDepthEnums

enum [spinAdcBitDepthEnums](#)

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

## Enumerator

AdcBitDepth_Bit8	
AdcBitDepth_Bit10	
AdcBitDepth_Bit12	
AdcBitDepth_Bit14	
NUM_ADCBITDEPTH	

## 6.2.2.5 spinAutoAlgorithmSelectorEnums

enum [spinAutoAlgorithmSelectorEnums](#)

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

## Enumerator

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

## 6.2.2.6 spinAutoExposureControlPriorityEnums

```
enum spinAutoExposureControlPriorityEnums
```

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

## Enumerator

AutoExposureControlPriority_Gain	
AutoExposureControlPriority_ExposureTime	
NUM_AUTOEXPOSURECONTROLPRIORITY	

## 6.2.2.7 spinAutoExposureLightingModeEnums

```
enum spinAutoExposureLightingModeEnums
```

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

## Enumerator

AutoExposureLightingMode_AutoDetect	
AutoExposureLightingMode_Backlight	
AutoExposureLightingMode_Frontlight	
AutoExposureLightingMode_Normal	
NUM_AUTOEXPOSURELIGHTINGMODE	

## 6.2.2.8 spinAutoExposureMeteringModeEnums

```
enum spinAutoExposureMeteringModeEnums
```

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

#### Enumerator

AutoExposureMeteringMode_Average	
AutoExposureMeteringMode_Spot	
AutoExposureMeteringMode_Partial	
AutoExposureMeteringMode_CenterWeighted	
AutoExposureMeteringMode_HistogramPeak	
NUM_AUTOEXPOSUREMETERINGMODE	

#### 6.2.2.9 spinAutoExposureTargetGreyValueAutoEnums

enum `spinAutoExposureTargetGreyValueAutoEnums`

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

#### Enumerator

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

#### 6.2.2.10 spinBalanceRatioSelectorEnums

enum `spinBalanceRatioSelectorEnums`

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

#### Enumerator

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



## 6.2.2.11 spinBalanceWhiteAutoEnums

```
enum spinBalanceWhiteAutoEnums
```

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

## Enumerator

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

## 6.2.2.12 spinBalanceWhiteAutoProfileEnums

```
enum spinBalanceWhiteAutoProfileEnums
```

< Selects the profile used by BalanceWhiteAuto.

## Enumerator

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

## 6.2.2.13 spinBinningHorizontalModeEnums

```
enum spinBinningHorizontalModeEnums
```

<

## Enumerator

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

#### 6.2.2.14 spinBinningSelectorEnums

enum `spinBinningSelectorEnums`

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

##### Enumerator

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

#### 6.2.2.15 spinBinningVerticalModeEnums

enum `spinBinningVerticalModeEnums`

<

##### Enumerator

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

#### 6.2.2.16 spinBlackLevelAutoBalanceEnums

enum `spinBlackLevelAutoBalanceEnums`

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

##### Enumerator

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

### 6.2.2.17 spinBlackLevelAutoEnums

enum `spinBlackLevelAutoEnums`

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

#### Enumerator

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

### 6.2.2.18 spinBlackLevelSelectorEnums

enum `spinBlackLevelSelectorEnums`

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

#### Enumerator

BlackLevelSelector_All	
BlackLevelSelector_Analog	
BlackLevelSelector_Digital	
NUM_BLACKLEVELSELECTOR	

### 6.2.2.19 spinChunkBlackLevelSelectorEnums

enum `spinChunkBlackLevelSelectorEnums`

< Selects which black level to retrieve

#### Enumerator

ChunkBlackLevelSelector_All	
NUM_CHUNKBLACKLEVELSELECTOR	

### 6.2.2.20 spinChunkCounterSelectorEnums

enum `spinChunkCounterSelectorEnums`

< Selects which counter to retrieve data from.

#### Enumerator

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

### 6.2.2.21 spinChunkEncoderSelectorEnums

enum `spinChunkEncoderSelectorEnums`

< Selects which Encoder to retrieve data from.

#### Enumerator

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

### 6.2.2.22 spinChunkEncoderStatusEnums

enum `spinChunkEncoderStatusEnums`

< Returns the motion status of the selected encoder.

#### Enumerator

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

### 6.2.2.23 spinChunkExposureTimeSelectorEnums

enum `spinChunkExposureTimeSelectorEnums`

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

#### Enumerator

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

#### 6.2.2.24 spinChunkGainSelectorEnums

enum `spinChunkGainSelectorEnums`

< Selects which gain to retrieve

#### Enumerator

ChunkGainSelector_All	
ChunkGainSelector_Red	
ChunkGainSelector_Green	
ChunkGainSelector_Blue	
NUM_CHUNKGAINSELECTOR	

#### 6.2.2.25 spinChunkImageComponentEnums

enum `spinChunkImageComponentEnums`

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

#### Enumerator

ChunkImageComponent_Intensity	The image data is the intensity component.
ChunkImageComponent_Color	The image data is color component.
ChunkImageComponent_Infrared	The image data is infrared component.
ChunkImageComponent_Ultraviolet	The image data is the ultraviolet component.

**Enumerator**

ChunkImageComponent_Range	The image data is the range (distance) component.
ChunkImageComponent_Disparity	The image data is the disparity component.
ChunkImageComponent_Confidence	The image data is the confidence map component.
ChunkImageComponent_Scatter	The image data is the scatter component.
NUM_CHUNKIMAGECOMPONENT	

**6.2.2.26 spinChunkPixelFormatEnums**

enum [spinChunkPixelFormatEnums](#)

< Format of the pixel provided by the camera

**Enumerator**

ChunkPixelFormat_Mono8	
ChunkPixelFormat_Mono12Packed	
ChunkPixelFormat_Mono16	
ChunkPixelFormat_RGB8Packed	
ChunkPixelFormat_YUV422Packed	
ChunkPixelFormat_BayerGR8	
ChunkPixelFormat_BayerRG8	
ChunkPixelFormat_BayerGB8	
ChunkPixelFormat_BayerBG8	
ChunkPixelFormat_YCbCr601_422_8_CbYCrY	
NUM_CHUNKPIXELFORMAT	

**6.2.2.27 spinChunkRegionIDEnums**

enum [spinChunkRegionIDEnums](#)

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

**Enumerator**

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

**6.2.2.28 spinChunkScan3dCoordinateReferenceSelectorEnums**

enum `spinChunkScan3dCoordinateReferenceSelectorEnums`

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

**Enumerator**

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

**6.2.2.29 spinChunkScan3dCoordinateSelectorEnums**

enum `spinChunkScan3dCoordinateSelectorEnums`

< Selects which Coordinate to retrieve data from.

**Enumerator**

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

**6.2.2.30 spinChunkScan3dCoordinateSystemEnums**

enum `spinChunkScan3dCoordinateSystemEnums`

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

**Enumerator**

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

### 6.2.2.31 spinChunkScan3dCoordinateSystemReferenceEnums

enum `spinChunkScan3dCoordinateSystemReferenceEnums`

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

#### Enumerator

ChunkScan3dCoordinateSystemReference_Anchor	Default value. Original fixed reference. The coordinate system fixed relative the camera reference point marker is used.
ChunkScan3dCoordinateSystemReference_↔ Transformed	Transformed reference system. The transformed coordinate system is used according to the definition in the rotation and translation matrices.
NUM_CHUNKSCAN3DCOORDINATESYSTEMREF↔ ERENCE	

### 6.2.2.32 spinChunkScan3dCoordinateTransformSelectorEnums

enum `spinChunkScan3dCoordinateTransformSelectorEnums`

< Selector for transform values.

#### Enumerator

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

### 6.2.2.33 spinChunkScan3dDistanceUnitEnums

enum `spinChunkScan3dDistanceUnitEnums`

< Returns the Distance Unit of the payload image.

#### Enumerator

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



## 6.2.2.34 spinChunkScan3dOutputModeEnums

enum [spinChunkScan3dOutputModeEnums](#)

< Returns the Calibrated Mode of the payload image.

## Enumerator

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

### 6.2.2.35 spinChunkSelectorEnums

enum `spinChunkSelectorEnums`

< Selects which chunk data to enable or disable.

#### Enumerator

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

### 6.2.2.36 spinChunkSourceIDEnums

enum `spinChunkSourceIDEnums`

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

#### Enumerator

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

### 6.2.2.37 spinChunkTimerSelectorEnums

enum `spinChunkTimerSelectorEnums`

< Selects which Timer to retrieve data from.

## Enumerator

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

## 6.2.2.38 spinChunkTransferStreamIDEnums

```
enum spinChunkTransferStreamIDEnums
```

< Returns identifier of the stream that generated this block.

## Enumerator

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

## 6.2.2.39 spinClConfigurationEnums

```
enum spinClConfigurationEnums
```

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

## Enumerator

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

#### 6.2.2.40 spinCITimeSlotsCountEnums

enum `spinClTimeSlotsCountEnums`

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

##### Enumerator

CITimeSlotsCount_One	One
CITimeSlotsCount_Two	Two
CITimeSlotsCount_Three	Three
NUM_CLTIMESLOTSCOUNT	

#### 6.2.2.41 spinColorTransformationSelectorEnums

enum `spinColorTransformationSelectorEnums`

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

##### Enumerator

ColorTransformationSelector_RGBtoRGB	
ColorTransformationSelector_RGBtoYUV	
NUM_COLORTRANSFORMATIONSELECTOR	

#### 6.2.2.42 spinColorTransformationValueSelectorEnums

enum `spinColorTransformationValueSelectorEnums`

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

##### Enumerator

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

## 6.2.2.43 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	

## 6.2.2.44 spinCounterEventSourceEnums

```
enum spinCounterEventSourceEnums
```

< Selects the event that will increment the counter

## Enumerator

CounterEventSource_Off	Off
CounterEventSource_MHzTick	MHzTick
CounterEventSource_Line0	Line0
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	

### 6.2.2.45 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	

### 6.2.2.46 spinCounterResetSourceEnums

enum `spinCounterResetSourceEnums`

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

#### Enumerator

CounterResetSource_Off	Off
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	

## 6.2.2.47 spinCounterSelectorEnums

```
enum spinCounterSelectorEnums
```

< Selects which counter to configure

## Enumerator

CounterSelector_Counter0	
CounterSelector_Counter1	
NUM_COUNTERSELECTOR	

## 6.2.2.48 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	

## 6.2.2.49 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	

### 6.2.2.50 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	

### 6.2.2.51 spinCxpConnectionTestModeEnums

enum [spinCxpConnectionTestModeEnums](#)

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

#### Enumerator

CxpConnectionTestMode_Off	Off
CxpConnectionTestMode_Mode1	Mode 1
NUM_CXPCONNECTIONTESTMODE	

### 6.2.2.52 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).
<code>CxpLinkConfiguration_CXP1_X5</code>	Force the Link to 5 Connections operating at CXP-1 speed (1.25 Gbps).
<code>CxpLinkConfiguration_CXP2_X5</code>	Force the Link to 5 Connections operating at CXP-2 speed (2.50 Gbps).
<code>CxpLinkConfiguration_CXP3_X5</code>	Force the Link to 5 Connections operating at CXP-3 speed (3.125 Gbps).
<code>CxpLinkConfiguration_CXP5_X5</code>	Force the Link to 5 Connections operating at CXP-5 speed (5.00 Gbps).
<code>CxpLinkConfiguration_CXP6_X5</code>	Force the Link to 5 Connections operating at CXP-6 speed (6.25 Gbps).
<code>CxpLinkConfiguration_CXP1_X6</code>	Force the Link to 6 Connections operating at CXP-1 speed (1.25 Gbps).
<code>CxpLinkConfiguration_CXP2_X6</code>	Force the Link to 6 Connections operating at CXP-2 speed (2.50 Gbps).
<code>CxpLinkConfiguration_CXP3_X6</code>	Force the Link to 6 Connections operating at CXP-3 speed (3.125 Gbps).
<code>CxpLinkConfiguration_CXP5_X6</code>	Force the Link to 6 Connections operating at CXP-5 speed (5.00 Gbps).
<code>CxpLinkConfiguration_CXP6_X6</code>	Force the Link to 6 Connections operating at CXP-6 speed (6.25 Gbps).
<code>NUM_CXPLINKCONFIGURATION</code>	

#### 6.2.2.53 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).
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	

## 6.2.2.54 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.

## Enumerator

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).
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	

## 6.2.2.55 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	

#### 6.2.2.56 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	

#### 6.2.2.57 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	

#### 6.2.2.58 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	

## 6.2.2.59 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	

## 6.2.2.60 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	

## 6.2.2.61 spinDeviceCharacterSetEnums

enum `spinDeviceCharacterSetEnums`

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

## Enumerator

DeviceCharacterSet_UTF8	
DeviceCharacterSet_ASCII	
NUM_DEVICECHARACTERSET	

### 6.2.2.62 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	

### 6.2.2.63 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	

### 6.2.2.64 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	

### 6.2.2.65 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	

**6.2.2.66 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	

**6.2.2.67 spinDevicePowerSupplySelectorEnums**

enum `spinDevicePowerSupplySelectorEnums`

< Selects the power supply source to control or read.

**Enumerator**

DevicePowerSupplySelector_External	
NUM_DEVICEPOWERSUPPLYSELECTOR	

**6.2.2.68 spinDeviceRegistersEndiannessEnums**

enum `spinDeviceRegistersEndiannessEnums`

< Endianness of the registers of the device.

**Enumerator**

DeviceRegistersEndianness_Little	
DeviceRegistersEndianness_Big	
NUM_DEVICEREGISTERSENDIANNES	



## 6.2.2.69 spinDeviceScanTypeEnums

```
enum spinDeviceScanTypeEnums
```

< Scan type of the sensor of the device.

## Enumerator

DeviceScanType_Areascan	
NUM_DEVICESCANTYPE	

## 6.2.2.70 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_DEVICESSERIALPORTBAUDRATE	

## 6.2.2.71 spinDeviceSerialPortSelectorEnums

```
enum spinDeviceSerialPortSelectorEnums
```

< Selects which serial port of the device to control.

## Enumerator

DeviceSerialPortSelector_CameraLink	Serial port associated to the Camera link connection.
NUM_DEVICESSERIALPORTSELECTOR	

### 6.2.2.72 spinDeviceStreamChannelEndiannessEnums

enum `spinDeviceStreamChannelEndiannessEnums`

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

#### Enumerator

<code>DeviceStreamChannelEndianness_Big</code>	Stream channel data is big Endian.
<code>DeviceStreamChannelEndianness_Little</code>	Stream channel data is little Endian.
<code>NUM_DEVICESTREAMCHANNELENDIANNESS</code>	

### 6.2.2.73 spinDeviceStreamChannelTypeEnums

enum `spinDeviceStreamChannelTypeEnums`

< Reports the type of the stream channel.

#### Enumerator

<code>DeviceStreamChannelType_Transmitter</code>	Data stream transmitter channel.
<code>DeviceStreamChannelType_Receiver</code>	Data stream receiver channel.
<code>NUM_DEVICESTREAMCHANNELTYPE</code>	

### 6.2.2.74 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

<code>DeviceTapGeometry_Geometry_1X_1Y</code>	Geometry_1X_1Y
<code>DeviceTapGeometry_Geometry_1X2_1Y</code>	Geometry_1X2_1Y
<code>DeviceTapGeometry_Geometry_1X2_1Y2</code>	Geometry_1X2_1Y2
<code>DeviceTapGeometry_Geometry_2X_1Y</code>	Geometry_2X_1Y
<code>DeviceTapGeometry_Geometry_2X_1Y2Geometry_2XE_1Y</code>	Geometry_2X_1Y2Geometry_2XE_1Y
<code>DeviceTapGeometry_Geometry_2XE_1Y2</code>	Geometry_2XE_1Y2
<code>DeviceTapGeometry_Geometry_2XM_1Y</code>	Geometry_2XM_1Y
<code>DeviceTapGeometry_Geometry_2XM_1Y2</code>	Geometry_2XM_1Y2
<code>DeviceTapGeometry_Geometry_1X_1Y2</code>	Geometry_1X_1Y2
<code>DeviceTapGeometry_Geometry_1X_2YE</code>	Geometry_1X_2YE
<code>DeviceTapGeometry_Geometry_1X3_1Y</code>	Geometry_1X3_1Y

## Enumerator

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
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_DEVICE_TAPGEOMETRY	

## 6.2.2.75 spinDeviceTemperatureSelectorEnums

```
enum spinDeviceTemperatureSelectorEnums
```

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

## Enumerator

DeviceTemperatureSelector_Sensor	
NUM_DEVICETEMPERATURESELECTOR	

## 6.2.2.76 spinDeviceTLTypeEnums

enum [spinDeviceTLTypeEnums](#)

< Transport Layer type of the device.

## Enumerator

DeviceTLType_GigEVision	
DeviceTLType_CameraLink	
DeviceTLType_CameraLinkHS	
DeviceTLType_CoaXPress	
DeviceTLType_USB3Vision	
DeviceTLType_Custom	
NUM_DEVICETLTYPE	

## 6.2.2.77 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_DEVICEYPE	

## 6.2.2.78 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	

## 6.2.2.79 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.
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	

## 6.2.2.80 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	

## 6.2.2.81 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.
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).

## Enumerator

EncoderResetSource_LinkTrigger2	Resets on the reception of the chosen Link Trigger (received from the transport layer).
NUM_ENCODERRESETSOURCE	

## 6.2.2.82 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	

## 6.2.2.83 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	

## 6.2.2.84 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	

**6.2.2.85 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	

**6.2.2.86 spinEventNotificationEnums**

enum `spinEventNotificationEnums`

< Enables/Disables the selected event.

**Enumerator**

EventNotification_On	
EventNotification_Off	
NUM_EVENTNOTIFICATION	

**6.2.2.87 spinEventSelectorEnums**

enum `spinEventSelectorEnums`

< Selects which Event to enable or disable.



## Enumerator

EventSelector_Error	
EventSelector_ExposureEnd	
EventSelector_SerialPortReceive	
NUM_EVENTSELECTOR	

## 6.2.2.88 spinExposureActiveModeEnums

enum `spinExposureActiveModeEnums`

< Control sensor active exposure mode.

## Enumerator

ExposureActiveMode_Line1	
ExposureActiveMode_AnyPixels	
ExposureActiveMode_AllPixels	
NUM_EXPOSUREACTIVEMODE	

## 6.2.2.89 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	

## 6.2.2.90 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	

## 6.2.2.91 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	

## 6.2.2.92 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	

## 6.2.2.93 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	

## 6.2.2.94 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	

## 6.2.2.95 spinFileOperationStatusEnums

```
enum spinFileOperationStatusEnums
```

< Represents the file operation execution status.

## Enumerator

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

### 6.2.2.96 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	

### 6.2.2.97 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	

### 6.2.2.98 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	

## 6.2.2.99 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	

## 6.2.2.100 spinGevCCPEnums

```
enum spinGevCCPEnums
```

< Controls the device access privilege of an application.

## Enumerator

GevCCP_OpenAccess	
GevCCP_ExclusiveAccess	
GevCCP_ControlAccess	
NUM_GEVCCP	

## 6.2.2.101 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	

#### 6.2.2.102 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	

#### 6.2.2.103 spinGevGVSPExtendedIDModeEnums

enum [spinGevGVSPExtendedIDModeEnums](#)

< Enables the extended IDs mode.

##### Enumerator

GevGVSPExtendedIDMode_Off	Off
GevGVSPExtendedIDMode_On	On
NUM_GEVGVSPEXTENDEDIDMODE	

#### 6.2.2.104 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	

#### 6.2.2.105 spinGevIEEE1588ModeEnums

enum [spinGevIEEE1588ModeEnums](#)

< Provides the mode of the IEEE 1588 clock.

## Enumerator

GevIEEE1588Mode_Auto	Automatic
GevIEEE1588Mode_SlaveOnly	Slave Only
NUM_GEVIEEE1588MODE	

## 6.2.2.106 spinGevIEEE1588StatusEnums

```
enum spinGevIEEE1588StatusEnums
```

< Provides the status of the IEEE 1588 clock.

## Enumerator

GevIEEE1588Status_Initializing	Initializing
GevIEEE1588Status_Faulty	Faulty
GevIEEE1588Status_Disabled	Disabled
GevIEEE1588Status_Listening	Listening
GevIEEE1588Status_PreMaster	Pre Master
GevIEEE1588Status_Master	Master
GevIEEE1588Status_Passive	Passive
GevIEEE1588Status_Uncalibrated	Uncalibrated
GevIEEE1588Status_Slave	Slave
NUM_GEVIEEE1588STATUS	

## 6.2.2.107 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	

## 6.2.2.108 spinGevPhysicalLinkConfigurationEnums

enum `spinGevPhysicalLinkConfigurationEnums`

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

## Enumerator

<code>GevPhysicalLinkConfiguration_SingleLink</code>	Single Link
<code>GevPhysicalLinkConfiguration_MultiLink</code>	Multi Link
<code>GevPhysicalLinkConfiguration_StaticLAG</code>	Static LAG
<code>GevPhysicalLinkConfiguration_DynamicLAG</code>	Dynamic LAG
<code>NUM_GEVPHYSICALLINKCONFIGURATION</code>	

## 6.2.2.109 spinGevSupportedOptionSelectorEnums

enum `spinGevSupportedOptionSelectorEnums`

< Selects the GEV option to interrogate for existing support.

## Enumerator

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



## 6.2.2.110 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.
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	

## 6.2.2.111 spinImageCompressionJPEGFormatOptionEnums

```
enum spinImageCompressionJPEGFormatOptionEnums
```

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

## Enumerator

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

**Enumerator**

ImageCompressionJPEGFormatOption_Progressive	Indicates this is a progressive (multi-scan) DCT-based JPEG.
NUM_IMAGECOMPRESSIONJPEGFORMATOPTION	

**6.2.2.112 spinImageCompressionModeEnums**

enum `spinImageCompressionModeEnums`

<

**Enumerator**

ImageCompressionMode_Off	
ImageCompressionMode_Lossless	
NUM_IMAGECOMPRESSIONMODE	

**6.2.2.113 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	

**6.2.2.114 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	

## 6.2.2.115 spinLineInputFilterSelectorEnums

```
enum spinLineInputFilterSelectorEnums
```

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

## Enumerator

LineInputFilterSelector_Deglitch	
LineInputFilterSelector_Debounce	
NUM_LINEINPUTFILTERSELECTOR	

## 6.2.2.116 spinLineModeEnums

```
enum spinLineModeEnums
```

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

## Enumerator

LineMode_Input	
LineMode_Output	
NUM_LINEMODE	

## 6.2.2.117 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	

**6.2.2.118 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	

**6.2.2.119 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	

## 6.2.2.120 spinLogicBlockLUTInputSelectorEnums

```
enum spinLogicBlockLUTInputSelectorEnums
```

< Controls which LogicBlockLUT Input Source & Activation to access.

## Enumerator

LogicBlockLUTInputSelector_Input0	
LogicBlockLUTInputSelector_Input1	
LogicBlockLUTInputSelector_Input2	
LogicBlockLUTInputSelector_Input3	
NUM_LOGICBLOCKLUTINPUTSELECTOR	

## 6.2.2.121 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

**Enumerator**

LogicBlockLUTInputSource_Counter1End	Counter1End
LogicBlockLUTInputSource_LogicBlock0	LogicBlock0
LogicBlockLUTInputSource_LogicBlock1	LogicBlock1
LogicBlockLUTInputSource_ExposureStart	ExposureStart
LogicBlockLUTInputSource_ExposureEnd	ExposureEnd
LogicBlockLUTInputSource_FrameTriggerWait	FrameTriggerWait
LogicBlockLUTInputSource_AcquisitionActive	AcquisitionActive
NUM_LOGICBLOCKLUTINPUTSOURCE	

**6.2.2.122 spinLogicBlockLUTSelectorEnums**

```
enum spinLogicBlockLUTSelectorEnums
```

< Selects which LogicBlock LUT to configure

**Enumerator**

LogicBlockLUTSelector_Value	
LogicBlockLUTSelector_Enable	
NUM_LOGICBLOCKLUTSELECTOR	

**6.2.2.123 spinLogicBlockSelectorEnums**

```
enum spinLogicBlockSelectorEnums
```

< Selects which LogicBlock to configure

**Enumerator**

LogicBlockSelector_LogicBlock0	
LogicBlockSelector_LogicBlock1	
NUM_LOGICBLOCKSELECTOR	

**6.2.2.124 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	

## 6.2.2.125 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.
PixelColorFilter_BayerGB	Bayer Green Blue filter.
PixelColorFilter_BayerGR	Bayer Green Red filter.
PixelColorFilter_BayerBG	Bayer Blue Green filter.
NUM_PIXELCOLORFILTER	

## 6.2.2.126 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	

## Enumerator

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	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



## Enumerator

PixelFormat_RGB8	Red-Green-Blue 8-bit
PixelFormat_RGB8_Planar	Red-Green-Blue 8-bit planar
PixelFormat_RGB10	Red-Green-Blue 10-bit unpacked
PixelFormat_RGB10_Planar	Red-Green-Blue 10-bit unpacked planar
PixelFormat_RGB10p	Red-Green-Blue 10-bit packed
PixelFormat_RGB10p32	Red-Green-Blue 10-bit packed into 32-bit
PixelFormat_RGB12	Red-Green-Blue 12-bit unpacked
PixelFormat_RGB12_Planar	Red-Green-Blue 12-bit unpacked planar
PixelFormat_RGB12p	Red-Green-Blue 12-bit packed
PixelFormat_RGB14	Red-Green-Blue 14-bit unpacked
PixelFormat_RGB16	Red-Green-Blue 16-bit
PixelFormat_RGB16s	Red-Green-Blue 16-bit signed
PixelFormat_RGB32f	Red-Green-Blue 32-bit float
PixelFormat_RGB16_Planar	Red-Green-Blue 16-bit planar
PixelFormat_RGB565p	Red-Green-Blue 5/6/5-bit packed
PixelFormat_BGRa10	Blue-Green-Red-alpha 10-bit unpacked
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

## Enumerator

PixelFormat_Coord3D_ABC16_Planar	3D coordinate A-B-C 16-bit planar
PixelFormat_Coord3D_ABC32f	3D coordinate A-B-C 32-bit floating point
PixelFormat_Coord3D_ABC32f_Planar	3D coordinate A-B-C 32-bit floating point planar
PixelFormat_Coord3D_AC8	3D coordinate A-C 8-bit
PixelFormat_Coord3D_AC8_Planar	3D coordinate A-C 8-bit planar
PixelFormat_Coord3D_AC10p	3D coordinate A-C 10-bit packed
PixelFormat_Coord3D_AC10p_Planar	3D coordinate A-C 10-bit packed planar
PixelFormat_Coord3D_AC12p	3D coordinate A-C 12-bit packed
PixelFormat_Coord3D_AC12p_Planar	3D coordinate A-C 12-bit packed planar
PixelFormat_Coord3D_AC16	3D coordinate A-C 16-bit
PixelFormat_Coord3D_AC16_Planar	3D coordinate A-C 16-bit planar
PixelFormat_Coord3D_AC32f	3D coordinate A-C 32-bit floating point
PixelFormat_Coord3D_AC32f_Planar	3D coordinate A-C 32-bit floating point planar
PixelFormat_Coord3D_A8	3D coordinate A 8-bit
PixelFormat_Coord3D_A10p	3D coordinate A 10-bit packed
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

## Enumerator

PixelFormat_SCF1WBWG12p	Sparse Color Filter #1 White-Blue-White-Green 12-bit packed
PixelFormat_SCF1WBWG14	Sparse Color Filter #1 White-Blue-White-Green 14-bit unpacked
PixelFormat_SCF1WBWG16	Sparse Color Filter #1 White-Blue-White-Green 16-bit unpacked
PixelFormat_SCF1WGWB8	Sparse Color Filter #1 White-Green-White-Blue 8-bit
PixelFormat_SCF1WGWB10	Sparse Color Filter #1 White-Green-White-Blue 10-bit unpacked
PixelFormat_SCF1WGWB10p	Sparse Color Filter #1 White-Green-White-Blue 10-bit packed
PixelFormat_SCF1WGWB12	Sparse Color Filter #1 White-Green-White-Blue 12-bit unpacked
PixelFormat_SCF1WGWB12p	Sparse Color Filter #1 White-Green-White-Blue 12-bit packed
PixelFormat_SCF1WGWB14	Sparse Color Filter #1 White-Green-White-Blue 14-bit unpacked
PixelFormat_SCF1WGWB16	Sparse Color Filter #1 White-Green-White-Blue 16-bit
PixelFormat_SCF1WGWR8	Sparse Color Filter #1 White-Green-White-Red 8-bit
PixelFormat_SCF1WGWR10	Sparse Color Filter #1 White-Green-White-Red 10-bit unpacked
PixelFormat_SCF1WGWR10p	Sparse Color Filter #1 White-Green-White-Red 10-bit packed
PixelFormat_SCF1WGWR12	Sparse Color Filter #1 White-Green-White-Red 12-bit unpacked
PixelFormat_SCF1WGWR12p	Sparse Color Filter #1 White-Green-White-Red 12-bit packed
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

## Enumerator

PixelFormat_YCbCr601_422_10	YCbCr 4:2:2 10-bit unpacked BT.601
PixelFormat_YCbCr601_422_10_CbYCrY	YCbCr 4:2:2 10-bit unpacked BT.601
PixelFormat_YCbCr601_422_10p	YCbCr 4:2:2 10-bit packed BT.601
PixelFormat_YCbCr601_422_10p_CbYCrY	YCbCr 4:2:2 10-bit packed BT.601
PixelFormat_YCbCr601_422_12	YCbCr 4:2:2 12-bit unpacked BT.601
PixelFormat_YCbCr601_422_12_CbYCrY	YCbCr 4:2:2 12-bit unpacked BT.601
PixelFormat_YCbCr601_422_12p	YCbCr 4:2:2 12-bit packed BT.601
PixelFormat_YCbCr601_422_12p_CbYCrY	YCbCr 4:2:2 12-bit packed BT.601
PixelFormat_YCbCr709_8_CbYCr	YCbCr 4:4:4 8-bit BT.709
PixelFormat_YCbCr709_10_CbYCr	YCbCr 4:4:4 10-bit unpacked BT.709
PixelFormat_YCbCr709_10p_CbYCr	YCbCr 4:4:4 10-bit packed BT.709
PixelFormat_YCbCr709_12_CbYCr	YCbCr 4:4:4 12-bit unpacked BT.709
PixelFormat_YCbCr709_12p_CbYCr	YCbCr 4:4:4 12-bit packed BT.709
PixelFormat_YCbCr709_411_8_CbYYCrYY	YCbCr 4:1:1 8-bit BT.709
PixelFormat_YCbCr709_422_8	YCbCr 4:2:2 8-bit BT.709
PixelFormat_YCbCr709_422_8_CbYCrY	YCbCr 4:2:2 8-bit BT.709
PixelFormat_YCbCr709_422_10	YCbCr 4:2:2 10-bit unpacked BT.709
PixelFormat_YCbCr709_422_10_CbYCrY	YCbCr 4:2:2 10-bit unpacked BT.709
PixelFormat_YCbCr709_422_10p	YCbCr 4:2:2 10-bit packed BT.709
PixelFormat_YCbCr709_422_10p_CbYCrY	YCbCr 4:2:2 10-bit packed BT.709
PixelFormat_YCbCr709_422_12	YCbCr 4:2:2 12-bit unpacked BT.709
PixelFormat_YCbCr709_422_12_CbYCrY	YCbCr 4:2:2 12-bit unpacked BT.709
PixelFormat_YCbCr709_422_12p	YCbCr 4:2:2 12-bit packed BT.709
PixelFormat_YCbCr709_422_12p_CbYCrY	YCbCr 4:2:2 12-bit packed BT.709
PixelFormat_YUV8_UYV	YUV 4:4:4 8-bit
PixelFormat_YUV411_8_UYYVYY	YUV 4:1:1 8-bit
PixelFormat_YUV422_8	YUV 4:2:2 8-bit
PixelFormat_YUV422_8_UYVY	YUV 4:2:2 8-bit
PixelFormat_Polarized8	Monochrome Polarized 8-bit
PixelFormat_Polarized10p	Monochrome Polarized 10-bit packed
PixelFormat_Polarized12p	Monochrome Polarized 12-bit packed
PixelFormat_Polarized16	Monochrome Polarized 16-bit
PixelFormat_BayerRGPolarized8	Polarized Bayer Red Green filter 8-bit
PixelFormat_BayerRGPolarized10p	Polarized Bayer Red Green filter 10-bit packed
PixelFormat_BayerRGPolarized12p	Polarized Bayer Red Green filter 12-bit packed
PixelFormat_BayerRGPolarized16	Polarized Bayer Red Green filter 16-bit
PixelFormat_LLCMono8	Lossless Compression Monochrome 8-bit
PixelFormat_LLCBayerRG8	Lossless Compression Bayer Red Green filter 8-bit
PixelFormat_JPEGMono8	JPEG Monochrome 8-bit
PixelFormat_JPEGColor8	JPEG Color 8-bit
PixelFormat_Raw16	Raw 16 bit.
PixelFormat_Raw8	Raw bit.
PixelFormat_R12_Jpeg	Red 12-bit JPEG.
PixelFormat_GR12_Jpeg	Green Red 12-bit JPEG.
PixelFormat_GB12_Jpeg	Green Blue 12-bit JPEG.
PixelFormat_B12_Jpeg	Blue 12-bit packed JPEG.
UNKNOWN_PIXELFORMAT	

## Enumerator

NUM_PIXELFORMAT	
-----------------	--

## 6.2.2.127 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

## Enumerator

PixelFormatInfoSelector_BayerRG12	Bayer Red-Green 12-bit unpacked
PixelFormatInfoSelector_BayerRG12p	Bayer Red-Green 12-bit packed
PixelFormatInfoSelector_BayerRG16	Bayer Red-Green 16-bit
PixelFormatInfoSelector_RGBa8	Red-Green-Blue-alpha 8-bit
PixelFormatInfoSelector_RGBa10	Red-Green-Blue-alpha 10-bit unpacked
PixelFormatInfoSelector_RGBa10p	Red-Green-Blue-alpha 10-bit packed
PixelFormatInfoSelector_RGBa12	Red-Green-Blue-alpha 12-bit unpacked
PixelFormatInfoSelector_RGBa12p	Red-Green-Blue-alpha 12-bit packed
PixelFormatInfoSelector_RGBa14	Red-Green-Blue-alpha 14-bit unpacked
PixelFormatInfoSelector_RGBa16	Red-Green-Blue-alpha 16-bit
PixelFormatInfoSelector_RGB8	Red-Green-Blue 8-bit
PixelFormatInfoSelector_RGB8_Planar	Red-Green-Blue 8-bit planar
PixelFormatInfoSelector_RGB10	Red-Green-Blue 10-bit unpacked
PixelFormatInfoSelector_RGB10_Planar	Red-Green-Blue 10-bit unpacked planar
PixelFormatInfoSelector_RGB10p	Red-Green-Blue 10-bit packed
PixelFormatInfoSelector_RGB10p32	Red-Green-Blue 10-bit packed into 32-bit
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

## Enumerator

PixelFormatInfoSelector_G12	Green 12-bit
PixelFormatInfoSelector_G16	Green 16-bit
PixelFormatInfoSelector_B8	Blue 8-bit
PixelFormatInfoSelector_B10	Blue 10-bit
PixelFormatInfoSelector_B12	Blue 12-bit
PixelFormatInfoSelector_B16	Blue 16-bit
PixelFormatInfoSelector_Coord3D_ABC8	3D coordinate A-B-C 8-bit
PixelFormatInfoSelector_Coord3D_ABC8_Planar	3D coordinate A-B-C 8-bit planar
PixelFormatInfoSelector_Coord3D_ABC10p	3D coordinate A-B-C 10-bit packed
PixelFormatInfoSelector_Coord3D_ABC10p_Planar	3D coordinate A-B-C 10-bit packed planar
PixelFormatInfoSelector_Coord3D_ABC12p	3D coordinate A-B-C 12-bit packed
PixelFormatInfoSelector_Coord3D_ABC12p_Planar	3D coordinate A-B-C 12-bit packed planar
PixelFormatInfoSelector_Coord3D_ABC16	3D coordinate A-B-C 16-bit
PixelFormatInfoSelector_Coord3D_ABC16_Planar	3D coordinate A-B-C 16-bit planar
PixelFormatInfoSelector_Coord3D_ABC32f	3D coordinate A-B-C 32-bit floating point
PixelFormatInfoSelector_Coord3D_ABC32f_Planar	3D coordinate A-B-C 32-bit floating point planar
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

## Enumerator

PixelFormatInfoSelector_BiColorBGRG10	Bi-color Blue/Green - Red/Green 10-bit unpacked
PixelFormatInfoSelector_BiColorBGRG10p	Bi-color Blue/Green - Red/Green 10-bit packed
PixelFormatInfoSelector_BiColorBGRG12	Bi-color Blue/Green - Red/Green 12-bit unpacked
PixelFormatInfoSelector_BiColorBGRG12p	Bi-color Blue/Green - Red/Green 12-bit packed
PixelFormatInfoSelector_BiColorRGBG8	Bi-color Red/Green - Blue/Green 8-bit
PixelFormatInfoSelector_BiColorRGBG10	Bi-color Red/Green - Blue/Green 10-bit unpacked
PixelFormatInfoSelector_BiColorRGBG10p	Bi-color Red/Green - Blue/Green 10-bit packed
PixelFormatInfoSelector_BiColorRGBG12	Bi-color Red/Green - Blue/Green 12-bit unpacked
PixelFormatInfoSelector_BiColorRGBG12p	Bi-color Red/Green - Blue/Green 12-bit packed
PixelFormatInfoSelector_SCF1WBWG8	Sparse Color Filter #1 White-Blue-White-Green 8-bit
PixelFormatInfoSelector_SCF1WBWG10	Sparse Color Filter #1 White-Blue-White-Green 10-bit unpacked
PixelFormatInfoSelector_SCF1WBWG10p	Sparse Color Filter #1 White-Blue-White-Green 10-bit packed
PixelFormatInfoSelector_SCF1WBWG12	Sparse Color Filter #1 White-Blue-White-Green 12-bit unpacked
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



## Enumerator

PixelFormatInfoSelector_SCF1WRWG10p	Sparse Color Filter #1 White-Red-White-Green 10-bit packed
PixelFormatInfoSelector_SCF1WRWG12	Sparse Color Filter #1 White-Red-White-Green 12-bit unpacked
PixelFormatInfoSelector_SCF1WRWG12p	Sparse Color Filter #1 White-Red-White-Green 12-bit packed
PixelFormatInfoSelector_SCF1WRWG14	Sparse Color Filter #1 White-Red-White-Green 14-bit unpacked
PixelFormatInfoSelector_SCF1WRWG16	Sparse Color Filter #1 White-Red-White-Green 16-bit
PixelFormatInfoSelector_YCbCr8	YCbCr 4:4:4 8-bit
PixelFormatInfoSelector_YCbCr8_CbYCr	YCbCr 4:4:4 8-bit
PixelFormatInfoSelector_YCbCr10_CbYCr	YCbCr 4:4:4 10-bit unpacked
PixelFormatInfoSelector_YCbCr10p_CbYCr	YCbCr 4:4:4 10-bit packed
PixelFormatInfoSelector_YCbCr12_CbYCr	YCbCr 4:4:4 12-bit unpacked
PixelFormatInfoSelector_YCbCr12p_CbYCr	YCbCr 4:4:4 12-bit packed
PixelFormatInfoSelector_YCbCr411_8	YCbCr 4:1:1 8-bit
PixelFormatInfoSelector_YCbCr411_8_CbYYCrYY	YCbCr 4:1:1 8-bit
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_CbYYCrYY	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_CbYYCrY	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_CbYYCrY	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_CbYYCrY	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_CbYYCrY	YCbCr 4:2:2 12-bit packed BT.601
PixelFormatInfoSelector_YCbCr709_8_CbYCr	YCbCr 4:4:4 8-bit BT.709

## Enumerator

PixelFormatInfoSelector_YCbCr709_10_CbYCr	YCbCr 4:4:4 10-bit unpacked BT.709
PixelFormatInfoSelector_YCbCr709_10p_CbYCr	YCbCr 4:4:4 10-bit packed BT.709
PixelFormatInfoSelector_YCbCr709_12_CbYCr	YCbCr 4:4:4 12-bit unpacked BT.709
PixelFormatInfoSelector_YCbCr709_12p_CbYCr	YCbCr 4:4:4 12-bit packed BT.709
PixelFormatInfoSelector_YCbCr709_411_8_CbYY↔ CrYY	YCbCr 4:1:1 8-bit BT.709
PixelFormatInfoSelector_YCbCr709_422_8	YCbCr 4:2:2 8-bit BT.709
PixelFormatInfoSelector_YCbCr709_422_8_CbYCrY	YCbCr 4:2:2 8-bit BT.709
PixelFormatInfoSelector_YCbCr709_422_10	YCbCr 4:2:2 10-bit unpacked BT.709
PixelFormatInfoSelector_YCbCr709_422_10_CbY↔ CrY	YCbCr 4:2:2 10-bit unpacked BT.709
PixelFormatInfoSelector_YCbCr709_422_10p	YCbCr 4:2:2 10-bit packed BT.709
PixelFormatInfoSelector_YCbCr709_422_10p_Cb↔ YCrY	YCbCr 4:2:2 10-bit packed BT.709
PixelFormatInfoSelector_YCbCr709_422_12	YCbCr 4:2:2 12-bit unpacked BT.709
PixelFormatInfoSelector_YCbCr709_422_12_CbY↔ CrY	YCbCr 4:2:2 12-bit unpacked BT.709
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	

## 6.2.2.128 spinPixelFormatEnums

```
enum spinPixelFormatEnums
```

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

## Enumerator

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

## 6.2.2.129 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	

## 6.2.2.130 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	

### 6.2.2.131 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	

### 6.2.2.132 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	

## 6.2.2.133 spinScan3dCoordinateReferenceSelectorEnums

enum `spinScan3dCoordinateReferenceSelectorEnums`

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

## Enumerator

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

## 6.2.2.134 spinScan3dCoordinateSelectorEnums

enum `spinScan3dCoordinateSelectorEnums`

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

## Enumerator

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

## 6.2.2.135 spinScan3dCoordinateSystemEnums

enum `spinScan3dCoordinateSystemEnums`

< Specifies the Coordinate system to use for the device.

## Enumerator

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

**6.2.2.136 spinScan3dCoordinateSystemReferenceEnums**

enum `spinScan3dCoordinateSystemReferenceEnums`

< Defines coordinate system reference location.

**Enumerator**

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

**6.2.2.137 spinScan3dCoordinateTransformSelectorEnums**

enum `spinScan3dCoordinateTransformSelectorEnums`

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

**Enumerator**

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

**6.2.2.138 spinScan3dDistanceUnitEnums**

enum `spinScan3dDistanceUnitEnums`

< Specifies the unit used when delivering calibrated distance data.

**Enumerator**

<code>Scan3dDistanceUnit_Millimeter</code>	Distance values are in millimeter units (default).
<code>Scan3dDistanceUnit_Inch</code>	Distance values are in inch units.
<code>NUM_SCAN3DDISTANCEUNIT</code>	

## 6.2.2.139 spinScan3dOutputModeEnums

enum `spinScan3dOutputModeEnums`

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

## Enumerator

<code>Scan3dOutputMode_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>Scan3dOutputMode_CalibratedABC_Grid</code>	3 Coordinates in grid organization. The full 3 coordinate data with the grid array organization from the sensor kept.
<code>Scan3dOutputMode_CalibratedABC_PointCloud</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>Scan3dOutputMode_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>Scan3dOutputMode_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>Scan3dOutputMode_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>Scan3dOutputMode_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>Scan3dOutputMode_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>Scan3dOutputMode_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>Scan3dOutputMode_DisparityC</code>	Disparity 2.5D Depth map. The distance is inversely proportional to the pixel (disparity) value.
<code>Scan3dOutputMode_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_SCAN3DOUTPUTMODE</code>	

## 6.2.2.140 spinSensorDigitizationTapsEnums

enum `spinSensorDigitizationTapsEnums`

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

#### Enumerator

SensorDigitizationTaps_One	1 tap.
SensorDigitizationTaps_Two	2 taps.
SensorDigitizationTaps_Three	3 taps.
SensorDigitizationTaps_Four	4 taps.
SensorDigitizationTaps_Eight	8 taps.
SensorDigitizationTaps_Ten	10 taps.
NUM_SENSORDIGITIZATIONTAPS	

#### 6.2.2.141 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	

#### 6.2.2.142 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	



**6.2.2.143 spinSequencerConfigurationModeEnums**

enum `spinSequencerConfigurationModeEnums`

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

**Enumerator**

SequencerConfigurationMode_Off	
SequencerConfigurationMode_On	
NUM_SEQUENCERCONFIGURATIONMODE	

**6.2.2.144 spinSequencerConfigurationValidEnums**

enum `spinSequencerConfigurationValidEnums`

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

**Enumerator**

SequencerConfigurationValid_No	
SequencerConfigurationValid_Yes	
NUM_SEQUENCERCONFIGURATIONVALID	

**6.2.2.145 spinSequencerModeEnums**

enum `spinSequencerModeEnums`

< Controls whether or not a sequencer is active.

**Enumerator**

SequencerMode_Off	
SequencerMode_On	
NUM_SEQUENCERMODE	

**6.2.2.146 spinSequencerSetValidEnums**

enum `spinSequencerSetValidEnums`

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

#### Enumerator

SequencerSetValid_No	
SequencerSetValid_Yes	
NUM_SEQUENCERSETVALID	

#### 6.2.2.147 spinSequencerTriggerActivationEnums

enum `spinSequencerTriggerActivationEnums`

< Specifies the activation mode of the sequencer trigger.

#### Enumerator

SequencerTriggerActivation_RisingEdge	
SequencerTriggerActivation_FallingEdge	
SequencerTriggerActivation_AnyEdge	
SequencerTriggerActivation_LevelHigh	
SequencerTriggerActivation_LevelLow	
NUM_SEQUENCERTRIGGERACTIVATION	

#### 6.2.2.148 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	

#### 6.2.2.149 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	

## 6.2.2.150 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	

## 6.2.2.151 spinSerialPortSelectorEnums

enum `spinSerialPortSelectorEnums`

< Selects which serial port of the device to control.

## Enumerator

SerialPortSelector_SerialPort0	
NUM_SERIALPORTSELECTOR	

### 6.2.2.152 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	

### 6.2.2.153 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	

### 6.2.2.154 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	

## 6.2.2.155 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	

## 6.2.2.156 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	

## 6.2.2.157 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	

## 6.2.2.158 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	

## 6.2.2.159 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	

## 6.2.2.160 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	

## 6.2.2.161 spinTimerTriggerSourceEnums

```
enum spinTimerTriggerSourceEnums
```

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

## Enumerator

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

**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	

**6.2.2.162 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	

**6.2.2.163 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	

## 6.2.2.164 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	

## 6.2.2.165 spinTransferQueueModeEnums

```
enum spinTransferQueueModeEnums
```

< Specifies the operation mode of the transfer queue.

## Enumerator

TransferQueueMode_FirstInFirstOut	Blocks first In are transferred Out first.
NUM_TRANSFERQUEUEMODE	

## 6.2.2.166 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	

### 6.2.2.167 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>	

### 6.2.2.168 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>	

### 6.2.2.169 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	

## 6.2.2.170 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	

## 6.2.2.171 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	

## 6.2.2.172 spinTriggerActivationEnums

```
enum spinTriggerActivationEnums
```

< Specifies the activation mode of the trigger.

## Enumerator

TriggerActivation_LevelLow	
TriggerActivation_LevelHigh	
TriggerActivation_FallingEdge	
TriggerActivation_RisingEdge	
TriggerActivation_AnyEdge	
NUM_TRIGGERACTIVATION	

## 6.2.2.173 spinTriggerModeEnums

```
enum spinTriggerModeEnums
```

< Controls whether or not trigger is active.

## Enumerator

TriggerMode_Off	
TriggerMode_On	
NUM_TRIGGERMODE	

## 6.2.2.174 spinTriggerOverlapEnums

```
enum spinTriggerOverlapEnums
```

< Specifies the overlap mode of the trigger.

## Enumerator

TriggerOverlap_Off	
TriggerOverlap_ReadOut	
TriggerOverlap_PreviousFrame	
NUM_TRIGGEROVERLAP	

## 6.2.2.175 spinTriggerSelectorEnums

```
enum spinTriggerSelectorEnums
```

< Selects the type of trigger to configure.

## Enumerator

TriggerSelector_AcquisitionStart	
TriggerSelector_FrameStart	
TriggerSelector_FrameBurstStart	
NUM_TRIGGERSELECTOR	

**6.2.2.176 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	

**6.2.2.177 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	

**6.2.2.178 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	

## 6.2.2.179 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	

## 6.2.2.180 spinWhiteClipSelectorEnums

```
enum spinWhiteClipSelectorEnums
```

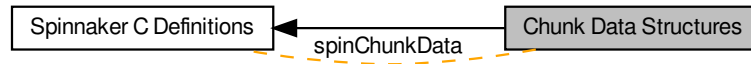
< Selects which White Clip to control.

## 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	

## 6.3 Chunk Data Structures

Collaboration diagram for Chunk Data Structures:



### Data Structures

- struct [spinChunkData](#)

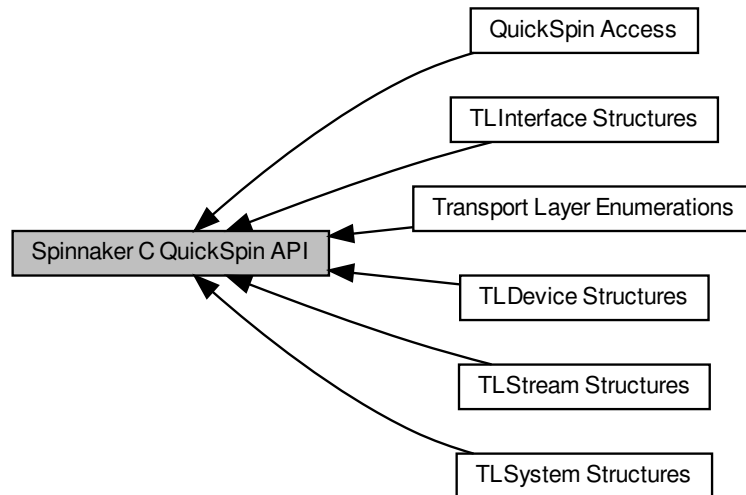
*The type of information that can be obtained from image chunk data.*

#### 6.3.1 Detailed Description



## 6.4 Spinnaker C QuickSpin API

Collaboration diagram for Spinnaker C QuickSpin API:



### Modules

- [QuickSpin Access](#)

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

- [Transport Layer Enumerations](#)
- [TLDevice Structures](#)
- [TLInterface Structures](#)
- [TLStream Structures](#)
- [TLSystem Structures](#)

### 6.4.1 Detailed Description

## 6.5 QuickSpin Access

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

Collaboration diagram for QuickSpin Access:



### Functions

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

### 6.5.1 Detailed Description

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

### 6.5.2 Function Documentation

#### 6.5.2.1 quickSpinInit()

```

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

### 6.5.2.2 quickSpinInitEx()

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

### 6.5.2.3 quickSpinTLDeviceInit()

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

### 6.5.2.4 quickSpinTLInterfaceInit()

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

### 6.5.2.5 quickSpinTLStreamInit()

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

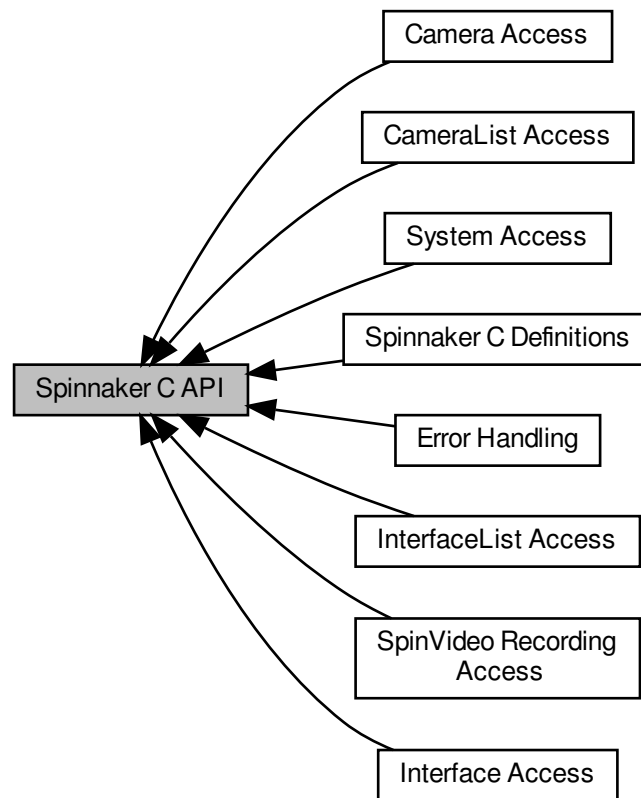
### 6.5.2.6 quickSpinTLSystemInit()

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

## 6.6 Spinnaker C API

SpinnakerPlatform C Include.

Collaboration diagram for Spinnaker C API:



### Modules

- [Spinnaker C Definitions](#)

*Definitions for Spinnaker C.*

- [Error Handling](#)

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

- [System Access](#)

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

- [InterfaceList Access](#)

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

- [CameraList Access](#)

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

- [Interface Access](#)

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

- [Camera Access](#)

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

- [SpinVideo Recording Access](#)

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

## Functions

- [SPINNAKERC\\_API spinCameraDiscoverMaxPacketSize](#) ([spinCamera](#) hCamera, unsigned int \*pMaxPacketSize)

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

### 6.6.1 Detailed Description

SpinnakerPlatform C Include.

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

Spinnaker C Definition Includes

### 6.6.2 Function Documentation

#### 6.6.2.1 spinCameraDiscoverMaxPacketSize()

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

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

See also

[spinError](#)

#### Parameters

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

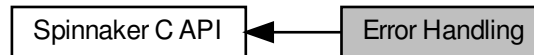
#### Returns

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

## 6.7 Error Handling

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

Collaboration diagram for Error Handling:



### Functions

- [SPINNAKERC\\_API spinErrorGetLast](#) ([spinError](#) \*pError)  
*Retrieves the error code of the last error.*
- [SPINNAKERC\\_API spinErrorGetLastMessage](#) (char \*pBuf, size\_t \*pBufLen)  
*Retrieves the error message of the last error.*
- [SPINNAKERC\\_API spinErrorGetLastBuildDate](#) (char \*pBuf, size\_t \*pBufLen)  
*Retrieves the build date of the last error.*
- [SPINNAKERC\\_API spinErrorGetLastBuildTime](#) (char \*pBuf, size\_t \*pBufLen)  
*Retrieves the build time of the last error.*
- [SPINNAKERC\\_API spinErrorGetLastFileName](#) (char \*pBuf, size\_t \*pBufLen)  
*Retrieves the filename of the last error.*
- [SPINNAKERC\\_API spinErrorGetLastFullMessage](#) (char \*pBuf, size\_t \*pBufLen)  
*Retrieves the full error message of the last error.*
- [SPINNAKERC\\_API spinErrorGetLastFunctionName](#) (char \*pBuf, size\_t \*pBufLen)  
*Retrieves the function name of the last error.*
- [SPINNAKERC\\_API spinErrorGetLastLineNumber](#) (int64\_t \*pLineNum)  
*Retrieves the line number of the last error.*

### 6.7.1 Detailed Description

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

### 6.7.2 Function Documentation

#### 6.7.2.1 spinErrorGetLast()

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

Retrieves the error code of the last error.

See also

[spinError](#)

## Parameters

<i>pError</i>	The error enum pointer in which the error message is returned
---------------	---

## Returns

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

## 6.7.2.2 spinErrorGetLastBuildDate()

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

Retrieves the build date of the last error.

## See also

[spinError](#)

## Parameters

<i>pBuf</i>	The c-string character buffer in which the build date is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

## Returns

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

## 6.7.2.3 spinErrorGetLastBuildTime()

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

Retrieves the build time of the last error.

## See also

[spinError](#)

## Parameters

<i>pBuf</i>	The c-string character buffer in which the build time is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

## Returns

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

## 6.7.2.4 spinErrorGetLastFileName()

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

Retrieves the filename of the last error.

## See also

[spinError](#)

## Parameters

<i>pBuf</i>	The c-string character buffer in which the file name is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

## Returns

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

## 6.7.2.5 spinErrorGetLastFullMessage()

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

Retrieves the full error message of the last error.

## See also

[spinError](#)



## Parameters

<i>pBuf</i>	The c-string character buffer in which the full error message is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

## Returns

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

## 6.7.2.6 spinErrorGetLastFunctionName()

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

Retrieves the function name of the last error.

## See also

[spinError](#)

## Parameters

<i>pBuf</i>	The c-string character buffer in which the function name is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

## Returns

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

## 6.7.2.7 spinErrorGetLastLineNumber()

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

Retrieves the line number of the last error.

## See also

[spinError](#)

**Parameters**

<i>pBuf</i>	The c-string character buffer in which the line number is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

**Returns**

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

**6.7.2.8 spinErrorGetLastMessage()**

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

Retrieves the error message of the last error.

**See also**

[spinError](#)

**Parameters**

<i>pBuf</i>	The c-string character buffer in which the error message is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

**Returns**

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

## 6.8 System Access

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

Collaboration diagram for System Access:



### Functions

- [SPINNAKERC\\_API spinSystemGetInstance \(spinSystem \\*phSystem\)](#)  
Retrieves an instance of the system object; the system is a singleton, so there will only ever be one instance; system instance must be destroyed by calling [spinSystemReleaseInstance](#).
- [SPINNAKERC\\_API spinSystemReleaseInstance \(spinSystem hSystem\)](#)  
Releases the system; make sure handle is cleaned up properly by setting it to NULL after system is released; the handle can only be used again after calling [spinSystemGetInstance](#).
- [SPINNAKERC\\_API spinSystemGetInterfaces \(spinSystem hSystem, spinInterfaceList hInterfaceList\)](#)  
Retrieves a list of detected (and enumerable) interfaces on the system; interface lists must be created and destroyed.
- [SPINNAKERC\\_API spinSystemGetCameras \(spinSystem hSystem, spinCameraList hCameraList\)](#)  
Retrieves a list of detected (and enumerable) cameras on the system; camera lists must be created and destroyed.
- [SPINNAKERC\\_API spinSystemGetCamerasEx \(spinSystem hSystem, bool8\\_t bUpdateInterfaces, bool8\\_t bUpdateCameras, spinCameraList hCameraList\)](#)  
Retrieves a list of detected (and enumerable) cameras on the system; manually set whether to update the current interface and camera lists; camera lists must be created and destroyed.
- [SPINNAKERC\\_API spinSystemSetLoggingLevel \(spinSystem hSystem, spinnakerLogLevel logLevel\)](#)  
Sets the logging level for all logging events on the system.
- [SPINNAKERC\\_API spinSystemGetLoggingLevel \(spinSystem hSystem, spinnakerLogLevel \\*pLogLevel\)](#)  
Retrieves the logging level for all logging events on the system.
- [SPINNAKERC\\_API spinSystemRegisterLogEvent \(spinSystem hSystem, spinLogEvent hLogEvent\)](#)  
Registers a logging event to the system (events registered in this way must be unregistered)
- [SPINNAKERC\\_API spinSystemUnregisterLogEvent \(spinSystem hSystem, spinLogEvent hLogEvent\)](#)  
Unregisters a selected logging event from the system.
- [SPINNAKERC\\_API spinSystemUnregisterAllLogEvents \(spinSystem hSystem\)](#)  
Unregisters all logging events from the system.
- [SPINNAKERC\\_API spinSystemIsInUse \(spinSystem hSystem, bool8\\_t \\*pbIsInUse\)](#)  
Checks whether a system is currently in use.
- [SPINNAKERC\\_API spinSystemRegisterArrivalEvent \(spinSystem hSystem, spinArrivalEvent hArrivalEvent\)](#)  
Registers an arrival event to every interface on the system (events registered this way must be unregistered)
- [SPINNAKERC\\_API spinSystemRegisterRemovalEvent \(spinSystem hSystem, spinRemovalEvent hRemovalEvent\)](#)  
Registers a removal event to the system to every interface on the system (events registered this way must be unregistered)
- [SPINNAKERC\\_API spinSystemUnregisterArrivalEvent \(spinSystem hSystem, spinArrivalEvent hArrivalEvent\)](#)

*Unregisters an arrival event from the system.*

- [SPINNAKERC\\_API spinSystemUnregisterRemovalEvent](#) ([spinSystem](#) hSystem, [spinRemovalEvent](#) hRemovalEvent)

*Unregisters a removal event from the system.*

- [SPINNAKERC\\_API spinSystemRegisterInterfaceEvent](#) ([spinSystem](#) hSystem, [spinInterfaceEvent](#) hInterfaceEvent)

*Registers an interface event (arrival and removal) to every interface on the system (interface events registered this way must be unregistered)*

- [SPINNAKERC\\_API spinSystemUnregisterInterfaceEvent](#) ([spinSystem](#) hSystem, [spinInterfaceEvent](#) hInterfaceEvent)

*Unregisters an interface event 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.*

## 6.8.1 Detailed Description

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

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

## 6.8.2 Function Documentation

### 6.8.2.1 spinSystemGetCameras()

```
SPINNAKERC_API spinSystemGetCameras (
    spinSystem hSystem,
    spinCameraList hCameraList )
```

Retrieves a list of detected (and enumerable) cameras on the system; camera lists must be created and destroyed.

See also

[spinCameraListCreateEmpty\(\)](#)  
[spinCameraListDestroy\(\)](#)  
[spinError](#)

## Parameters

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

## Returns

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

6.8.2.2 `spinSystemGetCamerasEx()`

```
SPINNAKERC_API spinSystemGetCamerasEx (
    spinSystem hSystem,
    bool8_t bUpdateInterfaces,
    bool8_t bUpdateCameras,
    spinCameraList hCameraList )
```

Retrieves a list of detected (and enumerable) cameras on the system; manually set whether to update the current interface and camera lists; camera lists must be created and destroyed.

## See also

[spinCameraListCreateEmpty\(\)](#)  
[spinCameraListDestroy\(\)](#)  
[spinError](#)

## Parameters

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

## Returns

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

6.8.2.3 `spinSystemGetInstance()`

```
SPINNAKERC_API spinSystemGetInstance (
    spinSystem * phSystem )
```

Retrieves an instance of the system object; the system is a singleton, so there will only ever be one instance; system instance must be destroyed by calling `spinSystemReleaseInstance`.

**See also**

[spinSystemReleaseInstance](#)  
[spinError](#)

## Parameters

<i>phSystem</i>	The system handle pointer in which the system instance is returned
-----------------	--

## Returns

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

## 6.8.2.4 spinSystemGetInterfaces()

```
SPINNAKERC_API spinSystemGetInterfaces (
    spinSystem hSystem,
    spinInterfaceList hInterfaceList )
```

Retrieves a list of detected (and enumerable) interfaces on the system; interface lists must be created and destroyed.

## See also

[spinInterfaceListCreateEmpty\(\)](#)  
[spinInterfaceListDestroy\(\)](#)  
[spinError](#)

## Parameters

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

## Returns

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

## 6.8.2.5 spinSystemGetLibraryVersion()

```
SPINNAKERC_API spinSystemGetLibraryVersion (
    spinSystem hSystem,
    spinLibraryVersion * hLibraryVersion )
```

Get current library version of Spinnaker.

## Returns

A struct containing the current version of Spinnaker(major, minor, type, build).

### 6.8.2.6 spinSystemGetLoggingLevel()

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

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

See also

[spinError](#)

#### Parameters

<i>hSystem</i>	The system, from which the logging level is retrieved
<i>logLevel</i>	The logging level enum pointer in which the current logging level is returned

#### Returns

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

### 6.8.2.7 spinSystemGetTLNodeMap()

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

Retrieves the transport layer nodemap from the system.

See also

[spinError](#)

#### Parameters

<i>hSystem</i>	The system handle.
<i>phNodeMap</i>	The nodemap handle pointer in which the transport layer system nodemap is returned.

#### Returns

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



### 6.8.2.8 spinSystemIsInUse()

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

Checks whether a system is currently in use.

See also

[spinError](#)

#### Parameters

<i>hSystem</i>	The system to check
<i>pbIsInUse</i>	The boolean pointer to return whether the system is currently in use

#### Returns

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

### 6.8.2.9 spinSystemRegisterArrivalEvent()

```
SPINNAKERC_API spinSystemRegisterArrivalEvent (
    spinSystem hSystem,
    spinArrivalEvent hArrivalEvent )
```

Registers an arrival event to every interface on the system (events registered this way must be unregistered)

See also

[spinError](#)

#### Parameters

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

#### Returns

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

#### 6.8.2.10 spinSystemRegisterInterfaceEvent()

```
SPINNAKERC_API spinSystemRegisterInterfaceEvent (
    spinSystem hSystem,
    spinInterfaceEvent hInterfaceEvent )
```

Registers an interface event (arrival and removal) to every interface on the system (interface events registered this way must be unregistered)

See also

[spinError](#)

##### Parameters

<i>hSystem</i>	The system, on which the interface event is registered
<i>hInterfaceEvent</i>	The interface event (arrival and removal) to register on the system

##### Returns

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

#### 6.8.2.11 spinSystemRegisterLogEvent()

```
SPINNAKERC_API spinSystemRegisterLogEvent (
    spinSystem hSystem,
    spinLogEvent hLogEvent )
```

Registers a logging event to the system (events registered in this way must be unregistered)

See also

[spinError](#)

##### Parameters

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

##### Returns

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

### 6.8.2.12 spinSystemRegisterRemovalEvent()

```
SPINNAKERC_API spinSystemRegisterRemovalEvent (
    spinSystem hSystem,
    spinRemovalEvent hRemovalEvent )
```

Registers a removal event to the system to every interface on the system (events registered this way must be unregistered)

See also

[spinError](#)

#### Parameters

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

#### Returns

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

### 6.8.2.13 spinSystemReleaseInstance()

```
SPINNAKERC_API spinSystemReleaseInstance (
    spinSystem hSystem )
```

Releases the system; make sure handle is cleaned up properly by setting it to NULL after system is released; the handle can only be used again after calling spinSystemGetInstance.

See also

[spinSystemGetInstance](#)

[spinError](#)

#### Parameters

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

#### Returns

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

#### 6.8.2.14 spinSystemSendActionCommand()

```
SPINNAKERC_API spinSystemSendActionCommand (
    spinSystem hSystem,
    size_t iDeviceKey,
    size_t iGroupKey,
    size_t iGroupMask,
    size_t iActionTime,
    size_t * piResultSize,
    actionCommandResult results[] )
```

Broadcast an Action Command to all devices on system.

See also

[spinError](#)

##### Parameters

<i>hSystem</i>	The system on which to send the action command to all devices.
<i>iDeviceKey</i>	The Action Command's device key
<i>iGroupKey</i>	The Action Command's group key
<i>iGroupMask</i>	The Action Command's group mask
<i>iActionTime</i>	(Optional) Time when to assert a future action. Zero means immediate action.
<i>piResultSize</i>	(Optional) The number of results in the results array. The value passed should be equal to the expected number of devices that acknowledge the command. Returns the number of received results.
<i>results</i>	(Optional) An Array with *piResultSize elements to hold the action command result status. The buffer is filled starting from index 0. If received results are less than expected number of devices that acknowledge the command, remaining results are not changed. If received results are more than expected number of devices that acknowledge the command, extra results are ignored and not appended to array. This parameter is ignored if piResultSize is 0. Thus this parameter can be NULL if pResultSize is 0 or NULL.

##### Returns

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

#### 6.8.2.15 spinSystemSetLoggingLevel()

```
SPINNAKERC_API spinSystemSetLoggingLevel (
    spinSystem hSystem,
    spinnakerLogLevel logLevel )
```

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

See also

[spinError](#)

## Parameters

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

## Returns

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

6.8.2.16 `spinSystemUnregisterAllLogEvents()`

```
SPINNAKERC_API spinSystemUnregisterAllLogEvents (  
    spinSystem hSystem )
```

Unregisters all logging events from the system.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.8.2.17 `spinSystemUnregisterArrivalEvent()`

```
SPINNAKERC_API spinSystemUnregisterArrivalEvent (  
    spinSystem hSystem,  
    spinArrivalEvent hArrivalEvent )
```

Unregisters an arrival event from the system.

## See also

[spinError](#)

## Parameters

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

**Returns**

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

**6.8.2.18 spinSystemUnregisterInterfaceEvent()**

```
SPINNAKERC_API spinSystemUnregisterInterfaceEvent (
    spinSystem hSystem,
    spinInterfaceEvent hInterfaceEvent )
```

Unregisters an interface event from the system.

**See also**

[spinError](#)

**Parameters**

<i>hSystem</i>	The system, from which the interface event is unregistered
<i>hInterfaceEvent</i>	The interface event (arrival and removal) to unregister from the system

**Returns**

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

**6.8.2.19 spinSystemUnregisterLogEvent()**

```
SPINNAKERC_API spinSystemUnregisterLogEvent (
    spinSystem hSystem,
    spinLogEvent hLogEvent )
```

Unregisters a selected logging event from the system.

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

**6.8.2.20 `spinSystemUnregisterRemovalEvent()`**

```
SPINNAKERC_API spinSystemUnregisterRemovalEvent (
    spinSystem hSystem,
    spinRemovalEvent hRemovalEvent )
```

Unregisters a removal event from the system.

**See also**

[`spinError`](#)

**Parameters**

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

**Returns**

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

**6.8.2.21 `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

**6.8.2.22 spinSystemUpdateCamerasEx()**

```
SPINNAKERC_API spinSystemUpdateCamerasEx (
    spinSystem hSystem,
    bool8_t bUpdateInterfaces,
    bool8_t * pbChanged )
```

Updates the list of cameras on the system, informing whether there has been any changes; manually set whether to update the current interface lists.

**See also**

[spinError](#)

**Parameters**

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

**Returns**

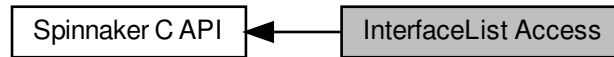
`spinError` The error code; returns `SPINNAKER_ERR_SUCCESS` (or 0) for no error



## 6.9 InterfaceList Access

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

Collaboration diagram for InterfaceList Access:



### Functions

- [SPINNAKERC\\_API spinInterfaceListCreateEmpty](#) ([spinInterfaceList](#) \*phInterfaceList)  
*Creates an empty interface list (interface lists created this way must be destroyed)*
- [SPINNAKERC\\_API spinInterfaceListDestroy](#) ([spinInterfaceList](#) hInterfaceList)  
*Destroys an interface list.*
- [SPINNAKERC\\_API spinInterfaceListGetSize](#) ([spinInterfaceList](#) hInterfaceList, [size\\_t](#) \*pSize)  
*Retrieves the number of interfaces in an interface list.*
- [SPINNAKERC\\_API spinInterfaceListGet](#) ([spinInterfaceList](#) hInterfaceList, [size\\_t](#) index, [spinInterface](#) \*ph↔  
Interface)  
*Retrieves an interface from an interface list using an index (interfaces retrieved this way must be released)*
- [SPINNAKERC\\_API spinInterfaceListClear](#) ([spinInterfaceList](#) hInterfaceList)  
*Clears an interface list.*

### 6.9.1 Detailed Description

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

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

### 6.9.2 Function Documentation

#### 6.9.2.1 spinInterfaceListClear()

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

Clears an interface list.

See also

[spinError](#)

## Parameters

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

## Returns

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

6.9.2.2 `spinInterfaceListCreateEmpty()`

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

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

## See also

[spinError](#)

## Parameters

<i>phInterfaceList</i>	The interface list handle pointer in which the empty interface list is returned
------------------------	---

## Returns

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

6.9.2.3 `spinInterfaceListDestroy()`

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

Destroys an interface list.

## See also

[spinError](#)

## Parameters

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

**Returns**

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

**6.9.2.4 spinInterfaceListGet()**

```
SPINNAKERC_API spinInterfaceListGet (
    spinInterfaceList hInterfaceList,
    size_t index,
    spinInterface * phInterface )
```

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

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

**6.9.2.5 spinInterfaceListGetSize()**

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

Retrieves the number of interfaces in an interface list.

**See also**

[spinError](#)

**Parameters**

<i>hInterfaceList</i>	The interface list where the interfaces to be counted are
<i>pSize</i>	The unsigned integer pointer in which the number of interfaces is returned

**Returns**

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

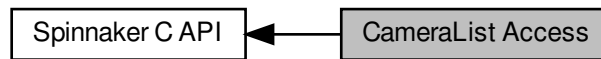
**See also**

[spinError](#)

## 6.10 CameraList Access

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

Collaboration diagram for CameraList Access:



### Functions

- [SPINNAKERC\\_API spinCameraListCreateEmpty](#) ([spinCameraList](#) \*phCameraList)  
*Creates an empty camera list (camera lists created this way must be destroyed)*
- [SPINNAKERC\\_API spinCameraListDestroy](#) ([spinCameraList](#) hCameraList)  
*Destroys a camera list.*
- [SPINNAKERC\\_API spinCameraListGetSize](#) ([spinCameraList](#) hCameraList, [size\\_t](#) \*pSize)  
*Retrieves the number of cameras on a camera list.*
- [SPINNAKERC\\_API spinCameraListGet](#) ([spinCameraList](#) hCameraList, [size\\_t](#) index, [spinCamera](#) \*phCamera)  
*Retrieves a camera from a camera list using an index.*
- [SPINNAKERC\\_API spinCameraListClear](#) ([spinCameraList](#) hCameraList)  
*Clears a camera list.*
- [SPINNAKERC\\_API spinCameraListRemove](#) ([spinCameraList](#) hCameraList, [size\\_t](#) index)  
*Removes a camera from a camera list using its index.*
- [SPINNAKERC\\_API spinCameraListAppend](#) ([spinCameraList](#) hCameraListBase, [spinCameraList](#) hCameraListToAppend)  
*Appends all the cameras from one camera list to another.*
- [SPINNAKERC\\_API spinCameraListGetBySerial](#) ([spinCameraList](#) hCameraList, [const char](#) \*pSerial, [spinCamera](#) \*phCamera)  
*Retrieves a camera from a camera list using its serial number.*
- [SPINNAKERC\\_API spinCameraListRemoveBySerial](#) ([spinCameraList](#) hCameraList, [const char](#) \*pSerial)  
*Removes a camera from a camera list using its serial number.*

### 6.10.1 Detailed Description

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

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

### 6.10.2 Function Documentation

#### 6.10.2.1 spinCameraListAppend()

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

Appends all the cameras from one camera list to another.

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.10.2.2 spinCameraListClear()

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

Clears a camera list.

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.10.2.3 spinCameraListCreateEmpty()

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

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

See also

[spinError](#)

Parameters

<i>phCameraList</i>	The camera list handle pointer in which the empty camera list is returned
---------------------	---

Returns

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

#### 6.10.2.4 spinCameraListDestroy()

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

Destroys a camera list.

See also

[spinError](#)

Parameters

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

Returns

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

#### 6.10.2.5 spinCameraListGet()

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

Retrieves a camera from a camera list using an index.

This function will return a SPINNAKER\_ERR\_INVALID\_PARAMETER error if the input index is out of range.

See also

[spinError](#)

**Parameters**

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

**Returns**

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

**6.10.2.6 spinCameraListGetBySerial()**

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

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

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

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

**6.10.2.7 spinCameraListGetSize()**

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

Retrieves the number of cameras on a camera list.

**See also**

[spinError](#)



## Parameters

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

## Returns

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

6.10.2.8 `spinCameraListRemove()`

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

Removes a camera from a camera list using its index.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.10.2.9 `spinCameraListRemoveBySerial()`

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

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

## See also

[spinError](#)

**Parameters**

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

**Returns**

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

## 6.11 Interface Access

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

Collaboration diagram for Interface Access:



### Functions

- [SPINNAKERC\\_API spinInterfaceUpdateCameras](#) ([spinInterface](#) hInterface, [bool8\\_t](#) \*pbChanged)  
*Checks whether any cameras have been connected or disconnected on an interface.*
- [SPINNAKERC\\_API spinInterfaceGetCameras](#) ([spinInterface](#) hInterface, [spinCameraList](#) hCameraList)  
*Retrieves a camera list from an interface; camera lists must be created and destroy.*
- [SPINNAKERC\\_API spinInterfaceGetCamerasEx](#) ([spinInterface](#) hInterface, [bool8\\_t](#) bUpdateCameras, [spinCameraList](#) hCameraList)  
*Retrieves a camera list from an interface; manually set whether to update the cameras; camera lists must be created and destroyed.*
- [SPINNAKERC\\_API spinInterfaceGetTLNodeMap](#) ([spinInterface](#) hInterface, [spinNodeMapHandle](#) \*phNodeMap)  
*Retrieves the transport layer nodemap from an interface.*
- [SPINNAKERC\\_API spinInterfaceRegisterArrivalEvent](#) ([spinInterface](#) hInterface, [spinArrivalEvent](#) hArrivalEvent)  
*Registers an arrival event on an interface (events registered in this way must be unregistered)*
- [SPINNAKERC\\_API spinInterfaceRegisterRemovalEvent](#) ([spinInterface](#) hInterface, [spinRemovalEvent](#) hRemovalEvent)  
*Registers a removal event on an interface (events registered in this way must be unregistered)*
- [SPINNAKERC\\_API spinInterfaceUnregisterArrivalEvent](#) ([spinInterface](#) hInterface, [spinArrivalEvent](#) hArrivalEvent)  
*Unregisters an arrival event from an interface.*
- [SPINNAKERC\\_API spinInterfaceUnregisterRemovalEvent](#) ([spinInterface](#) hInterface, [spinRemovalEvent](#) hRemovalEvent)  
*Unregisters a removal event from an interface.*
- [SPINNAKERC\\_API spinInterfaceRegisterInterfaceEvent](#) ([spinInterface](#) hInterface, [spinInterfaceEvent](#) hInterfaceEvent)  
*Registers an interface event (both arrival and removal) on an interface.*
- [SPINNAKERC\\_API spinInterfaceUnregisterInterfaceEvent](#) ([spinInterface](#) hInterface, [spinInterfaceEvent](#) hInterfaceEvent)  
*Unregisters an interface event 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.*

### 6.11.1 Detailed Description

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

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

### 6.11.2 Function Documentation

#### 6.11.2.1 spinInterfaceGetCameras()

```
SPINNAKERC_API spinInterfaceGetCameras (
    spinInterface hInterface,
    spinCameraList hCameraList )
```

Retrieves a camera list from an interface; camera lists must be created and destroy.

See also

[spinCameraListCreateEmpty\(\)](#)  
[spinCameraListDestroy\(\)](#)  
[spinError](#)

Parameters

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

Returns

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

#### 6.11.2.2 spinInterfaceGetCamerasEx()

```
SPINNAKERC_API spinInterfaceGetCamerasEx (
    spinInterface hInterface,
    bool8_t bUpdateCameras,
    spinCameraList hCameraList )
```

Retrieves a camera list from an interface; manually set whether to update the cameras; camera lists must be created and destroyed.

See also

[spinCameraListCreateEmpty\(\)](#)  
[spinCameraListDestroy\(\)](#)  
[spinError](#)

## Parameters

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

## Returns

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

6.11.2.3 `spinInterfaceGetTLNodeMap()`

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

Retrieves the transport layer nodemap from an interface.

## See also

[spinError](#)

## Parameters

<i>hInterface</i>	The interface of the nodemap to retrieve
<i>phNodeMap</i>	The nodemap handle pointer in which the transport layer interface nodemap is returned

## Returns

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

6.11.2.4 `spinInterfaceIsInUse()`

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

Checks whether an interface is in use.

## See also

[spinError](#)

**Parameters**

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

**6.11.2.5 spinInterfaceRegisterArrivalEvent()**

```
SPINNAKERC_API spinInterfaceRegisterArrivalEvent (
    spinInterface hInterface,
    spinArrivalEvent hArrivalEvent )
```

Registers an arrival event on an interface (events registered in this way must be unregistered)

**See also**

[spinError](#)

**Parameters**

<i>hInterface</i>	The interface on which to register the arrival event
<i>hArrivalEvent</i>	The arrival event to register

**Returns**

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

**6.11.2.6 spinInterfaceRegisterInterfaceEvent()**

```
SPINNAKERC_API spinInterfaceRegisterInterfaceEvent (
    spinInterface hInterface,
    spinInterfaceEvent hInterfaceEvent )
```

Registers an interface event (both arrival and removal) on an interface.

**See also**

[spinError](#)

## Parameters

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

## Returns

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

6.11.2.7 `spinInterfaceRegisterRemovalEvent()`

```
SPINNAKERC_API spinInterfaceRegisterRemovalEvent (
    spinInterface hInterface,
    spinRemovalEvent hRemovalEvent )
```

Registers a removal event on an interface (events registered in this way must be unregistered)

## See also

[`spinError`](#)

## Parameters

<i>hInterface</i>	the Interface on which to register the removal event
<i>hRemovalEvent</i>	The removal event to register

## Returns

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

6.11.2.8 `spinInterfaceRelease()`

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

Releases an interface.

## See also

[`spinError`](#)

## Parameters

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

**Returns**

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

**6.11.2.9 spinInterfaceSendActionCommand()**

```
SPINNAKERC_API spinInterfaceSendActionCommand (
    spinInterface hInterface,
    size_t iDeviceKey,
    size_t iGroupKey,
    size_t iGroupMask,
    size_t iActionTime,
    size_t * piResultSize,
    actionCommandResult results[] )
```

Broadcast an Action Command to all devices on interface.

**See also**

[spinError](#)

**Parameters**

<i>iDeviceKey</i>	The Action Command's device key
<i>iGroupKey</i>	The Action Command's group key
<i>iGroupMask</i>	The Action Command's group mask
<i>iActionTime</i>	(Optional) Time when to assert a future action. Zero means immediate action.
<i>piResultSize</i>	(Optional) The number of results in the results array. The value passed should be equal to the expected number of devices that acknowledge the command. Returns the number of received results.
<i>results</i>	(Optional) An Array with *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

**6.11.2.10 spinInterfaceUnregisterArrivalEvent()**

```
SPINNAKERC_API spinInterfaceUnregisterArrivalEvent (
    spinInterface hInterface,
    spinArrivalEvent hArrivalEvent )
```

Unregisters an arrival event from an interface.



See also

[spinError](#)

#### Parameters

<i>hInterface</i>	The interface from which to unregister the arrival event
<i>hArrivalEvent</i>	The arrival event to unregister

#### Returns

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

#### 6.11.2.11 [spinInterfaceUnregisterInterfaceEvent\(\)](#)

```
SPINNAKERC_API spinInterfaceUnregisterInterfaceEvent (
    spinInterface hInterface,
    spinInterfaceEvent hInterfaceEvent )
```

Unregisters an interface event from an interface.

See also

[spinError](#)

#### Parameters

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

#### Returns

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

#### 6.11.2.12 [spinInterfaceUnregisterRemovalEvent\(\)](#)

```
SPINNAKERC_API spinInterfaceUnregisterRemovalEvent (
    spinInterface hInterface,
    spinRemovalEvent hRemovalEvent )
```

Unregisters a removal event from an interface.

See also

[spinError](#)

**Parameters**

<i>hInterface</i>	The interface from which to unregister the removal event
<i>hRemovalEvent</i>	The removal event to unregister

**Returns**

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

**6.11.2.13 `spinInterfaceUpdateCameras()`**

```
SPINNAKERC_API spinInterfaceUpdateCameras (
    spinInterface hInterface,
    bool8_t * pbChanged )
```

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

**See also**

[`spinError`](#)

**Parameters**

<i>hInterface</i>	The interface of the list of attached cameras to update
<i>pbChanged</i>	The boolean pointer to return whether or not the cameras have changed

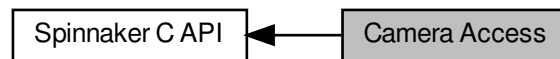
**Returns**

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

## 6.12 Camera Access

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

Collaboration diagram for Camera Access:



### Functions

- [SPINNAKERC\\_API spinCameraInit \(spinCamera hCamera\)](#)  
*Initializes a camera, allowing for much more interaction.*
- [SPINNAKERC\\_API spinCameraDeInit \(spinCamera hCamera\)](#)  
*Deinitializes a camera, greatly reducing functionality.*
- [SPINNAKERC\\_API spinCameraGetNodeMap \(spinCamera hCamera, spinNodeMapHandle \\*phNodeMap\)](#)  
*Retrieves the GenICam nodemap from a camera.*
- [SPINNAKERC\\_API spinCameraGetTLDeviceNodeMap \(spinCamera hCamera, spinNodeMapHandle \\*phNodeMap\)](#)  
*Retrieves the transport layer device nodemap from a camera.*
- [SPINNAKERC\\_API spinCameraGetTLStreamNodeMap \(spinCamera hCamera, spinNodeMapHandle \\*phNodeMap\)](#)  
*Retrieves the transport layer stream nodemap from a camera.*
- [SPINNAKERC\\_API spinCameraGetAccessMode \(spinCamera hCamera, spinAccessMode \\*pAccessMode\)](#)  
*Retrieves the access mode of a camera (as an enum, spinAccessMode)*
- [SPINNAKERC\\_API spinCameraReadPort \(spinCamera hCamera, uint64\\_t iAddress, void \\*pBuffer, size\\_t iSize\)](#)
- [SPINNAKERC\\_API spinCameraWritePort \(spinCamera hCamera, uint64\\_t iAddress, void \\*pBuffer, size\\_t iSize\)](#)
- [SPINNAKERC\\_API spinCameraBeginAcquisition \(spinCamera hCamera\)](#)  
*Has a camera start acquiring images.*
- [SPINNAKERC\\_API spinCameraEndAcquisition \(spinCamera hCamera\)](#)  
*Has a camera stop acquiring images.*
- [SPINNAKERC\\_API spinCameraGetNextImage \(spinCamera hCamera, spinImage \\*phImage\)](#)  
*Retrieves an image from a camera.*
- [SPINNAKERC\\_API spinCameraGetNextImageEx \(spinCamera hCamera, uint64\\_t grabTimeout, spinImage \\*phImage\)](#)  
*Retrieves an image from a camera; manually set the timeout in milliseconds.*
- [SPINNAKERC\\_API spinCameraGetUniqueID \(spinCamera hCamera, char \\*pBuf, size\\_t \\*pBufLen\)](#)  
*Retrieves a unique identifier for a camera.*
- [SPINNAKERC\\_API spinCameraIsStreaming \(spinCamera hCamera, bool8\\_t \\*pIsStreaming\)](#)  
*Checks whether a camera is currently acquiring images.*
- [SPINNAKERC\\_API spinCameraGetGuiXml \(spinCamera hCamera, char \\*pBuf, size\\_t \\*pBufLen\)](#)  
*Retrieves the GUI XML from a camera.*

- [SPINNAKERC\\_API spinCameraRegisterDeviceEvent](#) ([spinCamera](#) hCamera, [spinDeviceEvent](#) hDevice↔Event)  
*Registers a universal device event (every device event type) to a camera.*
- [SPINNAKERC\\_API spinCameraRegisterDeviceEventEx](#) ([spinCamera](#) hCamera, [spinDeviceEvent](#) hDevice↔Event, const char \*pName)  
*Registers a specific device event (only one device event type) to a camera.*
- [SPINNAKERC\\_API spinCameraUnregisterDeviceEvent](#) ([spinCamera](#) hCamera, [spinDeviceEvent](#) hDevice↔Event)  
*Unregisters a device event from a camera.*
- [SPINNAKERC\\_API spinCameraRegisterImageEvent](#) ([spinCamera](#) hCamera, [spinImageEvent](#) hImageEvent)  
*Registers an image event to a camera.*
- [SPINNAKERC\\_API spinCameraUnregisterImageEvent](#) ([spinCamera](#) hCamera, [spinImageEvent](#) hImage↔Event)  
*Unregisters an image event 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.*

### 6.12.1 Detailed Description

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

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

### 6.12.2 Function Documentation

#### 6.12.2.1 spinCameraBeginAcquisition()

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

Has a camera start acquiring images.

See also

[spinError](#)

Parameters

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

**Returns**

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

**6.12.2.2 spinCameraDeInit()**

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

Deinitializes a camera, greatly reducing functionality.

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

**6.12.2.3 spinCameraEndAcquisition()**

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

Has a camera stop acquiring images.

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

#### 6.12.2.4 spinCameraGetAccessMode()

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

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

See also

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

##### Parameters

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

##### Returns

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

#### 6.12.2.5 spinCameraGetGuiXml()

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

Retrieves the GUI XML from a camera.

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.12.2.6 spinCameraGetNextImage()

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

Retrieves an image from a camera.

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.12.2.7 spinCameraGetNextImageEx()

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

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

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.12.2.8 spinCameraGetNodeMap()

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

Retrieves the GenICam nodemap from a camera.

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.12.2.9 spinCameraGetTLDeviceNodeMap()

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

Retrieves the transport layer device nodemap from a camera.

See also

[spinError](#)

##### Parameters

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

##### Returns

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



#### 6.12.2.10 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

#### 6.12.2.11 spinCameraGetUniqueID()

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

Retrieves a unique identifier for a camera.

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.12.2.12 spinCameraInit()

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

Initializes a camera, allowing for much more interaction.

See also

[spinError](#)

Parameters

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

Returns

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

#### 6.12.2.13 spinCameraIsInitialized()

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

Checks whether a camera is currently initialized.

See also

[spinError](#)

Parameters

<i>hCamera</i>	The camera to check
<i>pbInit</i>	The boolean pointer to return whether or not the camera is initialized

Returns

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

#### 6.12.2.14 spinCameraIsStreaming()

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

Checks whether a camera is currently acquiring images.

See also

[spinError](#)

#### Parameters

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

#### Returns

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

#### 6.12.2.15 `spinCamerasValid()`

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

Checks whether a camera is still valid for use.

See also

[spinError](#)

#### Parameters

<i>hCamera</i>	The camera to check
<i>pbValid</i>	The boolean pointer to return whether or not the camera is valid

#### Returns

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

#### 6.12.2.16 `spinCameraReadPort()`

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

#### 6.12.2.17 spinCameraRegisterDeviceEvent()

```
SPINNAKERC_API spinCameraRegisterDeviceEvent (
    spinCamera hCamera,
    spinDeviceEvent hDeviceEvent )
```

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

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.12.2.18 spinCameraRegisterDeviceEventEx()

```
SPINNAKERC_API spinCameraRegisterDeviceEventEx (
    spinCamera hCamera,
    spinDeviceEvent hDeviceEvent,
    const char * pName )
```

Registers a specific device event (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
<i>hDeviceEvent</i>	The device event to register
<i>pName</i>	The name of the device event to register

##### Returns

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

### 6.12.2.19 spinCameraRegisterImageEvent()

```
SPINNAKERC_API spinCameraRegisterImageEvent (
    spinCamera hCamera,
    spinImageEvent hImageEvent )
```

Registers an image event to a camera.

See also

[spinError](#)

#### Parameters

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

#### Returns

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

### 6.12.2.20 spinCameraRelease()

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

Releases a camera.

See also

[spinError](#)

#### Parameters

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

#### Returns

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

### 6.12.2.21 spinCameraUnregisterDeviceEvent()

```
SPINNAKERC_API spinCameraUnregisterDeviceEvent (
    spinCamera hCamera,
    spinDeviceEvent hDeviceEvent )
```

Unregisters a device event from a camera.

See also

[spinError](#)

Parameters

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

Returns

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

#### 6.12.2.22 `spinCameraUnregisterImageEvent()`

```
SPINNAKERC_API spinCameraUnregisterImageEvent (
    spinCamera hCamera,
    spinImageEvent hImageEvent )
```

Unregisters an image event from a camera.

See also

[spinError](#)

Parameters

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

Returns

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

#### 6.12.2.23 `spinCameraWritePort()`

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

## 6.13 Image Access

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

### Functions

- [SPINNAKERC\\_API spinImageCreateEmpty](#) ([spinImage](#) \*phImage)  
*Creates an empty image; images created this way must be destroyed.*
- [SPINNAKERC\\_API spinImageCreate](#) ([spinImage](#) hSrcImage, [spinImage](#) \*phDestImage)  
*Creates an image from another; images created this way must be destroyed.*
- [SPINNAKERC\\_API spinImageCreateEx](#) ([spinImage](#) \*phImage, [size\\_t](#) width, [size\\_t](#) height, [size\\_t](#) offsetX, [size\\_t](#) offsetY, [spinPixelFormatEnums](#) pixelFormat, void \*pData)  
*Creates an image with some set properties; images created this way must be destroyed.*
- [SPINNAKERC\\_API spinImageDestroy](#) ([spinImage](#) hImage)  
*Destroys an image.*
- [SPINNAKERC\\_API spinImageSetDefaultColorProcessing](#) ([spinColorProcessingAlgorithm](#) algorithm)  
*Sets the default color processing algorithm of all images (if not otherwise set)*
- [SPINNAKERC\\_API spinImageGetDefaultColorProcessing](#) ([spinColorProcessingAlgorithm](#) \*pAlgorithm)  
*Retrieves the default color processing algorithm.*
- [SPINNAKERC\\_API spinImageGetColorProcessing](#) ([spinImage](#) hImage, [spinColorProcessingAlgorithm](#) \*pAlgorithm)  
*Retrieves the color processing algorithm of a specific image.*
- [SPINNAKERC\\_API spinImageConvert](#) ([spinImage](#) hSrcImage, [spinPixelFormatEnums](#) pixelFormat, [spinImage](#) hDestImage)  
*Converts the pixel format of one image into a new image.*
- [SPINNAKERC\\_API spinImageConvertEx](#) ([spinImage](#) hSrcImage, [spinPixelFormatEnums](#) pixelFormat, [spinColorProcessingAlgorithm](#) algorithm, [spinImage](#) hDestImage)  
*Converts the pixel format and color processing algorithm of one image into a new image.*
- [SPINNAKERC\\_API spinImageReset](#) ([spinImage](#) hImage, [size\\_t](#) width, [size\\_t](#) height, [size\\_t](#) offsetX, [size\\_t](#) offsetY, [spinPixelFormatEnums](#) pixelFormat)  
*Resets an image with some set properties.*
- [SPINNAKERC\\_API spinImageResetEx](#) ([spinImage](#) hImage, [size\\_t](#) width, [size\\_t](#) height, [size\\_t](#) offsetX, [size\\_t](#) offsetY, [spinPixelFormatEnums](#) pixelFormat, void \*pData)  
*Resets an image with some set properties and image data.*
- [SPINNAKERC\\_API spinImageGetID](#) ([spinImage](#) hImage, [uint64\\_t](#) \*pId)  
*Retrieves the ID of an image.*
- [SPINNAKERC\\_API spinImageGetData](#) ([spinImage](#) hImage, void \*\*ppData)  
*Retrieves the image data of an image.*
- [SPINNAKERC\\_API spinImageGetPrivateData](#) ([spinImage](#) hImage, void \*\*ppData)  
*Retrieves the private data of an image.*
- [SPINNAKERC\\_API spinImageGetBufferSize](#) ([spinImage](#) hImage, [size\\_t](#) \*pSize)  
*Retrieves the buffer size of an image.*
- [SPINNAKERC\\_API spinImageDeepCopy](#) ([spinImage](#) hSrcImage, [spinImage](#) hDestImage)  
*Creates a deep copy of an image (the destination image must be created as an empty image prior to the deep copy)*
- [SPINNAKERC\\_API spinImageGetWidth](#) ([spinImage](#) hImage, [size\\_t](#) \*pWidth)  
*Retrieves the width of an image.*
- [SPINNAKERC\\_API spinImageGetHeight](#) ([spinImage](#) hImage, [size\\_t](#) \*pHeight)  
*Retrieves the height of an image.*
- [SPINNAKERC\\_API spinImageGetOffsetX](#) ([spinImage](#) hImage, [size\\_t](#) \*pOffsetX)  
*Retrieves the offset of an image along its X axis.*

- [SPINNAKERC\\_API spinImageGetOffsetY](#) ([spinImage](#) hImage, [size\\_t](#) \*pOffsetY)  
*Retrieves the offset of an image along its Y axis.*
- [SPINNAKERC\\_API spinImageGetPaddingX](#) ([spinImage](#) hImage, [size\\_t](#) \*pPaddingX)  
*Retrieves the padding of an image along its X axis.*
- [SPINNAKERC\\_API spinImageGetPaddingY](#) ([spinImage](#) hImage, [size\\_t](#) \*pPaddingY)  
*Retrieves the padding of an image along its Y axis.*
- [SPINNAKERC\\_API spinImageGetFrameID](#) ([spinImage](#) hImage, [uint64\\_t](#) \*pFrameID)  
*Retrieves the frame ID of an image.*
- [SPINNAKERC\\_API spinImageGetTimeStamp](#) ([spinImage](#) hImage, [uint64\\_t](#) \*pTimeStamp)  
*Retrieves the timestamp of an image.*
- [SPINNAKERC\\_API spinImageGetPayloadType](#) ([spinImage](#) hImage, [size\\_t](#) \*pPayloadType)  
*Retrieves the payload type of an image (as an enum, [spinPayloadTypeInfos](#))*
- [SPINNAKERC\\_API spinImageGetTLPayloadType](#) ([spinImage](#) hImage, [spinPayloadTypeInfoIDs](#) \*pPayloadType)  
*Retrieves the transport layer payload type of an image (as an enum, [spinPayloadTypeInfos](#))*
- [SPINNAKERC\\_API spinImageGetPixelFormat](#) ([spinImage](#) hImage, [spinPixelFormatEnums](#) \*pPixelFormat)  
*Retrieves the pixel format of an image (as an enum, [spinPixelFormatEnums](#))*
- [SPINNAKERC\\_API spinImageGetTLPixelFormat](#) ([spinImage](#) hImage, [uint64\\_t](#) \*pPixelFormat)  
*Retrieves the transport layer pixel format of an image (as an unsigned integer)*
- [SPINNAKERC\\_API spinImageGetTLPixelFormatNamespace](#) ([spinImage](#) hImage, [spinPixelFormatNamespaceID](#) \*pPixelFormatNamespace)  
*Retrieves the transport layer pixel format namespace of an image (as an enum, [spinPixelFormatNamespaceID](#))*
- [SPINNAKERC\\_API spinImageGetPixelFormatName](#) ([spinImage](#) hImage, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves the pixel format of an image (as a symbolic)*
- [SPINNAKERC\\_API spinImageIsIncomplete](#) ([spinImage](#) hImage, [bool8\\_t](#) \*pIsIncomplete)  
*Checks whether an image is incomplete.*
- [SPINNAKERC\\_API spinImageGetValidPayloadSize](#) ([spinImage](#) hImage, [size\\_t](#) \*pSize)  
*Retrieves the valid payload size of an image.*
- [SPINNAKERC\\_API spinImageSave](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [spinImageFileFormat](#) format)  
*Saves an image using a specified file format (using an enum, [spinImageFileFormat](#))*
- [SPINNAKERC\\_API spinImageSaveFromExt](#) ([spinImage](#) hImage, [const char](#) \*pFilename)  
*Saves an image using a specified file format (using the extension of the filename)*
- [SPINNAKERC\\_API spinImageSavePng](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinPNGOption](#) \*pOption)  
*Saves an image as a PNG image.*
- [SPINNAKERC\\_API spinImageSavePpm](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinPPMOption](#) \*pOption)  
*Saves an image as a PPM image.*
- [SPINNAKERC\\_API spinImageSavePgm](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinPGMOption](#) \*pOption)  
*Saves an image as an PGM image.*
- [SPINNAKERC\\_API spinImageSaveTiff](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinTIFFOption](#) \*pOption)  
*Saves an image as a TIFF image.*
- [SPINNAKERC\\_API spinImageSaveJpeg](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinJPEGOption](#) \*pOption)  
*Saves an image as a JPEG image.*
- [SPINNAKERC\\_API spinImageSaveJpg2](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinJPG2Option](#) \*pOption)  
*Saves an image as a JPEG 2000 image.*



- [SPINNAKERC\\_API spinImageSaveBmp](#) ([spinImage](#) hImage, const char \*pFilename, const [spinBMPOption](#) \*pOption)  
*Saves an image as a BMP image.*
- [SPINNAKERC\\_API spinImageGetChunkLayoutID](#) ([spinImage](#) hImage, uint64\_t \*pId)  
*Retrieves the chunk layout ID of an image.*
- [SPINNAKERC\\_API spinImageCalculateStatistics](#) ([spinImage](#) hImage, const [spinImageStatistics](#) hStatistics)  
*Calculates the image statistics of an image.*
- [SPINNAKERC\\_API spinImageGetStatus](#) ([spinImage](#) hImage, [spinImageStatus](#) \*pStatus)  
*Retrieves the image status of an image.*
- [SPINNAKERC\\_API spinImageGetStatusDescription](#) ([spinImageStatus](#) status, char \*pBuf, size\_t \*pBufLen)  
*Retrieves the description of image status.*
- [SPINNAKERC\\_API spinImageRelease](#) ([spinImage](#) hImage)  
*Releases an image.*
- [SPINNAKERC\\_API spinImageHasCRC](#) ([spinImage](#) hImage, bool8\_t \*pbHasCRC)  
*Checks whether an image has CRC.*
- [SPINNAKERC\\_API spinImageCheckCRC](#) ([spinImage](#) hImage, bool8\_t \*pbCheckCRC)  
*Checks whether the CRC of an image is correct.*
- [SPINNAKERC\\_API spinImageGetBitsPerPixel](#) ([spinImage](#) hImage, size\_t \*pBitsPerPixel)  
*Retrieves the number of bits per pixel of an image.*
- [SPINNAKERC\\_API spinImageGetSize](#) ([spinImage](#) hImage, size\_t \*pImageSize)  
*Retrieves the size of an image.*
- [SPINNAKERC\\_API spinImageGetStride](#) ([spinImage](#) hImage, size\_t \*pStride)  
*Retrieves the stride of an image.*

### 6.13.1 Detailed Description

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

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

### 6.13.2 Function Documentation

#### 6.13.2.1 spinImageCalculateStatistics()

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

Calculates the image statistics of an image.

See also

[spinError](#)

## Parameters

<i>hImage</i>	The image to be saved
<i>hStatistics</i>	The image statistics context in which the calculated statistics are returned

## Returns

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

6.13.2.2 `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

6.13.2.3 `spinImageConvert()`

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

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

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.13.2.4 `spinImageConvertEx()`

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

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

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.13.2.5 `spinImageCreate()`

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

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

## See also

[spinError](#)

**Parameters**

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

**Returns**

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

**6.13.2.6 spinImageCreateEmpty()**

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

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

**See also**

[spinError](#)

**Parameters**

<i>phImage</i>	The image handle pointer in which the empty image is returned
----------------	---

**Returns**

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

**6.13.2.7 spinImageCreateEx()**

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

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

**See also**

[spinError](#)

## Parameters

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

## Returns

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

6.13.2.8 `spinImageDeepCopy()`

```
SPINNAKER_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

6.13.2.9 `spinImageDestroy()`

```
SPINNAKER_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

6.13.2.10 `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

6.13.2.11 `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

**6.13.2.12 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

**6.13.2.13 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

**6.13.2.14 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

**6.13.2.15 spinImageGetDefaultColorProcessing()**

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

Retrieves the default color processing algorithm.

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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



## 6.13.2.16 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

## 6.13.2.17 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

#### 6.13.2.18 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

#### 6.13.2.19 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

## 6.13.2.20 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

## 6.13.2.21 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

#### 6.13.2.22 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

#### 6.13.2.23 spinImageGetPayloadType()

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

Retrieves the payload type of an image (as an enum, [spinPayloadTypeIn folds](#))

See also

[spinError](#)

[spinPayloadTypeIn folds](#)

##### Parameters

<i>hImage</i>	The image of the payload type to retrieve
<i>pPayloadType</i>	The payload type enum pointer in which the payload type is returned

##### Returns

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

## 6.13.2.24 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

## 6.13.2.25 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

#### 6.13.2.26 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

#### 6.13.2.27 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

#### 6.13.2.28 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

#### 6.13.2.29 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

### 6.13.2.30 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

### 6.13.2.31 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



#### 6.13.2.32 spinImageGetTLPayloadType()

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

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

See also

[spinError](#)  
[spinPayloadTypeInfolDs](#)

Parameters

<i>hImage</i>	The image of the TL payload type to retrieve
<i>pPayloadType</i>	The payload type enum pointer in which the TL payload type is returned

Returns

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

#### 6.13.2.33 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

#### 6.13.2.34 spinImageGetTLPixelFormatNamespace()

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

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

See also

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

##### Parameters

<i>hImage</i>	The image of the TL pixel format namespace to retrieve
<i>pPixelFormatNamespace</i>	The pixel format namespace pointer in which the pixel format namespace is returned

##### Returns

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

#### 6.13.2.35 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

## 6.13.2.36 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

## 6.13.2.37 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

#### 6.13.2.38 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

#### 6.13.2.39 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

#### 6.13.2.40 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

#### 6.13.2.41 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

**6.13.2.42 spinImageSave()**

```
SPINNAKERC_API spinImageSave (
    spinImage hImage,
    const char * pFilename,
    spinImageFileFormat format )
```

Saves an image using a specified file format (using an enum, `spinImageFileFormat`)

**See also**

[spinError](#)  
[spinImageFileFormat](#)

**Parameters**

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

**Returns**

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

**6.13.2.43 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

**6.13.2.44 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

**6.13.2.45 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

**6.13.2.46 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

**6.13.2.47 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

**6.13.2.48 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

**6.13.2.49 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

**6.13.2.50 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

**6.13.2.51 spinImageSetDefaultColorProcessing()**

```
SPINNAKERC_API spinImageSetDefaultColorProcessing (
    spinColorProcessingAlgorithm algorithm )
```

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

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

## 6.14 Event Access

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

### Functions

- [SPINNAKERC\\_API spinDeviceEventCreate](#) ([spinDeviceEvent](#) \*phDeviceEvent, [spinDeviceEventFunction](#) pFunction, void \*pUserData)  
*Creates a device event.*
- [SPINNAKERC\\_API spinDeviceEventDestroy](#) ([spinDeviceEvent](#) hDeviceEvent)  
*Destroys a device event.*
- [SPINNAKERC\\_API spinImageEventCreate](#) ([spinImageEvent](#) \*phImageEvent, [spinImageEventFunction](#) pFunction, void \*pUserData)  
*Creates an image event.*
- [SPINNAKERC\\_API spinImageEventDestroy](#) ([spinImageEvent](#) hImageEvent)  
*Destroys an image event.*
- [SPINNAKERC\\_API spinArrivalEventCreate](#) ([spinArrivalEvent](#) \*phArrivalEvent, [spinArrivalEventFunction](#) pFunction, void \*pUserData)  
*Creates an arrival event.*
- [SPINNAKERC\\_API spinArrivalEventDestroy](#) ([spinArrivalEvent](#) hArrivalEvent)  
*Destroys an arrival event.*
- [SPINNAKERC\\_API spinRemovalEventCreate](#) ([spinRemovalEvent](#) \*phRemovalEvent, [spinRemovalEventFunction](#) pFunction, void \*pUserData)  
*Creates a removal event.*
- [SPINNAKERC\\_API spinRemovalEventDestroy](#) ([spinRemovalEvent](#) hRemovalEvent)  
*Destroys a removal event.*
- [SPINNAKERC\\_API spinInterfaceEventCreate](#) ([spinInterfaceEvent](#) \*phInterfaceEvent, [spinArrivalEventFunction](#) pArrivalFunction, [spinRemovalEventFunction](#) pRemovalFunction, void \*pUserData)  
*Creates an interface event (both arrival and removal)*
- [SPINNAKERC\\_API spinInterfaceEventDestroy](#) ([spinInterfaceEvent](#) hInterfaceEvent)  
*Destroys an interface event (both arrival and removal)*
- [SPINNAKERC\\_API spinLogEventCreate](#) ([spinLogEvent](#) \*phLogEvent, [spinLogEventFunction](#) pFunction, void \*pUserData)  
*Creates a log event.*
- [SPINNAKERC\\_API spinLogEventDestroy](#) ([spinLogEvent](#) hLogEvent)  
*Destroys a log event.*

### 6.14.1 Detailed Description

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

### 6.14.2 Function Documentation

#### 6.14.2.1 spinArrivalEventCreate()

```
SPINNAKERC_API spinArrivalEventCreate (
    spinArrivalEvent * phArrivalEvent,
    spinArrivalEventFunction pFunction,
    void * pUserData )
```

Creates an arrival event.

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.14.2.2 spinArrivalEventDestroy()

```
SPINNAKERC_API spinArrivalEventDestroy (
    spinArrivalEvent hArrivalEvent )
```

Destroys an arrival event.

See also

[spinError](#)

##### Parameters

<i>hArrivalEvent</i>	The arrival event to destroy
----------------------	------------------------------

##### Returns

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

#### 6.14.2.3 spinDeviceEventCreate()

```
SPINNAKERC_API spinDeviceEventCreate (
    spinDeviceEvent * phDeviceEvent,
    spinDeviceEventFunction pFunction,
    void * pUserData )
```

Creates a device event.

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.14.2.4 spinDeviceEventDestroy()

```
SPINNAKERC_API spinDeviceEventDestroy (
    spinDeviceEvent hDeviceEvent )
```

Destroys a device event.

See also

[spinError](#)

##### Parameters

<i>hDeviceEvent</i>	The device event to destroy
---------------------	-----------------------------

##### Returns

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

#### 6.14.2.5 spinImageEventCreate()

```
SPINNAKERC_API spinImageEventCreate (
    spinImageEvent * phImageEvent,
    spinImageEventFunction pFunction,
    void * pUserData )
```

Creates an image event.

See also

[spinError](#)

##### Parameters

<i>phImageEvent</i>	The image event handle pointer in which the image event context is created
<i>pFunction</i>	The function to be called at image event occurrences; signature to match: void(<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

#### 6.14.2.6 spinImageEventDestroy()

```
SPINNAKERC_API spinImageEventDestroy (
    spinImageEvent hImageEvent )
```

Destroys an image event.

See also

[spinError](#)

##### Parameters

<i>hImageEvent</i>	The image event to destroy
--------------------	----------------------------

##### Returns

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

#### 6.14.2.7 spinInterfaceEventCreate()

```
SPINNAKERC_API spinInterfaceEventCreate (
    spinInterfaceEvent * phInterfaceEvent,
    spinArrivalEventFunction pArrivalFunction,
    spinRemovalEventFunction pRemovalFunction,
    void * pUserData )
```

Creates an interface event (both arrival and removal)

See also

[spinError](#)

##### Parameters

<i>phInterfaceEvent</i>	The interface event handle pointer in which the interface event context is created
<i>pArrivalFunction</i>	The function to be called at arrival event occurrences; signature to match: void(<em>spinArrivalEventFunction)(void pUserData)
<i>hRemovalFunction</i>	The function to be called at removal 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

#### 6.14.2.8 spinInterfaceEventDestroy()

```
SPINNAKERC_API spinInterfaceEventDestroy (
    spinInterfaceEvent hInterfaceEvent )
```

Destroys an interface event (both arrival and removal)

See also

[spinError](#)

##### Parameters

<i>hInterfaceEvent</i>	The interface event to destroy
------------------------	--------------------------------

##### Returns

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



## 6.14.2.9 spinLogEventCreate()

```
SPINNAKERC_API spinLogEventCreate (
    spinLogEvent * phLogEvent,
    spinLogEventFunction pFunction,
    void * pUserData )
```

Creates a log event.

See also

[spinError](#)

## Parameters

<i>phLogEvent</i>	The log event handle pointer in which the log event context is created
<i>pFunction</i>	The function to be called at device event occurrences; signature to match: void(<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

## 6.14.2.10 spinLogEventDestroy()

```
SPINNAKERC_API spinLogEventDestroy (
    spinLogEvent hLogEvent )
```

Destroys a log event.

See also

[spinError](#)

## Parameters

<i>hLogEvent</i>	The log event to destroy
------------------	--------------------------

## Returns

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

#### 6.14.2.11 spinRemovalEventCreate()

```
SPINNAKERC_API spinRemovalEventCreate (
    spinRemovalEvent * phRemovalEvent,
    spinRemovalEventFunction pFunction,
    void * pUserData )
```

Creates a removal event.

See also

[spinError](#)

##### Parameters

<i>phRemovalEvent</i>	The removal event handle pointer in which the 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

#### 6.14.2.12 spinRemovalEventDestroy()

```
SPINNAKERC_API spinRemovalEventDestroy (
    spinRemovalEvent hRemovalEvent )
```

Destroys a removal event.

See also

[spinError](#)

##### Parameters

<i>hRemovalEvent</i>	The removal event to destroy
----------------------	------------------------------

##### Returns

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

## 6.15 ImageStatistics Access

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

### Functions

- [SPINNAKERC\\_API spinImageStatisticsCreate](#) ([spinImageStatistics](#) \*phStatistics)  
*Creates an image statistics context.*
- [SPINNAKERC\\_API spinImageStatisticsDestroy](#) ([spinImageStatistics](#) hStatistics)  
*Destroys an image statistics context.*
- [SPINNAKERC\\_API spinImageStatisticsEnableAll](#) ([spinImageStatistics](#) hStatistics)  
*Enables all channels of an image statistics context.*
- [SPINNAKERC\\_API spinImageStatisticsDisableAll](#) ([spinImageStatistics](#) hStatistics)  
*Disables all channels of an image statistics context.*
- [SPINNAKERC\\_API spinImageStatisticsEnableGreyOnly](#) ([spinImageStatistics](#) hStatistics)  
*Disables all channels of an image statistics context except grey-scale.*
- [SPINNAKERC\\_API spinImageStatisticsEnableRgbOnly](#) ([spinImageStatistics](#) hStatistics)  
*Disables all channels of an image statistics context except red, blue, and green.*
- [SPINNAKERC\\_API spinImageStatisticsEnableHslOnly](#) ([spinImageStatistics](#) hStatistics)  
*Disables all channels of an image statistics context except hue, saturation, and lightness.*
- [SPINNAKERC\\_API spinImageStatisticsGetChannelStatus](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, [bool8\\_t](#) \*pbEnabled)  
*Checks whether an image statistics context is enabled.*
- [SPINNAKERC\\_API spinImageStatisticsSetChannelStatus](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, [bool8\\_t](#) bEnable)  
*Sets the status of an image statistics channel.*
- [SPINNAKERC\\_API spinImageStatisticsGetRange](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, unsigned int \*pMin, unsigned int \*pMax)  
*Retrieves the range of an image statistics channel.*
- [SPINNAKERC\\_API spinImageStatisticsGetPixelValueRange](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, unsigned int \*pMin, unsigned int \*pMax)  
*Retrieves the pixel value range of an image statistics channel.*
- [SPINNAKERC\\_API spinImageStatisticsGetNumPixelValues](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, unsigned int \*pNumValues)  
*Retrieves the number of pixel values of an image statistics channel.*
- [SPINNAKERC\\_API spinImageStatisticsGetMean](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, float \*pMean)  
*Retrieves the mean of pixel values of an image statistics channel.*
- [SPINNAKERC\\_API spinImageStatisticsGetHistogram](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, int \*\*ppHistogram)  
*Retrieves a histogram of an image statistics channel.*
- [SPINNAKERC\\_API spinImageStatisticsGetAll](#) ([spinImageStatistics](#) hStatistics, [spinStatisticsChannel](#) channel, unsigned int \*pRangeMin, unsigned int \*pRangeMax, unsigned int \*pPixelValueMin, unsigned int \*pPixelValueMax, unsigned int \*pNumPixelValues, float \*pPixelValueMean, int \*\*ppHistogram)  
*Retrieves all available information of an image statistics channel.*

### 6.15.1 Detailed Description

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

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

## 6.15.2 Function Documentation

### 6.15.2.1 spinImageStatisticsCreate()

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

Creates an image statistics context.

#### Parameters

<i>phStatistics</i>	The statistics handle pointer in which the image statistics context is returned
---------------------	---

#### Returns

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

### 6.15.2.2 spinImageStatisticsDestroy()

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

Destroys an image statistics context.

#### See also

[spinError](#)

#### Parameters

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

#### Returns

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

### 6.15.2.3 spinImageStatisticsDisableAll()

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

Disables all channels of an image statistics context.

See also

[spinError](#)

Parameters

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

Returns

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

#### 6.15.2.4 spinImageStatisticsEnableAll()

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

Enables all channels of an image statistics context.

See also

[spinError](#)

Parameters

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

Returns

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

#### 6.15.2.5 spinImageStatisticsEnableGreyOnly()

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

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

See also

[spinError](#)

**Parameters**

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

**Returns**

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

**6.15.2.6 spinImageStatisticsEnableHslOnly()**

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

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

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

**6.15.2.7 spinImageStatisticsEnableRgbOnly()**

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

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

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

**6.15.2.8 spinImageStatisticsGetAll()**

```
SPINNAKERC_API spinImageStatisticsGetAll (
    spinImageStatistics hStatistics,
    spinStatisticsChannel channel,
    unsigned int * pRangeMin,
    unsigned int * pRangeMax,
    unsigned int * pPixelValueMin,
    unsigned int * pPixelValueMax,
    unsigned int * pNumPixelValues,
    float * pPixelValueMean,
    int ** ppHistogram )
```

Retrieves all available information of an image statistics channel.

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

**6.15.2.9 spinImageStatisticsGetChannelStatus()**

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

Checks whether an image statistics context is enabled.

See also

[spinError](#)

#### Parameters

<i>hStatistics</i>	The image statistics context of the channel
<i>channel</i>	The channel to check
<i>pbEnabled</i>	The boolean pointer to return whether or not the channel is enabled

#### Returns

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

#### 6.15.2.10 spinImageStatisticsGetHistogram()

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

Retrieves a histogram of an image statistics channel.

See also

[spinError](#)

#### Parameters

<i>hStatistics</i>	The image statistics context of the channel
<i>channel</i>	The channel of the histogram to be returned
<i>pHistogram</i>	The pointer to the integer pointer in which the histogram data is returned

#### Returns

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

#### 6.15.2.11 spinImageStatisticsGetMean()

```
SPINNAKERC_API spinImageStatisticsGetMean (
    spinImageStatistics hStatistics,
    spinStatisticsChannel channel,
    float * pMean )
```

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



See also

[spinError](#)

#### Parameters

<i>hStatistics</i>	The image statistics context of the channel
<i>channel</i>	The channel of the mean pixel value to be retrieved
<i>pMean</i>	The float pointer in which the mean pixel value is returned

#### Returns

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

#### 6.15.2.12 `spinImageStatisticsGetNumPixelValues()`

```
SPINNAKERC_API spinImageStatisticsGetNumPixelValues (
    spinImageStatistics hStatistics,
    spinStatisticsChannel channel,
    unsigned int * pNumValues )
```

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

See also

[spinError](#)

#### Parameters

<i>hStatistics</i>	The image statistics context of the channel
<i>channel</i>	The channel where the pixel values to be counted are
<i>iNumValues</i>	The unsigned integer pointer in which the number of pixel values is returned

#### Returns

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

#### 6.15.2.13 `spinImageStatisticsGetPixelValueRange()`

```
SPINNAKERC_API spinImageStatisticsGetPixelValueRange (
    spinImageStatistics hStatistics,
    spinStatisticsChannel channel,
    unsigned int * pMin,
    unsigned int * pMax )
```

Retrieves the pixel value range of an image statistics channel.

See also

[spinError](#)

#### Parameters

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

#### Returns

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

#### 6.15.2.14 spinImageStatisticsGetRange()

```
SPINNAKERC_API spinImageStatisticsGetRange (
    spinImageStatistics hStatistics,
    spinStatisticsChannel channel,
    unsigned int * pMin,
    unsigned int * pMax )
```

Retrieves the range of an image statistics channel.

See also

[spinError](#)

#### Parameters

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

#### Returns

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

#### 6.15.2.15 spinImageStatisticsSetChannelStatus()

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

Sets the status of an image statistics channel.

See also

[spinError](#)

#### Parameters

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

#### Returns

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

## 6.16 Logging Event Data Access

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

### Functions

- [SPINNAKERC\\_API spinLogDataGetCategoryName](#) ([spinLogEventData](#) hLogEventData, char \*pBuf, size\_t \*pBufLen)  
*Retrieves the category name of a log event.*
- [SPINNAKERC\\_API spinLogDataGetPriority](#) ([spinLogEventData](#) hLogEventData, int64\_t \*pValue)  
*Retrieves the priority of a log event.*
- [SPINNAKERC\\_API spinLogDataGetPriorityName](#) ([spinLogEventData](#) hLogEventData, char \*pBuf, size\_t \*pBufLen)  
*Retrieves the priority name of a log event.*
- [SPINNAKERC\\_API spinLogDataGetTimestamp](#) ([spinLogEventData](#) hLogEventData, char \*pBuf, size\_t \*pBufLen)  
*Retrieves the timestamp of a log event.*
- [SPINNAKERC\\_API spinLogDataGetNDC](#) ([spinLogEventData](#) hLogEventData, char \*pBuf, size\_t \*pBufLen)  
*Retrieves the NDC of a log event.*
- [SPINNAKERC\\_API spinLogDataGetThreadName](#) ([spinLogEventData](#) hLogEventData, char \*pBuf, size\_t \*pBufLen)  
*Retrieves the thread name of a log event.*
- [SPINNAKERC\\_API spinLogDataGetLogMessage](#) ([spinLogEventData](#) hLogEventData, char \*pBuf, size\_t \*pBufLen)  
*Retrieves the log message of a log event.*

### 6.16.1 Detailed Description

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

### 6.16.2 Function Documentation

#### 6.16.2.1 spinLogDataGetCategoryName()

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

Retrieves the category name of a log event.

See also

[spinError](#)

## Parameters

<i>hLogEventData</i>	The log event data received from the log event
<i>pBuf</i>	The c-string character buffer in which the category name of the log event is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

## Returns

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

6.16.2.2 `spinLogDataGetLogMessage()`

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

Retrieves the log message of a log event.

## See also

[spinError](#)

## Parameters

<i>hLogEventData</i>	The log event data received from the log event
<i>pBuf</i>	The c-string character buffer in which the log message of the log event is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

## Returns

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

6.16.2.3 `spinLogDataGetNDC()`

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

Retrieves the NDC of a log event.

## See also

[spinError](#)

## Parameters

<i>hLogEventData</i>	The log event data received from the log event
<i>pBuf</i>	The c-string character buffer in which the NDC of the log event is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

## Returns

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

6.16.2.4 `spinLogDataGetPriority()`

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

Retrieves the priority of a log event.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.16.2.5 `spinLogDataGetPriorityName()`

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

Retrieves the priority name of a log event.

## See also

[spinError](#)

## Parameters

<i>hLogEventData</i>	The log event data received from the log event
<i>pBuf</i>	The c-string character buffer in which the priority name of the log event is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

## Returns

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

6.16.2.6 `spinLogDataGetThreadName()`

```
SPINNAKER_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

6.16.2.7 `spinLogDataGetTimestamp()`

```
SPINNAKER_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



## 6.17 Device Event Data Access

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

### Functions

- [SPINNAKERC\\_API spinDeviceEventGetId](#) ([spinDeviceEventData](#) hDeviceEventData, [uint64\\_t](#) \*pEventId)  
*Retrieves the event ID of a device event.*
- [SPINNAKERC\\_API spinDeviceEventGetPayloadData](#) ([spinDeviceEventData](#) hDeviceEventData, [const uint8\\_t](#) \*pBuf, [size\\_t](#) \*pBufSize)  
*Retrieves the payload data of a device event.*
- [SPINNAKERC\\_API spinDeviceEventGetPayloadDataSize](#) ([spinDeviceEventData](#) hDeviceEventData, [size\\_t](#) \*pBufSize)  
*Retrieves the payload data size of a device event.*
- [SPINNAKERC\\_API spinDeviceEventGetName](#) ([spinDeviceEventData](#) hDeviceEventData, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves the event name of a device event.*

### 6.17.1 Detailed Description

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

### 6.17.2 Function Documentation

#### 6.17.2.1 spinDeviceEventGetId()

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

Retrieves the event ID of a device event.

See also

[spinError](#)

#### Parameters

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

#### Returns

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

### 6.17.2.2 spinDeviceEventGetName()

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

Retrieves the event name of a device event.

See also

[spinError](#)

#### Parameters

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

#### Returns

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

### 6.17.2.3 spinDeviceEventGetPayloadData()

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

Retrieves the payload data of a device event.

See also

[spinError](#)

#### Parameters

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

**Returns**

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

**6.17.2.4 spinDeviceEventGetPayloadDataSize()**

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

Retrieves the payload data size of a device event.

**See also**

[`spinError`](#)

**Parameters**

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

**Returns**

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

## 6.18 AVIRecorder Access

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

### Functions

- [SPINNAKERC\\_API\\_DEPRECATED](#) ("spinAVIRecorderOpenUncompressed is deprecated, use [spinVideoOpenUncompressed](#) instead.", spinAVIRecorderOpenUncompressed([spinAVIRecorder](#) \*phRecorder, const char \*pName, [spinAVIOption](#) option))
- [SPINNAKERC\\_API\\_DEPRECATED](#) ("spinAVIRecorderOpenMJPEG is deprecated, use [spinVideoOpenMJPEG](#) instead.", spinAVIRecorderOpenMJPEG([spinAVIRecorder](#) \*phRecorder, const char \*pName, [spinMJPEGOption](#) option))
- [SPINNAKERC\\_API\\_DEPRECATED](#) ("spinAVIRecorderOpenH264 is deprecated, use [spinVideoOpenH264](#) instead.", spinAVIRecorderOpenH264([spinAVIRecorder](#) \*phRecorder, const char \*pName, [spinH264Option](#) option))
- [SPINNAKERC\\_API\\_DEPRECATED](#) ("spinAVIRecorderAppend is deprecated, use [spinVideoAppend](#) instead.", spinAVIRecorderAppend([spinAVIRecorder](#) hRecorder, [spinImage](#) hImage))
- [SPINNAKERC\\_API\\_DEPRECATED](#) ("spinAVISetMaximumSize is deprecated, use [spinVideoSetMaximumFileSize](#) instead.", spinAVISetMaximumSize([spinAVIRecorder](#) hRecorder, unsigned int size))  
*Set the maximum file size (in megabytes) of a AVI/MP4 file.*
- [SPINNAKERC\\_API\\_DEPRECATED](#) ("spinAVIRecorderClose is deprecated, use [spinVideoClose](#) instead.", spinAVIRecorderClose([spinAVIRecorder](#) hRecorder))

### 6.18.1 Detailed Description

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

NOTE: This class is deprecated and replaced by SpinVideo. Refer to [SpinVideoC.h](#) instead.

### 6.18.2 Function Documentation

#### 6.18.2.1 SPINNAKERC\_API\_DEPRECATED() [1/6]

```
SPINNAKERC_API_DEPRECATED (
    "spinAVIRecorderOpenUncompressed is deprecated,
    use spinVideoOpenUncompressed instead." ,
    spinAVIRecorderOpenUncompressed(spinAVIRecorder *phRecorder, const char *pName,
spinAVIOption option) )
```

## 6.18.2.2 SPINNAKERC\_API\_DEPRECATED() [2/6]

```
SPINNAKERC_API_DEPRECATED (
    "spinAVIRecorderOpenMJPEG is deprecated,
    use spinVideoOpenMJPEG instead." ,
    spinAVIRecorderOpenMJPEG (spinAVIRecorder *phRecorder, const char *pName, spinMJPEG↵
Option option) )
```

## 6.18.2.3 SPINNAKERC\_API\_DEPRECATED() [3/6]

```
SPINNAKERC_API_DEPRECATED (
    "spinAVIRecorderOpenH264 is deprecated,
    use spinVideoOpenH264 instead." ,
    spinAVIRecorderOpenH264 (spinAVIRecorder *phRecorder, const char *pName, spinH264↵
Option option) )
```

## 6.18.2.4 SPINNAKERC\_API\_DEPRECATED() [4/6]

```
SPINNAKERC_API_DEPRECATED (
    "spinAVIRecorderAppend is deprecated,
    use spinVideoAppend instead." ,
    spinAVIRecorderAppend (spinAVIRecorder hRecorder, spinImage hImage) )
```

## 6.18.2.5 SPINNAKERC\_API\_DEPRECATED() [5/6]

```
SPINNAKERC_API_DEPRECATED (
    "spinAVISetMaximumSize is deprecated,
    use spinVideoSetMaximumFileSize instead." ,
    spinAVISetMaximumSize (spinAVIRecorder hRecorder, 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>spinAVIRecorder</i>	The AVI recorder to append the image to
<i>size</i>	The maximum AVI file size in MB.

## Returns

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

#### 6.18.2.6 SPINNAKERC\_API\_DEPRECATED() [6/6]

```
SPINNAKERC_API_DEPRECATED (
    "spinAVIRecorderClose is deprecated,
    use spinVideoClose instead." ,
    spinAVIRecorderClose(spinAVIRecorder hRecorder) )
```

## 6.19 Chunk data access

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

### Functions

- [SPINNAKERC\\_API spinImageChunkDataGetIntValue](#) ([spinImage](#) hImage, const char \*pName, int64\_t \*pValue)
- [SPINNAKERC\\_API spinImageChunkDataGetFloatValue](#) ([spinImage](#) hImage, const char \*pName, double \*pValue)

### 6.19.1 Detailed Description

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

### 6.19.2 Function Documentation

#### 6.19.2.1 spinImageChunkDataGetFloatValue()

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

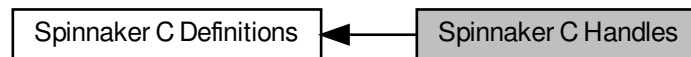
#### 6.19.2.2 spinImageChunkDataGetIntValue()

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

## 6.20 Spinnaker C Handles

Spinnaker C handle definitions.

Collaboration diagram for Spinnaker C Handles:



### Typedefs

- typedef void \* [spinSystem](#)  
*Handle for system functionality.*
- typedef void \* [spinInterfaceList](#)  
*Handle for interface list functionality.*
- typedef void \* [spinInterface](#)  
*Handle for interface functionality.*
- typedef void \* [spinCameraList](#)  
*Handle for interface functionality.*
- typedef void \* [spinCamera](#)  
*Handle for camera functionality.*
- typedef void \* [spinImage](#)  
*Handle for image functionality.*
- typedef void \* [spinImageStatistics](#)  
*Handle for image statistics functionality.*
- typedef void \* [spinDeviceEvent](#)  
*Handle for device event functionality.*
- typedef void \* [spinImageEvent](#)  
*Handle for image event functionality.*
- typedef void \* [spinArrivalEvent](#)  
*Handle for arrival event functionality.*
- typedef void \* [spinRemovalEvent](#)  
*Handle for removal event functionality.*
- typedef void \* [spinInterfaceEvent](#)  
*Handle for interface event functionality.*
- typedef void \* [spinLogEvent](#)  
*Handle for logging event functionality.*
- typedef void \* [spinLogEventData](#)  
*Handle for logging event data functionality.*
- typedef void \* [spinDeviceEventData](#)  
*Handle for device event data functionality.*
- typedef void \* [spinAVIRecorder](#)  
*Handle for video recording functionality.*
- typedef void \* [spinVideo](#)



### 6.20.1 Detailed Description

Spinnaker C handle definitions.

### 6.20.2 Typedef Documentation

#### 6.20.2.1 spinArrivalEvent

```
typedef void* spinArrivalEvent
```

Handle for arrival event functionality.

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

#### 6.20.2.2 spinAVIRecorder

```
typedef void* spinAVIRecorder
```

Handle for video recording functionality.

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

Note: spinAVIRecorder is deprecated, use spinVideo instead.

#### 6.20.2.3 spinCamera

```
typedef void* spinCamera
```

Handle for camera functionality.

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

#### 6.20.2.4 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.

#### 6.20.2.5 spinDeviceEvent

```
typedef void* spinDeviceEvent
```

Handle for device event functionality.

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

#### 6.20.2.6 spinDeviceEventData

```
typedef void* spinDeviceEventData
```

Handle for device event data functionality.

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

#### 6.20.2.7 spinImage

```
typedef void* spinImage
```

Handle for image functionality.

Created by calling [spinCameraGetNextImage\(\)](#) or [spinCameraGetNextImageEx\(\)](#), which require a call to [spinImageRelease\(\)](#) to remove from buffer, or [spinImageCreateEmpty\(\)](#), [spinImageCreateEx\(\)](#), or [spinImageCreate\(\)](#), which require a call to [spinImageDestroy\(\)](#) to destroy.

#### 6.20.2.8 spinImageEvent

```
typedef void* spinImageEvent
```

Handle for image event functionality.

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

#### 6.20.2.9 spinImageStatistics

```
typedef void* spinImageStatistics
```

Handle for image statistics functionality.

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

#### 6.20.2.10 spinInterface

```
typedef void* spinInterface
```

Handle for interface functionality.

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

#### 6.20.2.11 spinInterfaceEvent

```
typedef void* spinInterfaceEvent
```

Handle for interface event functionality.

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

#### 6.20.2.12 spinInterfaceList

```
typedef void* spinInterfaceList
```

Handle for interface list functionality.

Created by calling [spinSystemGetInterfaces\(\)](#), which requires a call to [spinInterfaceListClear\(\)](#) to clear, or [spinInterfaceListCreateEmpty\(\)](#), which requires a call to [spinInterfaceListDestroy\(\)](#) to destroy.

#### 6.20.2.13 spinLogEvent

```
typedef void* spinLogEvent
```

Handle for logging event functionality.

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

#### 6.20.2.14 spinLogEventData

```
typedef void* spinLogEventData
```

Handle for logging event data functionality.

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

#### 6.20.2.15 spinRemovalEvent

```
typedef void* spinRemovalEvent
```

Handle for removal event functionality.

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

#### 6.20.2.16 spinSystem

```
typedef void* spinSystem
```

Handle for system functionality.

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

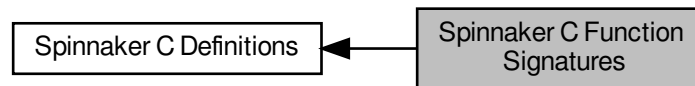
#### 6.20.2.17 spinVideo

```
typedef void* spinVideo
```

## 6.21 Spinnaker C Function Signatures

Spinnaker C function signature definitions.

Collaboration diagram for Spinnaker C Function Signatures:



### Typedefs

- typedef void(\* [spinDeviceEventFunction](#)) (const [spinDeviceEventData](#) hEventData, const char \*pEventName, void \*pUserData)
- Function signatures are used to create and trigger callbacks and events.*
- typedef void(\* [spinImageEventFunction](#)) (const [spinImage](#) hImage, void \*pUserData)
  - typedef void(\* [spinArrivalEventFunction](#)) (uint64\_t deviceSerialNumber, void \*pUserData)
  - typedef void(\* [spinRemovalEventFunction](#)) (uint64\_t deviceSerialNumber, void \*pUserData)
  - typedef void(\* [spinLogEventFunction](#)) (const [spinLogEventData](#) hEventData, void \*pUserData)

### 6.21.1 Detailed Description

Spinnaker C function signature definitions.

### 6.21.2 Typedef Documentation

#### 6.21.2.1 spinArrivalEventFunction

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

#### 6.21.2.2 spinDeviceEventFunction

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

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

### 6.21.2.3 spinImageEventFunction

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

### 6.21.2.4 spinLogEventFunction

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

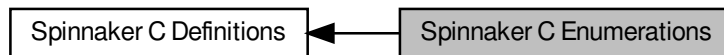
### 6.21.2.5 spinRemovalEventFunction

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

## 6.22 Spinnaker C Enumerations

Spinnaker C enumeration definitions.

Collaboration diagram for Spinnaker C Enumerations:



### Enumerations

- enum `spinError` {
  - `SPINNAKER_ERR_SUCCESS` = 0,
  - `SPINNAKER_ERR_ERROR` = -1001,
  - `SPINNAKER_ERR_NOT_INITIALIZED` = -1002,
  - `SPINNAKER_ERR_NOT_IMPLEMENTED` = -1003,
  - `SPINNAKER_ERR_RESOURCE_IN_USE` = -1004,
  - `SPINNAKER_ERR_ACCESS_DENIED` = -1005,
  - `SPINNAKER_ERR_INVALID_HANDLE` = -1006,
  - `SPINNAKER_ERR_INVALID_ID` = -1007,
  - `SPINNAKER_ERR_NO_DATA` = -1008,
  - `SPINNAKER_ERR_INVALID_PARAMETER` = -1009,
  - `SPINNAKER_ERR_IO` = -1010,
  - `SPINNAKER_ERR_TIMEOUT` = -1011,
  - `SPINNAKER_ERR_ABORT` = -1012,
  - `SPINNAKER_ERR_INVALID_BUFFER` = -1013,
  - `SPINNAKER_ERR_NOT_AVAILABLE` = -1014,
  - `SPINNAKER_ERR_INVALID_ADDRESS` = -1015,
  - `SPINNAKER_ERR_BUFFER_TOO_SMALL` = -1016,
  - `SPINNAKER_ERR_INVALID_INDEX` = -1017,
  - `SPINNAKER_ERR_PARSING_CHUNK_DATA` = -1018,
  - `SPINNAKER_ERR_INVALID_VALUE` = -1019,
  - `SPINNAKER_ERR_RESOURCE_EXHAUSTED` = -1020,
  - `SPINNAKER_ERR_OUT_OF_MEMORY` = -1021,
  - `SPINNAKER_ERR_BUSY` = -1022,
  - `GENICAM_ERR_INVALID_ARGUMENT` = -2001,
  - `GENICAM_ERR_OUT_OF_RANGE` = -2002,
  - `GENICAM_ERR_PROPERTY` = -2003,
  - `GENICAM_ERR_RUN_TIME` = -2004,
  - `GENICAM_ERR_LOGICAL` = -2005,
  - `GENICAM_ERR_ACCESS` = -2006,
  - `GENICAM_ERR_TIMEOUT` = -2007,
  - `GENICAM_ERR_DYNAMIC_CAST` = -2008,
  - `GENICAM_ERR_GENERIC` = -2009,
  - `GENICAM_ERR_BAD_ALLOCATION` = -2010,
  - `SPINNAKER_ERR_IM_CONVERT` = -3001,
  - `SPINNAKER_ERR_IM_COPY` = -3002,
  - `SPINNAKER_ERR_IM_MALLOC` = -3003,
  - `SPINNAKER_ERR_IM_NOT_SUPPORTED` = -3004,

```
SPINNAKER_ERR_IM_HISTOGRAM_RANGE = -3005,
SPINNAKER_ERR_IM_HISTOGRAM_MEAN = -3006,
SPINNAKER_ERR_IM_MIN_MAX = -3007,
SPINNAKER_ERR_IM_COLOR_CONVERSION = -3008,
SPINNAKER_ERR_CUSTOM_ID = -10000 }
```

*The error codes used in Spinnaker C.*

- enum `spinColorProcessingAlgorithm` {  
`DEFAULT`,  
`NO_COLOR_PROCESSING`,  
`NEAREST_NEIGHBOR`,  
`NEAREST_NEIGHBOR_AVG`,  
`BILINEAR`,  
`EDGE_SENSING`,  
`HQ_LINEAR`,  
`IPP`,  
`DIRECTIONAL_FILTER`,  
`RIGOROUS`,  
`WEIGHTED_DIRECTIONAL_FILTER` }

*Color processing algorithms.*

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

*Channels that allow statistics to be calculated.*

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

*File formats to be used for saving images to disk.*

- enum `spinPixelFormatNamespaceID` {  
`SPINNAKER_PIXELFORMAT_NAMESPACE_UNKNOWN` = 0,  
`SPINNAKER_PIXELFORMAT_NAMESPACE_GEV` = 1,  
`SPINNAKER_PIXELFORMAT_NAMESPACE_IIDC` = 2,  
`SPINNAKER_PIXELFORMAT_NAMESPACE_PFNC_16BIT` = 3,  
`SPINNAKER_PIXELFORMAT_NAMESPACE_PFNC_32BIT` = 4,  
`SPINNAKER_PIXELFORMAT_NAMESPACE_CUSTOM_ID` = 1000 }

*This enum represents the namespace in which the TL specific pixel format resides.*

- enum `spinImageStatus` {  
`IMAGE_UNKNOWN_ERROR` = -1,  
`IMAGE_NO_ERROR` = 0,  
`IMAGE_CRC_CHECK_FAILED` = 1,  
`IMAGE_DATA_OVERFLOW` = 2,  
`IMAGE_MISSING_PACKETS` = 3,  
`IMAGE_LEADER_BUFFER_SIZE_INCONSISTENT` = 4,  
`IMAGE_TRAILER_BUFFER_SIZE_INCONSISTENT` = 5,

```

IMAGE_PACKETID_INCONSISTENT = 6,
IMAGE_MISSING_LEADER = 7,
IMAGE_MISSING_TRAILER = 8,
IMAGE_DATA_INCOMPLETE = 9,
IMAGE_INFO_INCONSISTENT = 10,
IMAGE_CHUNK_DATA_INVALID = 11,
IMAGE_NO_SYSTEM_RESOURCES = 12 }

```

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

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

*log levels*

- enum `spinPayloadTypeInfoIds` {  
`PAYLOAD_TYPE_UNKNOWN` = 0,  
`PAYLOAD_TYPE_IMAGE` = 1,  
`PAYLOAD_TYPE_RAW_DATA` = 2,  
`PAYLOAD_TYPE_FILE` = 3,  
`PAYLOAD_TYPE_CHUNK_DATA` = 4,  
`PAYLOAD_TYPE_JPEG` = 5,  
`PAYLOAD_TYPE_JPEG2000` = 6,  
`PAYLOAD_TYPE_H264` = 7,  
`PAYLOAD_TYPE_CHUNK_ONLY` = 8,  
`PAYLOAD_TYPE_DEVICE_SPECIFIC` = 9,  
`PAYLOAD_TYPE_MULTI_PART` = 10,  
`PAYLOAD_TYPE_CUSTOM_ID` = 1000,  
`PAYLOAD_TYPE_EXTENDED_CHUNK` = 1001 }

### 6.22.1 Detailed Description

Spinnaker C enumeration definitions.

### 6.22.2 Enumeration Type Documentation

#### 6.22.2.1 `spinColorProcessingAlgorithm`

```
enum spinColorProcessingAlgorithm
```

Color processing algorithms.

Please refer to our knowledge base at article at <http://www.ptgrey.com/support/kb/index.asp?a=4&q=33> for complete details for each algorithm.



## Enumerator

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

## 6.22.2.2 spinError

```
enum spinError
```

The error codes used in Spinnaker C.

These codes are returned from every function in Spinnaker C. The error codes in the range of -2000 to -2999 are reserved for GenICam related errors. The error codes in the range of -3000 to -3999 are reserved for image processing related errors.

## Enumerator

SPINNAKER_ERR_SUCCESS	An error code of 0 means that the function has run without error.
SPINNAKER_ERR_ERROR	The error codes in the range of -1000 to -1999 are reserved for Spinnaker exceptions.
SPINNAKER_ERR_NOT_INITIALIZED	
SPINNAKER_ERR_NOT_IMPLEMENTED	
SPINNAKER_ERR_RESOURCE_IN_USE	
SPINNAKER_ERR_ACCESS_DENIED	
SPINNAKER_ERR_INVALID_HANDLE	
SPINNAKER_ERR_INVALID_ID	
SPINNAKER_ERR_NO_DATA	
SPINNAKER_ERR_INVALID_PARAMETER	
SPINNAKER_ERR_IO	
SPINNAKER_ERR_TIMEOUT	
SPINNAKER_ERR_ABORT	
SPINNAKER_ERR_INVALID_BUFFER	
SPINNAKER_ERR_NOT_AVAILABLE	
SPINNAKER_ERR_INVALID_ADDRESS	
SPINNAKER_ERR_BUFFER_TOO_SMALL	
SPINNAKER_ERR_INVALID_INDEX	
SPINNAKER_ERR_PARSING_CHUNK_DATA	

## Enumerator

SPINNAKER_ERR_INVALID_VALUE	
SPINNAKER_ERR_RESOURCE_EXHAUSTED	
SPINNAKER_ERR_OUT_OF_MEMORY	
SPINNAKER_ERR_BUSY	
GENICAM_ERR_INVALID_ARGUMENT	The error codes in the range of -2000 to -2999 are reserved for Gen API related errors.
GENICAM_ERR_OUT_OF_RANGE	
GENICAM_ERR_PROPERTY	
GENICAM_ERR_RUN_TIME	
GENICAM_ERR_LOGICAL	
GENICAM_ERR_ACCESS	
GENICAM_ERR_TIMEOUT	
GENICAM_ERR_DYNAMIC_CAST	
GENICAM_ERR_GENERIC	
GENICAM_ERR_BAD_ALLOCATION	
SPINNAKER_ERR_IM_CONVERT	The error codes in the range of -3000 to -3999 are reserved for image processing related errors.
SPINNAKER_ERR_IM_COPY	
SPINNAKER_ERR_IM_MALLOC	
SPINNAKER_ERR_IM_NOT_SUPPORTED	
SPINNAKER_ERR_IM_HISTOGRAM_RANGE	
SPINNAKER_ERR_IM_HISTOGRAM_MEAN	
SPINNAKER_ERR_IM_MIN_MAX	
SPINNAKER_ERR_IM_COLOR_CONVERSION	
SPINNAKER_ERR_CUSTOM_ID	Error codes less than -10000 are reserved for user-defined custom errors.

## 6.22.2.3 spinImageFileFormat

```
enum spinImageFileFormat
```

File formats to be used for saving images to disk.

## Enumerator

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

## 6.22.2.4 spinImageStatus

enum `spinImageStatus`

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

## Enumerator

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

## 6.22.2.5 spinnakerLogLevel

enum `spinnakerLogLevel`

log levels

## Enumerator

LOG_LEVEL_OFF	
LOG_LEVEL_FATAL	
LOG_LEVEL_ALERT	
LOG_LEVEL_CRIT	
LOG_LEVEL_ERROR	
LOG_LEVEL_WARN	
LOG_LEVEL_NOTICE	
LOG_LEVEL_INFO	
LOG_LEVEL_DEBUG	
LOG_LEVEL_NOTSET	

### 6.22.2.6 spinPayloadTypeInfoIDs

enum [spinPayloadTypeInfoIDs](#)

#### Enumerator

PAYLOAD_TYPE_UNKNOWN	
PAYLOAD_TYPE_IMAGE	
PAYLOAD_TYPE_RAW_DATA	
PAYLOAD_TYPE_FILE	
PAYLOAD_TYPE_CHUNK_DATA	
PAYLOAD_TYPE_JPEG	
PAYLOAD_TYPE_JPEG2000	
PAYLOAD_TYPE_H264	
PAYLOAD_TYPE_CHUNK_ONLY	
PAYLOAD_TYPE_DEVICE_SPECIFIC	
PAYLOAD_TYPE_MULTI_PART	
PAYLOAD_TYPE_CUSTOM_ID	
PAYLOAD_TYPE_EXTENDED_CHUNK	

### 6.22.2.7 spinPixelFormatNamespaceID

enum [spinPixelFormatNamespaceID](#)

This enum represents the namespace in which the TL specific pixel format resides.

This enum is returned from a captured image when calling [spinImageGetTLPixelFormatNamespace\(\)](#). It can be used to interpret the raw pixel format returned from [spinImageGetTLPixelFormat\(\)](#).

#### See also

[spinImageGetTLPixelFormat\(\)](#)  
[spinImageGetTLPixelFormatNamespace\(\)](#)

#### Enumerator

SPINNAKER_PIXELFORMAT_NAMESPACE_UNKNOWN	
SPINNAKER_PIXELFORMAT_NAMESPACE_GEV	
SPINNAKER_PIXELFORMAT_NAMESPACE_IIDC	
SPINNAKER_PIXELFORMAT_NAMESPACE_PFNC_16BIT	
SPINNAKER_PIXELFORMAT_NAMESPACE_PFNC_32BIT	
SPINNAKER_PIXELFORMAT_NAMESPACE_CUSTOM_ID	

## 6.22.2.8 spinStatisticsChannel

```
enum spinStatisticsChannel
```

Channels that allow statistics to be calculated.

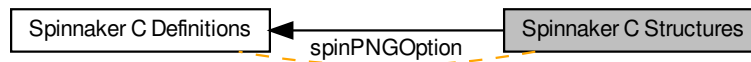
## Enumerator

GREY	
RED	
GREEN	
BLUE	
HUE	
SATURATION	
LIGHTNESS	
NUM_STATISTICS_CHANNELS	

## 6.23 Spinnaker C Structures

Spinnaker C structure definitions.

Collaboration diagram for Spinnaker C Structures:



### Data Structures

- struct [spinPNGOption](#)  
*Options for saving PNG images.*
- struct [spinPPMOption](#)  
*Options for saving PPM images.*
- struct [spinPGMOption](#)  
*Options for saving PGM images.*
- struct [spinTIFFOption](#)  
*Options for saving TIFF images.*
- struct [spinJPEGOption](#)  
*Options for saving JPEG images.*
- struct [spinJPG2Option](#)  
*Options for saving JPEG 2000 images.*
- struct [spinBMPOption](#)  
*Options for saving BMP images.*
- struct [spinMJPGOption](#)  
*Options for saving MJPG videos.*
- struct [spinH264Option](#)  
*Options for saving H264 videos.*
- struct [spinAVIOption](#)  
*Options for saving uncompressed videos.*
- struct [spinLibraryVersion](#)  
*Provides easier access to the current version of Spinnaker.*
- struct [actionCommandResult](#)  
*Action Command Result.*

### Enumerations

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

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

- enum [actionCommandStatus](#) {  
[ACTION\\_COMMAND\\_STATUS\\_OK](#) = 0,  
[ACTION\\_COMMAND\\_STATUS\\_NO\\_REF\\_TIME](#) = 0x8013,  
[ACTION\\_COMMAND\\_STATUS\\_OVERFLOW](#) = 0x8015,  
[ACTION\\_COMMAND\\_STATUS\\_ACTION\\_LATE](#) = 0x8016,  
[ACTION\\_COMMAND\\_STATUS\\_ERROR](#) = 0x8FFF }

Possible Status Codes Returned from Action Command.

### 6.23.1 Detailed Description

Spinnaker C structure definitions.

### 6.23.2 Enumeration Type Documentation

#### 6.23.2.1 [actionCommandStatus](#)

enum [actionCommandStatus](#)

Possible Status Codes Returned from Action Command.

Enumerator

<a href="#">ACTION_COMMAND_STATUS_OK</a>	The device acknowledged the command.
<a href="#">ACTION_COMMAND_STATUS_NO_REF_TIME</a>	
<a href="#">ACTION_COMMAND_STATUS_OVERFLOW</a>	
<a href="#">ACTION_COMMAND_STATUS_ACTION_LATE</a>	
<a href="#">ACTION_COMMAND_STATUS_ERROR</a>	

#### 6.23.2.2 [spinCompressionMethod](#)

enum [spinCompressionMethod](#)

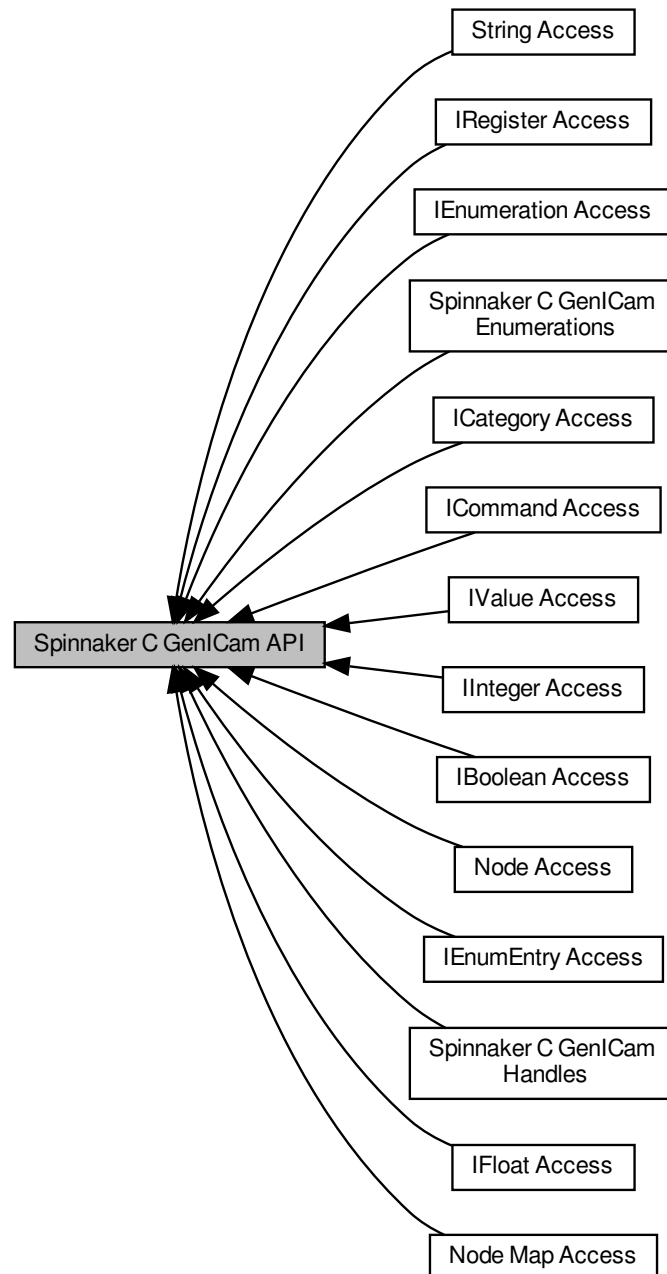
Compression method used in saving TIFF images in the [spinTIFFOption](#) struct.

Enumerator

<a href="#">NONE</a>	
<a href="#">PACKBITS</a>	
<a href="#">DEFLATE</a>	
<a href="#">ADOBE_DEFLATE</a>	
<a href="#">CCITTFAX3</a>	
<a href="#">CCITTFAX4</a>	
<a href="#">LZW</a>	
<a href="#">JPG</a>	

## 6.24 Spinnaker C GenICam API

Collaboration diagram for Spinnaker C GenICam API:



### Modules

- [Node Map Access](#)

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



- [Node Access](#)

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

- [IValue Access](#)

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

- [String Access](#)

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

- [Integer Access](#)

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

- [IFloat Access](#)

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

- [IEnumeration Access](#)

*The functions in this section provide access to enum nodes.*

- [IEnumEntry Access](#)

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

- [IBoolean Access](#)

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

- [ICommand Access](#)

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

- [ICategory Access](#)

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

- [IRegister Access](#)

*The functions in this section provide access to register nodes.*

- [Spinnaker C GenICam Handles](#)

*Handle definitions for Spinnaker C GenICam API.*

- [Spinnaker C GenICam Enumerations](#)

*Enumeration definitions for Spinnaker C GenICam API.*

### 6.24.1 Detailed Description

## 6.25 Node Map Access

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

Collaboration diagram for Node Map Access:



### Functions

- [SPINNAKERC\\_API spinNodeMapGetNode](#) ([spinNodeMapHandle](#) hNodeMap, const char \*pName, [spinNodeHandle](#) \*pNode)  
*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](#) \*pNode)  
*Retrieves a node from the nodemap by index.*
- [SPINNAKERC\\_API spinNodeMapPoll](#) ([spinNodeMapHandle](#) hNodeMap, int64\_t timestamp)  
*Fires nodes which have a polling time.*

### 6.25.1 Detailed Description

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

This includes nodes, node counts, and polling.

### 6.25.2 Function Documentation

#### 6.25.2.1 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>phNode</i>	The node handle pointer in which the node is returned

## Returns

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

6.25.2.2 `spinNodeMapGetNodeByIndex()`

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

Retrieves a node from the nodemap by index.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.25.2.3 `spinNodeMapGetNumNodes()`

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

Gets the number of nodes in the map.

## See also

[spinError](#)

**Parameters**

<i>hNodeMap</i>	The node map where the nodes to be counted are
<i>pValue</i>	The unsigned integer pointer in which the number of nodes is returned

**Returns**

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

**6.25.2.4 spinNodeMapPoll()**

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

Fires nodes which have a polling time.

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

## 6.26 Node Access

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

Collaboration diagram for Node Access:



### Functions

- [SPINNAKERC\\_API spinNodesImplemented](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbResult)  
*Checks whether a node is implemented.*
- [SPINNAKERC\\_API spinNodesReadable](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbResult)  
*Checks whether a node is readable.*
- [SPINNAKERC\\_API spinNodesWritable](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbResult)  
*Checks whether a node is writable.*
- [SPINNAKERC\\_API spinNodesAvailable](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbResult)  
*Checks whether a node is available.*
- [SPINNAKERC\\_API spinNodesEqual](#) ([spinNodeHandle](#) hNodeFirst, [spinNodeHandle](#) hNodeSecond, [bool8\\_t](#) \*pbResult)  
*Checks whether two nodes are equal.*
- [SPINNAKERC\\_API spinNodeGetAccessMode](#) ([spinNodeHandle](#) hNode, [spinAccessMode](#) \*pAccessMode)  
*Retrieves the access mode of a node (as an enum, spinAccessMode)*
- [SPINNAKERC\\_API spinNodeGetName](#) ([spinNodeHandle](#) hNode, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves the name of a node (no whitespace)*
- [SPINNAKERC\\_API spinNodeGetNameSpace](#) ([spinNodeHandle](#) hNode, [spinNameSpace](#) \*pNamespace)  
*Retrieve the namespace of a node (as an enum, spinNameSpace)*
- [SPINNAKERC\\_API spinNodeGetVisibility](#) ([spinNodeHandle](#) hNode, [spinVisibility](#) \*pVisibility)  
*Retrieves the recommended visibility of a node (as an enum, spinVisibility)*
- [SPINNAKERC\\_API spinNodeInvalidateNode](#) ([spinNodeHandle](#) hNode)  
*Invalidates a node in case its values may have changed, rendering it no longer valid.*
- [SPINNAKERC\\_API spinNodeGetCachingMode](#) ([spinNodeHandle](#) hNode, [spinCachingMode](#) \*pCachingMode)  
*Retrieves the caching mode of a node (as an enum, spinCachingMode)*
- [SPINNAKERC\\_API spinNodeGetToolTip](#) ([spinNodeHandle](#) hNode, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves a short description of a node.*
- [SPINNAKERC\\_API spinNodeGetDescription](#) ([spinNodeHandle](#) hNode, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves a longer description of a node.*
- [SPINNAKERC\\_API spinNodeGetDisplayName](#) ([spinNodeHandle](#) hNode, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves the display name of a node (whitespace possible)*
- [SPINNAKERC\\_API spinNodeGetType](#) ([spinNodeHandle](#) hNode, [spinNodeType](#) \*pType)  
*Retrieves the type of a node (as an enum, spinNodeType)*
- [SPINNAKERC\\_API spinNodeGetPollingTime](#) ([spinNodeHandle](#) hNode, [int64\\_t](#) \*pPollingTime)

*Retrieve the polling time of a node.*

- [SPINNAKERC\\_API spinNodeRegisterCallback](#) ([spinNodeHandle](#) hNode, [spinNodeCallbackFunction](#) pCb↔  
Function, [spinNodeCallbackHandle](#) \*phCb)

*Registers a callback to a node.*

- [SPINNAKERC\\_API spinNodeDeregisterCallback](#) ([spinNodeHandle](#) hNode, [spinNodeCallbackHandle](#) hCb)

*Unregisters a callback from a node.*

- [SPINNAKERC\\_API spinNodeGetImposedAccessMode](#) ([spinNodeHandle](#) hNode, [spinAccessMode](#) imposedAccessMode)

*Retrieves the imposed access mode of a node.*

- [SPINNAKERC\\_API spinNodeGetImposedVisibility](#) ([spinNodeHandle](#) hNode, [spinVisibility](#) imposedVisibility)

*Retrieves the imposed visibility of a node.*

## 6.26.1 Detailed Description

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

This includes node properties and callback registration.

## 6.26.2 Function Documentation

### 6.26.2.1 spinNodeDeregisterCallback()

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

Unregisters a callback from a node.

See also

[spinError](#)

#### Parameters

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

#### Returns

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

### 6.26.2.2 spinNodeGetAccessMode()

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

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

See also

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

#### Parameters

<i>hNode</i>	The node of the access mode to retrieve
<i>pAccessMode</i>	The access mode enum pointer in which the access mode is returned

#### Returns

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

### 6.26.2.3 spinNodeGetCachingMode()

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

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

See also

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

#### Parameters

<i>hNode</i>	The node of the caching mode to retrieve
<i>pCachingMode</i>	The caching mode enum pointer in which the caching mode is returned

#### Returns

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

#### 6.26.2.4 spinNodeGetDescription()

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

Retrieves a longer description of a node.

See also

[spinError](#)

##### Parameters

<i>hNode</i>	The node of the description to retrieve
<i>pBuf</i>	The c-string character buffer in which the longer description of the node is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

##### Returns

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

#### 6.26.2.5 spinNodeGetDisplayName()

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

Retrieves the display name of a node (whitespace possible)

See also

[spinError](#)

##### Parameters

<i>hNode</i>	The node of the display name to retrieve
<i>pBuf</i>	The c-string character buffer in which the display name of the node is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

##### Returns

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



#### 6.26.2.6 spinNodeGetImposedAccessMode()

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

Retrieves the imposed access mode of a node.

See also

[spinError](#)

##### Parameters

<i>hNode</i>	The node of the imposed access mode to retrieve
<i>imposedAccessMode</i>	The access mode enum pointer in which the imposed access mode is returned

##### Returns

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

#### 6.26.2.7 spinNodeGetImposedVisibility()

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

Retrieves the imposed visibility of a node.

See also

[spinError](#)

##### Parameters

<i>hNode</i>	The node of the visibility to impose
<i>imposedVisibility</i>	The visibility enum pointer in which the imposed visibility is returned

##### Returns

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

#### 6.26.2.8 spinNodeGetName()

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

Retrieves the name of a node (no whitespace)

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.26.2.9 spinNodeGetNamespace()

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

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

See also

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

##### Parameters

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

##### Returns

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

#### 6.26.2.10 spinNodeGetPollingTime()

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

Retrieve the polling time of a node.

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.26.2.11 spinNodeGetToolTip()

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

Retrieves a short description of a node.

See also

[spinError](#)

##### Parameters

<i>hNode</i>	The node of the tooltip to retrieve
<i>pBuf</i>	The c-string character buffer in which the short description of the node is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

##### Returns

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

#### 6.26.2.12 spinNodeGetType()

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

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

See also

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

##### Parameters

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

##### Returns

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

#### 6.26.2.13 spinNodeGetVisibility()

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

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

See also

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

##### Parameters

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

##### Returns

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

#### 6.26.2.14 spinNodeInvalidateNode()

```
SPINNAKERC_API spinNodeInvalidateNode (
    spinNodeHandle hNode )
```

Invalidates a node in case its values may have changed, rendering it no longer valid.

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.26.2.15 spinNodeIsAvailable()

```
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

#### 6.26.2.16 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

#### 6.26.2.17 spinNodesImplemented()

```
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

#### 6.26.2.18 spinNodesReadable()

```
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

#### 6.26.2.19 `spinNodesWritable()`

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

Checks whether a node is writable.

See also

[spinError](#)

#### Parameters

<i>hNode</i>	The node to check
<i>pbResult</i>	The boolean pointer to return whether or not the node is writable

#### Returns

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

#### 6.26.2.20 `spinNodeRegisterCallback()`

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

Registers a callback to a node.

See also

[spinError](#)

**Parameters**

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

**Returns**

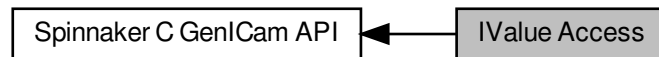
spinError The error code; returns SPINNAKER\_ERR\_SUCCESS (or 0) for no error



## 6.27 IValue Access

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

Collaboration diagram for IValue Access:



### Functions

- [SPINNAKERC\\_API spinNodeToString](#) ([spinNodeHandle](#) hNode, char \*pBuf, size\_t \*pBufLen)  
*Retrieves the value of any node type as a c-string.*
- [SPINNAKERC\\_API spinNodeToStringEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, char \*pBuf, size\_t \*pBufLen)  
*Retrieves the value of any node type as a c-string; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinNodeFromString](#) ([spinNodeHandle](#) hNode, const char \*pBuf)  
*Sets the value of any node type from a c-string; it is important to ensure that the value of the c-string is appropriate to the node type.*
- [SPINNAKERC\\_API spinNodeFromStringEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, const char \*pBuf)  
*Sets the value of any node type from a c-string; manually set whether to verify the node; ensure the value of the c-string is appropriate to the node type.*

### 6.27.1 Detailed Description

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

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

### 6.27.2 Function Documentation

#### 6.27.2.1 spinNodeFromString()

```

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

Sets the value of any node type from a c-string; it is important to ensure that the value of the c-string is appropriate to the node type.

See also

[spinError](#)

## Parameters

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

## Returns

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

6.27.2.2 `spinNodeFromStringEx()`

```
SPINNAKERC_API spinNodeFromStringEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    const char * pBuf )
```

Sets the value of any node type from a c-string; manually set whether to verify the node; ensure the value of the c-string is appropriate to the node type.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.27.2.3 `spinNodeToString()`

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

Retrieves the value of any node type as a c-string.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.27.2.4 `spinNodeToStringEx()`

```
SPINNAKERC_API spinNodeToStringEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the value of any node type as a c-string; manually set whether to verify the node.

## See also

[`spinError`](#)

## Parameters

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

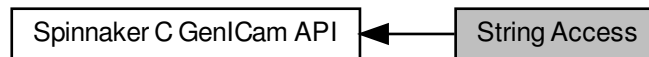
## Returns

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

## 6.28 String Access

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

Collaboration diagram for String Access:



### Functions

- [SPINNAKERC\\_API spinStringSetValue](#) ([spinNodeHandle](#) hNode, const char \*pBuf)  
*Sets the value of a string node.*
- [SPINNAKERC\\_API spinStringSetValueEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, const char \*pBuf)  
*Sets the value of a string node; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinStringGetValue](#) ([spinNodeHandle](#) hNode, char \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves the value of a string node as a c-string.*
- [SPINNAKERC\\_API spinStringGetValueEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, char \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves the value of a string node as a cstring; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinStringGetMaxLength](#) ([spinNodeHandle](#) hNode, [int64\\_t](#) \*pValue)  
*Retrieves the maximum length of the c-string to be returned.*

### 6.28.1 Detailed Description

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

This includes getters and setters of values and value lengths.

### 6.28.2 Function Documentation

#### 6.28.2.1 spinStringGetMaxLength()

```

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

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

See also

[spinError](#)

## Parameters

<i>hNode</i>	The string node of the length to retrieve
<i>pValue</i>	The integer pointer in which the maximum length of the c-string is returned

## Returns

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

6.28.2.2 `spinStringGetValue()`

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

Retrieves the value of a string node as a c-string.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.28.2.3 `spinStringGetValueEx()`

```
SPINNAKERC_API spinStringGetValueEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    char * pBuf,
    size_t * pBufLen )
```

Retrieves the value of a string node as a cstring; manually set whether to verify the node.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.28.2.4 `spinStringSetValue()`

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

Sets the value of a string node.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.28.2.5 `spinStringSetValueEx()`

```
SPINNAKERC_API spinStringSetValueEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    const char * pBuf )
```

Sets the value of a string node; manually set whether to verify the node.

## See also

[spinError](#)

## Parameters

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

## Returns

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

## 6.29 Integer Access

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

Collaboration diagram for Integer Access:



### Functions

- [SPINNAKERC\\_API spinIntegerSetValue](#) ([spinNodeHandle](#) hNode, `int64_t` value)  
*Sets the value of an integer node.*
- [SPINNAKERC\\_API spinIntegerSetValueEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, `int64_t` value)  
*Sets the value of an integer node; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinIntegerGetValue](#) ([spinNodeHandle](#) hNode, `int64_t` \*pValue)  
*Retrieves the value of an integer node.*
- [SPINNAKERC\\_API spinIntegerGetValueEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, `int64_t` \*pValue)  
*Retrieves the value of an integer node; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinIntegerGetMin](#) ([spinNodeHandle](#) hNode, `int64_t` \*pValue)  
*Retrieves the minimum value of an integer node; all potential values must be greater than or equal to the minimum.*
- [SPINNAKERC\\_API spinIntegerGetMax](#) ([spinNodeHandle](#) hNode, `int64_t` \*pValue)  
*Retrieves the maximum value of an integer node; all potential values must be lesser than or equal to the maximum.*
- [SPINNAKERC\\_API spinIntegerGetInc](#) ([spinNodeHandle](#) hNode, `int64_t` \*pValue)  
*Retrieves the increment of an integer node; all possible values must be divisible by the increment.*
- [SPINNAKERC\\_API spinIntegerGetRepresentation](#) ([spinNodeHandle](#) hNode, [spinRepresentation](#) \*pValue)  
*Retrieves the numerical representation of the value of a node; i.e.*

### 6.29.1 Detailed Description

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

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

### 6.29.2 Function Documentation

#### 6.29.2.1 spinIntegerGetInc()

```

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

Retrieves the increment of an integer node; all possible values must be divisible by the increment.

See also

[spinError](#)



## Parameters

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

## Returns

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

6.29.2.2 `spinIntegerGetMax()`

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

Retrieves the maximum value of an integer node; all potential values must be lesser than or equal to the maximum.

## See also

[`spinError`](#)

## Parameters

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

## Returns

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

6.29.2.3 `spinIntegerGetMin()`

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

Retrieves the minimum value of an integer node; all potential values must be greater than or equal to the minimum.

## See also

[`spinError`](#)

## Parameters

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

## Returns

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

6.29.2.4 `spinIntegerGetRepresentation()`

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

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

linear, logarithmic, hexadecimal, MAC address, etc.

## See also

[spinError](#)

## Parameters

<i>hNode</i>	The integer node of the numerical representation to retrieve
<i>pValue</i>	The representation enum pointer in which the type of numerical representation is returned

## Returns

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

6.29.2.5 `spinIntegerGetValue()`

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

Retrieves the value of an integer node.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.29.2.6 `spinIntegerGetValueEx()`

```
SPINNAKERC_API spinIntegerGetValueEx (  
    spinNodeHandle hNode,  
    bool8_t bVerify,  
    int64_t * pValue )
```

Retrieves the value of an integer node; manually set whether to verify the node.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.29.2.7 `spinIntegerSetValue()`

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

Sets the value of an integer node.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.29.2.8 `spinIntegerSetValueEx()`

```
SPINNAKERC_API spinIntegerSetValueEx (  
    spinNodeHandle hNode,  
    bool8_t bVerify,  
    int64_t value )
```

Sets the value of an integer node; manually set whether to verify the node.

## See also

[spinError](#)

## Parameters

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

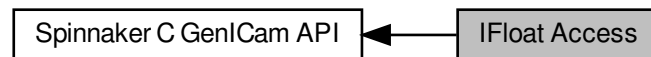
## Returns

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

## 6.30 IFloat Access

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

Collaboration diagram for IFloat Access:



### Functions

- [SPINNAKERC\\_API spinFloatSetValue](#) ([spinNodeHandle](#) hNode, double value)  
*Sets the value of a float node.*
- [SPINNAKERC\\_API spinFloatSetValueEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, double value)  
*Sets the value of a float node; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinFloatGetValue](#) ([spinNodeHandle](#) hNode, double \*pValue)  
*Retrieves the value of a float node.*
- [SPINNAKERC\\_API spinFloatGetValueEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, double \*pValue)  
*Retrieves the value of a float node; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinFloatGetMin](#) ([spinNodeHandle](#) hNode, double \*pValue)  
*Retrieves the minimum value of a float node; all potential values must be greater than or equal to the minimum.*
- [SPINNAKERC\\_API spinFloatGetMax](#) ([spinNodeHandle](#) hNode, double \*pValue)  
*Retrieves the maximum value of a float node; all potential values must be lesser than or equal to the maximum.*
- [SPINNAKERC\\_API spinFloatGetRepresentation](#) ([spinNodeHandle](#) hNode, [spinRepresentation](#) \*pValue)  
*Retrieves the numerical representation of the value of a node; i.e.*
- [SPINNAKERC\\_API spinFloatGetUnit](#) ([spinNodeHandle](#) hNode, char \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves the units of the float node value.*

### 6.30.1 Detailed Description

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

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

### 6.30.2 Function Documentation

#### 6.30.2.1 spinFloatGetMax()

```

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

Retrieves the maximum value of a float node; all potential values must be lesser than or equal to the maximum.

See also

[spinError](#)

**Parameters**

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

**Returns**

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

**6.30.2.2 spinFloatGetMin()**

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

Retrieves the minimum value of a float node; all potential values must be greater than or equal to the minimum.

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

**6.30.2.3 spinFloatGetRepresentation()**

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

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

linear, logarithmic, hexadecimal, MAC address, etc.

**See also**

[spinError](#)

## Parameters

<i>hNode</i>	The float node of the numerical representation to retrieve
<i>pValue</i>	The representation enum pointer in which the type of numerical representation is returned

## Returns

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

6.30.2.4 `spinFloatGetUnit()`

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

Retrieves the units of the float node value.

## See also

[spinError](#)

## Parameters

<i>hNode</i>	The float node of the units to retrieve
<i>pBuf</i>	The c-string character buffer in which the value units are returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

## Returns

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

6.30.2.5 `spinFloatGetValue()`

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

Retrieves the value of a float node.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.30.2.6 `spinFloatGetValueEx()`

```
SPINNAKERC_API spinFloatGetValueEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    double * pValue )
```

Retrieves the value of a float node; manually set whether to verify the node.

## See also

[spinError](#)

## Parameters

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

## Returns

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

6.30.2.7 `spinFloatSetValue()`

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

Sets the value of a float node.

## See also

[spinError](#)



## Parameters

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

## Returns

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

6.30.2.8 `spinFloatSetValueEx()`

```
SPINNAKERC_API spinFloatSetValueEx (  
    spinNodeHandle hNode,  
    bool8_t bVerify,  
    double value )
```

Sets the value of a float node; manually set whether to verify the node.

## See also

[spinError](#)

## Parameters

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

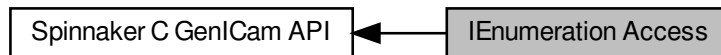
## Returns

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

## 6.31 IEnumeration Access

The functions in this section provide access to enum nodes.

Collaboration diagram for IEnumeration Access:



### Functions

- [SPINNAKERC\\_API spinEnumerationGetNumEntries](#) ([spinNodeHandle](#) hNode, [size\\_t](#) \*pValue)  
*Retrieves the number of entries of an enum node.*
- [SPINNAKERC\\_API spinEnumerationGetEntryByIndex](#) ([spinNodeHandle](#) hNode, [size\\_t](#) index, [spinNodeHandle](#) \*phEntry)  
*Retrieves an entry node from an enum node using an index.*
- [SPINNAKERC\\_API spinEnumerationGetEntryByName](#) ([spinNodeHandle](#) hNode, [const char](#) \*pName, [spinNodeHandle](#) \*phEntry)  
*Retrieves an entry node from an enum node using the entry's symbolic.*
- [SPINNAKERC\\_API spinEnumerationGetCurrentEntry](#) ([spinNodeHandle](#) hNode, [spinNodeHandle](#) \*phEntry)  
*Retrieves the currently selected entry node from an enum node.*
- [SPINNAKERC\\_API spinEnumerationSetIntValue](#) ([spinNodeHandle](#) hNode, [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](#) hNode, [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](#).*

### 6.31.1 Detailed Description

The functions in this section provide access to enum nodes.

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

### 6.31.2 Function Documentation

#### 6.31.2.1 spinEnumerationGetCurrentEntry()

```

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

Retrieves the currently selected entry node from an enum node.

See also

[spinError](#)

## Parameters

<i>hNode</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

6.31.2.2 `spinEnumerationGetEntryByIndex()`

```
SPINNAKERC_API spinEnumerationGetEntryByIndex (
    spinNodeHandle hNode,
    size_t index,
    spinNodeHandle * phEntry )
```

Retrieves an entry node from an enum node using an index.

## See also

[spinError](#)

## Parameters

<i>hNode</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

6.31.2.3 `spinEnumerationGetEntryByName()`

```
SPINNAKERC_API spinEnumerationGetEntryByName (
    spinNodeHandle hNode,
    const char * pName,
    spinNodeHandle * phEntry )
```

Retrieves an entry node from an enum node using the entry's symbolic.

## See also

[spinError](#)

## Parameters

<i>hNode</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

**6.31.2.4 spinEnumerationGetNumEntries()**

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

Retrieves the number of entries of an enum node.

## See also

[spinError](#)

## Parameters

<i>hNode</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

**6.31.2.5 spinEnumerationSetEnumValue()**

```
SPINNAKERC_API spinEnumerationSetEnumValue (
    spinNodeHandle hNode,
    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>hNode</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

6.31.2.6 `spinEnumerationSetIntValue()`

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

Sets a new entry using its integer value retrieved from a call to [spinEnumerationEntryGetIntValue\(\)](#); note that enumeration entry int and enum values are different - int values defined on camera, enum values found in [Spinnaker↔ DefsC.h](#).

## See also

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

## Parameters

<i>hNode</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

## 6.32 IEnumEntry Access

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

Collaboration diagram for IEnumEntry Access:



### Functions

- [SPINNAKERC\\_API spinEnumerationEntryGetIntValue](#) ([spinNodeHandle](#) hNode, [int64\\_t](#) \*pValue)  
*Retrieves the integer value of an entry node; note that enumeration entry int and enum values are different - int values defined on camera, enum values found in [SpinnakerDefsC.h](#).*
- [SPINNAKERC\\_API spinEnumerationEntryGetEnumValue](#) ([spinNodeHandle](#) hNode, [size\\_t](#) \*pValue)  
*Retrieves the enum value (as an integer) of an entry node; note that enumeraiton entry int and enum values are different - int values defined on camera, enum values found in [SpinnakerDefsC.h](#).*
- [SPINNAKERC\\_API spinEnumerationEntryGetSymbolic](#) ([spinNodeHandle](#) hNode, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves the symbolic of an entry node as a c-string.*

### 6.32.1 Detailed Description

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

### 6.32.2 Function Documentation

#### 6.32.2.1 spinEnumerationEntryGetEnumValue()

```
SPINNAKERC_API spinEnumerationEntryGetEnumValue (
    spinNodeHandle hNode,
    size_t * pValue )
```

Retrieves the enum value (as an integer) of an entry node; note that enumeraiton entry int and enum values are different - int values defined on camera, enum values found in [SpinnakerDefsC.h](#).

See also

[spinEnumerationSetEnumValue\(\)](#)  
[spinError](#)

## Parameters

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

## Returns

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

6.32.2.2 `spinEnumerationEntryGetIntValue()`

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

Retrieves the integer value of an entry node; note that enumeration entry int and enum values are different - int values defined on camera, enum values found in [SpinnakerDefsC.h](#).

## See also

[spinEnumerationSetIntValue\(\)](#)  
[spinError](#)

## Parameters

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

## Returns

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

6.32.2.3 `spinEnumerationEntryGetSymbolic()`

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

Retrieves the symbolic of an entry node as a c-string.

## See also

[spinError](#)

**Parameters**

<i>hNode</i>	The entry node of the symbolic to retrieve
<i>pBuf</i>	The c-string character buffer in which the symbolic of the entry node is returned
<i>pBufLen</i>	The unsigned integer pointer in which the length of the c-string is returned; the input value is the maximum length

**Returns**

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



## 6.33 IBoolean Access

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

Collaboration diagram for IBoolean Access:



### Functions

- [SPINNAKERC\\_API spinBooleanSetValue](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) value)  
*Sets the value of a boolean node; boolean values are represented by 'True' (which equals '0') and 'False' (which equals '1')*
- [SPINNAKERC\\_API spinBooleanGetValue](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbValue)  
*Retrieves the value of a boolean node; boolean values are represented by 'True' (which equals '0') and 'False' (which equals '1')*

#### 6.33.1 Detailed Description

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

This includes value getters and setters.

#### 6.33.2 Function Documentation

##### 6.33.2.1 spinBooleanGetValue()

```

SPINNAKERC_API spinBooleanGetValue (
    spinNodeHandle hNode,
    bool8_t * pbValue )
  
```

Retrieves the value of a boolean node; boolean values are represented by 'True' (which equals '0') and 'False' (which equals '1')

See also

[spinError](#)

**Parameters**

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

**Returns**

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

**6.33.2.2 spinBooleanSetValue()**

```
SPINNAKERC_API spinBooleanSetValue (
    spinNodeHandle hNode,
    bool8_t value )
```

Sets the value of a boolean node; boolean values are represented by 'True' (which equals '0') and 'False' (which equals '1')

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

## 6.34 ICommand Access

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

Collaboration diagram for ICommand Access:



### Functions

- [SPINNAKERC\\_API spinCommandExecute](#) ([spinNodeHandle](#) hNode)  
*Executes the action associated to a command node.*
- [SPINNAKERC\\_API spinCommandIsDone](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbValue)  
*Retrieves whether or not the action of a command node has completed.*

#### 6.34.1 Detailed Description

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

This includes node properties and callbacks.

#### 6.34.2 Function Documentation

##### 6.34.2.1 spinCommandExecute()

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

Executes the action associated to a command node.

See also

[spinError](#)

##### Parameters

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

**Returns**

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

**6.34.2.2 spinCommandIsDone()**

```
SPINNAKERC_API spinCommandIsDone (
    spinNodeHandle hNode,
    bool8_t * pbValue )
```

Retrieves whether or not the action of a command node has completed.

**See also**

[`spinError`](#)

**Parameters**

<i>hNode</i>	The command node to check
<i>pValue</i>	The boolean pointer to return whether or not the command has completed

**Returns**

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

## 6.35 ICategory Access

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

Collaboration diagram for ICategory Access:



### Functions

- [SPINNAKERC\\_API spinCategoryGetNumFeatures](#) ([spinNodeHandle](#) hNode, [size\\_t](#) \*pValue)  
*Retrieves the number of a features (or child nodes) or a category node.*
- [SPINNAKERC\\_API spinCategoryGetFeatureByIndex](#) ([spinNodeHandle](#) hNode, [size\\_t](#) index, [spinNodeHandle](#) \*phFeature)  
*Retrieves a node from a category node using an index.*

### 6.35.1 Detailed Description

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

This includes node properties and callbacks.

### 6.35.2 Function Documentation

#### 6.35.2.1 spinCategoryGetFeatureByIndex()

```

SPINNAKERC_API spinCategoryGetFeatureByIndex (
    spinNodeHandle hNode,
    size_t index,
    spinNodeHandle * phFeature )
  
```

Retrieves a node from a category node using an index.

See also

[spinError](#)

**Parameters**

<i>hNode</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

**6.35.2.2 spinCategoryGetNumFeatures()**

```
SPINNAKERC_API spinCategoryGetNumFeatures (
    spinNodeHandle hNode,
    size_t * pValue )
```

Retrieves the number of a features (or child nodes) or a category node.

**See also**

[spinError](#)

**Parameters**

<i>hNode</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

## 6.36 IRegister Access

The functions in this section provide access to register nodes.

Collaboration diagram for IRegister Access:



### Functions

- [SPINNAKERC\\_API spinRegisterGet](#) ([spinNodeHandle](#) hNode, [uint8\\_t](#) \*pBuf, [int64\\_t](#) length)  
*Retrieves the value of a register node.*
- [SPINNAKERC\\_API spinRegisterGetEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, [bool8\\_t](#) bIgnoreCache, [uint8\\_t](#) \*pBuf, [int64\\_t](#) length)  
*Retrieves the value of a register node; manually set whether to verify the node and whether to ignore the cache.*
- [SPINNAKERC\\_API spinRegisterGetAddress](#) ([spinNodeHandle](#) hNode, [int64\\_t](#) \*pAddress)  
*Retrieves the address of a register node.*
- [SPINNAKERC\\_API spinRegisterGetLength](#) ([spinNodeHandle](#) hNode, [int64\\_t](#) \*pLength)  
*Retrieves the length (in bytes) of the value of a register node.*
- [SPINNAKERC\\_API spinRegisterSet](#) ([spinNodeHandle](#) hNode, [const uint8\\_t](#) \*pBuf, [int64\\_t](#) length)  
*Sets the value of a register node.*
- [SPINNAKERC\\_API spinRegisterSetEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, [const uint8\\_t](#) \*pBuf, [int64\\_t](#) length)  
*Sets the value of a register node; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinRegisterSetReference](#) ([spinNodeHandle](#) hNode, [spinNodeHandle](#) hRef)  
*Uses a second node as a reference for a register node.*

### 6.36.1 Detailed Description

The functions in this section provide access to register nodes.

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

### 6.36.2 Function Documentation

#### 6.36.2.1 spinRegisterGet()

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

Retrieves the value of a register node.

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.36.2.2 spinRegisterGetAddress()

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

Retrieves the address of a register node.

See also

[spinError](#)

##### Parameters

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

##### Returns

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



## 6.36.2.3 spinRegisterGetEx()

```
SPINNAKERC_API spinRegisterGetEx (
    spinNodeHandle hNode,
    bool8_t bVerify,
    bool8_t bIgnoreCache,
    uint8_t * pBuf,
    int64_t length )
```

Retrieves the value of a register node; manually set whether to verify the node and whether to ignore the cache.

See also

[spinError](#)

## Parameters

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

## Returns

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

## 6.36.2.4 spinRegisterGetLength()

```
SPINNAKERC_API spinRegisterGetLength (
    spinNodeHandle hNode,
    int64_t * pLength )
```

Retrieves the length (in bytes) of the value of a register node.

See also

[spinError](#)

## Parameters

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

**Returns**

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

**6.36.2.5 spinRegisterSet()**

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

Sets the value of a register node.

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

**6.36.2.6 spinRegisterSetEx()**

```
SPINNAKERC_API spinRegisterSetEx (  
    spinNodeHandle hNode,  
    bool8_t bVerify,  
    const uint8_t * pBuf,  
    int64_t length )
```

Sets the value of a register node; manually set whether to verify the node.

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

**6.36.2.7 spinRegisterSetReference()**

```
SPINNAKERC_API spinRegisterSetReference (
    spinNodeHandle hNode,
    spinNodeHandle hRef )
```

Uses a second node as a reference for a register node.

**See also**

[spinError](#)

**Parameters**

<i>hNode</i>	The register node that houses the reference
<i>hRef</i>	The reference node

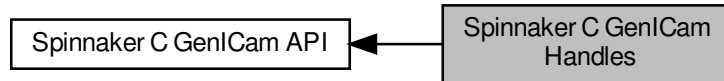
**Returns**

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

## 6.37 Spinnaker C GenICam Handles

Handle definitions for Spinnaker C GenICam API.

Collaboration diagram for Spinnaker C GenICam Handles:



### Typedefs

- typedef void \* [spinNodeMapHandle](#)  
*Handle for nodemap functionality.*
- typedef void \* [spinNodeHandle](#)  
*Handle for node functionality.*
- typedef void \* [spinNodeCallbackHandle](#)  
*Handle for callback functionality.*
- typedef void(\* [spinNodeCallbackFunction](#)) ([spinNodeHandle](#) hNode)  
*Function signatures are used to create and trigger callbacks and events.*

### 6.37.1 Detailed Description

Handle definitions for Spinnaker C GenICam API.

### 6.37.2 Typedef Documentation

#### 6.37.2.1 spinNodeCallbackFunction

```
typedef void(* spinNodeCallbackFunction) (spinNodeHandle hNode)
```

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

#### 6.37.2.2 spinNodeCallbackHandle

```
typedef void* spinNodeCallbackHandle
```

Handle for callback functionality.

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

### 6.37.2.3 spinNodeHandle

```
typedef void* spinNodeHandle
```

Handle for node functionality.

Created by calling [spinNodeMapGetNode\(\)](#). No need to release, clear, or destroy.

### 6.37.2.4 spinNodeMapHandle

```
typedef void* spinNodeMapHandle
```

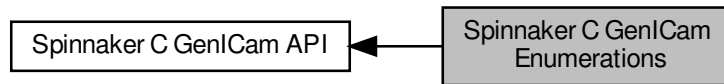
Handle for nodemap functionality.

Created by calling [spinCameraGetNodemap\(\)](#), [spinCameraGetTLDeviceNodeMap\(\)](#), [spinCameraGetTLStreamNodeMap\(\)](#) or [spinInterfaceGetTLNodeMap\(\)](#). No need to release, clear, or destroy.

## 6.38 Spinnaker C GenICam Enumerations

Enumeration definitions for Spinnaker C GenICam API.

Collaboration diagram for Spinnaker C GenICam Enumerations:



### Enumerations

- enum `spinNodeType` {  
`ValueNode`,  
`BaseNode`,  
`IntegerNode`,  
`BooleanNode`,  
`FloatNode`,  
`CommandNode`,  
`StringNode`,  
`RegisterNode`,  
`EnumerationNode`,  
`EnumEntryNode`,  
`CategoryNode`,  
`PortNode`,  
`UnknownNode` = -1 }
- enum `spinSign` {  
`Signed`,  
`Unsigned`,  
`_UndefinedSign` }
- enum `spinAccessMode` {  
`NI`,  
`NA`,  
`WO`,  
`RO`,  
`RW`,  
`_UndefinedAccesMode`,  
`_CycleDetectAccesMode` }
- enum `spinVisibility` {  
`Beginner` = 0,  
`Expert` = 1,  
`Guru` = 2,  
`Invisible` = 3,  
`_UndefinedVisibility` = 99 }
- enum `spinCachingMode` {  
`NoCache`,  
`WriteThrough`,  
`WriteAround`,  
`_UndefinedCachingMode` }

- enum `spinRepresentation` {  
`Linear`,  
`Logarithmic`,  
`Boolean`,  
`PureNumber`,  
`HexNumber`,  
`IPV4Address`,  
`MACAddress`,  
`_UndefinedRepresentation` }  
*recommended representation of a node value*
- enum `spinEndianness` {  
`BigEndian`,  
`LittleEndian`,  
`_UndefinedEndian` }  
*Endianness of a value in a register.*
- enum `spinNameSpace` {  
`Custom`,  
`Standard`,  
`_UndefinedNameSpace` }  
*Defines if a node name is standard or custom.*
- enum `spinStandardNameSpace` {  
`None`,  
`GEV`,  
`IIDC`,  
`CL`,  
`USB`,  
`_UndefinedStandardNameSpace` }  
*Defines from which standard namespace a node name comes from.*
- enum `spinYesNo` {  
`Yes` = 1,  
`No` = 0,  
`_UndefinedYesNo` = 2 }  
*Defines the chices of a Yes/No alternaitve.*
- enum `spinSlope` {  
`Increasing`,  
`Decreasing`,  
`Varying`,  
`Automatic`,  
`_UndefinedESlope` }  
*typedef for fomula type*
- enum `spinXMLValidation` {  
`xvLoad` = 0x00000001L,  
`xvCycles` = 0x00000002L,  
`xvSFNC` = 0x00000004L,  
`xvDefault` = 0x00000000L,  
`xvAll` = 0xffffffffL,  
`_UndefinedEXMLValidation` = 0x80000000L }  
*typedef describing the different validity checks which can be performed on an XML file*
- enum `spinDisplayNotation` {  
`fnAutomatic`,  
`fnFixed`,  
`fnScientific`,  
`_UndefinedEDisplayNotation` }  
*typedef for float notation*
- enum `spinInterfaceType` {  
`intfIValue`,

```

intfIBase,
intfInteger,
intfBoolean,
intfCommand,
intfFloat,
intfString,
intfRegister,
intfCategory,
intfEnumeration,
intfEnumEntry,
intfIPort }

```

*typedef for interface type*

- enum `spinLinkType` {  
`ctAllDependingNodes`,  
`ctAllTerminalNodes`,  
`ctInvalidators`,  
`ctReadingChildren`,  
`ctWritingChildren`,  
`ctDependingChildren` }

*typedef for link type*

- enum `spinIncMode` {  
`noIncrement`,  
`fixedIncrement`,  
`listIncrement` }

*typedef for increment mode*

- enum `spinInputDirection` {  
`idFrom`,  
`idTo`,  
`idNone` }

*typedef for link type*

### 6.38.1 Detailed Description

Enumeration definitions for Spinnaker C GenICam API.

### 6.38.2 Enumeration Type Documentation

#### 6.38.2.1 `spinAccessMode`

```
enum spinAccessMode
```

Enumerator

NI	
NA	
WO	
RO	
RW	
_UndefinedAccesMode	
_CycleDetectAccesMode	



## 6.38.2.2 spinCachingMode

```
enum spinCachingMode
```

## Enumerator

NoCache	
WriteThrough	
WriteAround	
_UndefinedCachingMode	

## 6.38.2.3 spinDisplayNotation

```
enum spinDisplayNotation
```

typedef for float notation

## Enumerator

fnAutomatic	
fnFixed	the notation is either scientific or fixed depending on what is shorter
fnScientific	the notation is fixed, e.g. 123.4
_UndefinedEDisplayNotation	the notation is scientific, e.g. 1.234e2 Object is not yet initialized

## 6.38.2.4 spinEndianess

```
enum spinEndianess
```

Endianess of a value in a register.

## Enumerator

BigEndian	Register is big endian.
LittleEndian	Register is little endian.
_UndefinedEndian	Object is not yet initialized.

### 6.38.2.5 spinIncMode

enum `spinIncMode`

typedef for increment mode

#### Enumerator

noIncrement	
fixedIncrement	
listIncrement	

### 6.38.2.6 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

### 6.38.2.7 spinInterfaceType

enum `spinInterfaceType`

typedef for interface type

#### Enumerator

intfIValue	
intfIBase	IValue interface

## Enumerator

intfInteger	IBase interface
intfBoolean	IInteger interface
intfCommand	IBoolean interface
intfFloat	ICommand interface
intfString	IFloat interface
intfRegister	IString interface
intfCategory	IRegister interface
intfEnumeration	ICategory interface
intfEnumEntry	IEnumeration interface
intfIPort	IEnumEntry interface IPort interface

## 6.38.2.8 spinLinkType

```
enum spinLinkType
```

```
typedef for link type
```

## Enumerator

ctAllDependingNodes	
ctAllTerminalNodes	All nodes which will be invalidated if this node becomes invalid
ctInvalidators	All terminal nodes which may be written to by this node

**Enumerator**

ctReadingChildren	List of references to nodes which may invalidate this node
ctWritingChildren	All child nodes which influence this node's AccessMode
ctDependingChildren	All child nodes which may be written to All child nodes which will cause this node to be invalidated

**6.38.2.9 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.

**6.38.2.10 spinNodeType**

enum `spinNodeType`

**Enumerator**

ValueNode	
BaseNode	
IntegerNode	
BooleanNode	
FloatNode	
CommandNode	
StringNode	
RegisterNode	
EnumerationNode	
EnumEntryNode	
CategoryNode	
PortNode	
UnknownNode	

### 6.38.2.11 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	

### 6.38.2.12 spinSign

enum `spinSign`

#### Enumerator

Signed	
Unsigned	
_UndefinedSign	

### 6.38.2.13 spinSlope

enum `spinSlope`

typedef for fomula type

#### Enumerator

Increasing	
Decreasing	strictly monotonous increasing
Varying	strictly monotonous decreasing

**Enumerator**

Automatic	slope changes, e.g. at run-time
_UndefinedESlope	slope is determined automatically by probing the function Object is not yet initialized

**6.38.2.14 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.

**6.38.2.15 spinVisibility**

enum `spinVisibility`

**Enumerator**

Beginner	
Expert	
Guru	
Invisible	
_UndefinedVisibility	

**6.38.2.16 spinXMLValidation**

enum `spinXMLValidation`

typedef describing the different validity checks which can be performed on an XML file

The enum values for a bitfield of lenght uint32\_t

## Enumerator

xvLoad	
xvCycles	Creates a dummy node map
xvSFNC	checks for write and dependency cycles (implies xvLoad)
xvDefault	checks for conformance with the standard feature naming convention (SFNC)
xvAll	checks performed if nothing else is said
_UndefinedEXMLValidation	all possible checks Object is not yet initialized

## 6.38.2.17 spinYesNo

enum `spinYesNo`

Defines the chices of a Yes/No alternative.

## Enumerator

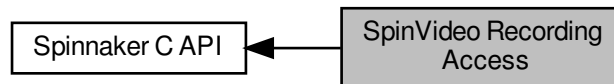
Yes	yes
No	no
_UndefinedYesNo	Object is not yet initialized.



## 6.39 SpinVideo Recording Access

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

Collaboration diagram for SpinVideo Recording Access:



### Functions

- [SPINNAKERC\\_API spinVideoOpenUncompressed](#) ([spinVideo](#) \*phSpinVideo, const char \*pName, [spinAVIOOption](#) option)
- [SPINNAKERC\\_API spinVideoOpenMJPEG](#) ([spinVideo](#) \*phSpinVideo, const char \*pName, [spinMJPEGOption](#) option)
- [SPINNAKERC\\_API spinVideoOpenH264](#) ([spinVideo](#) \*phSpinVideo, const char \*pName, [spinH264Option](#) option)
- [SPINNAKERC\\_API spinVideoAppend](#) ([spinVideo](#) hSpinVideo, [spinImage](#) hImage)
- [SPINNAKERC\\_API spinVideoSetMaximumFileSize](#) ([spinVideo](#) hSpinVideo, unsigned int size)  
*Set the maximum file size (in megabytes) of a AVI/MP4 file.*
- [SPINNAKERC\\_API spinVideoClose](#) ([spinVideo](#) hSpinVideo)

### 6.39.1 Detailed Description

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

### 6.39.2 Function Documentation

#### 6.39.2.1 spinVideoAppend()

```

SPINNAKERC_API spinVideoAppend (
    spinVideo hSpinVideo,
    spinImage hImage )
  
```

### 6.39.2.2 spinVideoClose()

```
SPINNAKERC_API spinVideoClose (
    spinVideo hSpinVideo )
```

### 6.39.2.3 spinVideoOpenH264()

```
SPINNAKERC_API spinVideoOpenH264 (
    spinVideo * phSpinVideo,
    const char * pName,
    spinH264Option option )
```

### 6.39.2.4 spinVideoOpenMJPG()

```
SPINNAKERC_API spinVideoOpenMJPG (
    spinVideo * phSpinVideo,
    const char * pName,
    spinMJPGOption option )
```

### 6.39.2.5 spinVideoOpenUncompressed()

```
SPINNAKERC_API spinVideoOpenUncompressed (
    spinVideo * phSpinVideo,
    const char * pName,
    spinAVIOption option )
```

### 6.39.2.6 spinVideoSetMaximumFileSize()

```
SPINNAKERC_API spinVideoSetMaximumFileSize (
    spinVideo hSpinVideo,
    unsigned int size )
```

Set the maximum file size (in megabytes) of a AVI/MP4 file.

A new AVI/MP4 file is created automatically when file size limit is reached. Setting a maximum size of 0 indicates no limit on file size.

#### Parameters

<i>hSpinVideo</i>	The spin video recorder to append the image to
<i>size</i>	The maximum video file size in MB.

**Returns**

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

## 6.40 Transport Layer Enumerations

Collaboration diagram for Transport Layer Enumerations:



### Enumerations

- enum `spinTLStreamTypeEnums` {  
`StreamType_Mixed`,  
`StreamType_Custom`,  
`StreamType_GEV`,  
`StreamType_CL`,  
`StreamType_IIDC`,  
`StreamType_UVC`,  
`StreamType_CXP`,  
`StreamType_CLHS`,  
`StreamType_U3V`,  
`StreamType_ETHERNET`,  
`StreamType_PCI`,  
`NUMSTREAMTYPE` }
- The enumeration definitions for transport layer nodes.*
- enum `spinTLStreamDefaultBufferCountModeEnums` {  
`StreamDefaultBufferCountMode_Manual`,  
`StreamDefaultBufferCountMode_Auto`,  
`NUMSTREAMDEFAULTBUFFERCOUNTMODE` }
- enum `spinTLStreamBufferCountModeEnums` {  
`StreamBufferCountMode_Manual`,  
`StreamBufferCountMode_Auto`,  
`NUMSTREAMBUFFERCOUNTMODE` }
- enum `spinTLStreamBufferHandlingModeEnums` {  
`StreamBufferHandlingMode_OldestFirst`,  
`StreamBufferHandlingMode_OldestFirstOverwrite`,  
`StreamBufferHandlingMode_NewestFirst`,  
`StreamBufferHandlingMode_NewestFirstOverwrite`,  
`StreamBufferHandlingMode_NewestOnly`,  
`NUMSTREAMBUFFERHANDLINGMODE` }
- enum `spinTLDeviceTypeEnums` {  
`DeviceType_Mixed`,  
`DeviceType_Custom`,  
`DeviceType_GEV`,  
`DeviceType_CL`,  
`DeviceType_IIDC`,  
`DeviceType_UVC`,  
`DeviceType_CXP`,  
`DeviceType_CLHS`,  
`DeviceType_U3V`,  
`DeviceType_ETHERNET`,  
`DeviceType_PCI`,  
`NUMDEVICETYPE` }

- enum `spinTLDeviceAccessStatusEnums` {  
`DeviceAccessStatus_Unknown`,  
`DeviceAccessStatus_ReadWrite`,  
`DeviceAccessStatus_ReadOnly`,  
`DeviceAccessStatus_NoAccess`,  
`NUMDEVICEACCESSSTATUS` }
- enum `spinTLGevCCPEnums` {  
`GevCCP_EnumEntry_GevCCP_OpenAccess`,  
`GevCCP_EnumEntry_GevCCP_ExclusiveAccess`,  
`GevCCP_EnumEntry_GevCCP_ControlAccess`,  
`NUMGEVCCP` }
- enum `spinTLGUIXMLLocationEnums` {  
`GUIXMLLocation_Device`,  
`GUIXMLLocation_Host`,  
`NUMGUIXMLLOCATION` }
- enum `spinTLGenICamXMLLocationEnums` {  
`GenICamXMLLocation_Device`,  
`GenICamXMLLocation_Host`,  
`NUMGENICAMXMLLOCATION` }
- enum `spinTLDeviceEndiannessMechanismEnums` {  
`DeviceEndiannessMechanism_Legacy`,  
`DeviceEndiannessMechanism_Standard`,  
`NUMDEVICEENDIANESSMECHANISM` }
- enum `spinTLDeviceCurrentSpeedEnums` {  
`DeviceCurrentSpeed_UnknownSpeed`,  
`DeviceCurrentSpeed_LowSpeed`,  
`DeviceCurrentSpeed_FullSpeed`,  
`DeviceCurrentSpeed_HighSpeed`,  
`DeviceCurrentSpeed_SuperSpeed`,  
`NUMDEVICECURRENTSPEED` }
- enum `spinTLPOEStatusEnums` {  
`POEStatus_NotSupported`,  
`POEStatus_PowerOff`,  
`POEStatus_PowerOn`,  
`NUMPOESTATUS` }
- enum `spinTLFilterDriverStatusEnums` {  
`FilterDriverStatus_NotSupported`,  
`FilterDriverStatus_Disabled`,  
`FilterDriverStatus_Enabled`,  
`NUMFILTERDRIVERSTATUS` }

### 6.40.1 Detailed Description

### 6.40.2 Enumeration Type Documentation

#### 6.40.2.1 `spinTLDeviceAccessStatusEnums`

```
enum spinTLDeviceAccessStatusEnums
```

< Gets the access status the transport layer Producer has on the device.

**Enumerator**

DeviceAccessStatus_Unknown	Unknown status
DeviceAccessStatus_ReadWrite	Full access
DeviceAccessStatus_ReadOnly	Read-only access
DeviceAccessStatus_NoAccess	Non-available devices
NUMDEVICEACCESSSTATUS	

**6.40.2.2 spinTLDeviceCurrentSpeedEnums**

enum [spinTLDeviceCurrentSpeedEnums](#)

< The USB Speed that the device is currently operating at.

**Enumerator**

DeviceCurrentSpeed_UnknownSpeed	Unknown-Speed.
DeviceCurrentSpeed_LowSpeed	Low-Speed.
DeviceCurrentSpeed_FullSpeed	Full-Speed.
DeviceCurrentSpeed_HighSpeed	High-Speed.
DeviceCurrentSpeed_SuperSpeed	Super-Speed.
NUMDEVICECURRENTSPEED	

**6.40.2.3 spinTLDeviceEndiannessMechanismEnums**

enum [spinTLDeviceEndiannessMechanismEnums](#)

< Identifies the endianness handling mode.

**Enumerator**

DeviceEndiannessMechanism_Legacy	Handling the device endianness according to GenICam Schema 1.0
DeviceEndiannessMechanism_Standard	Handling the device endianness according to GenICam Schema 1.1 and later
NUMDEVICEENDIANESSMECHANISM	

**6.40.2.4 spinTLDeviceTypeEnums**

enum [spinTLDeviceTypeEnums](#)

< Transport layer type of the device.

## Enumerator

DeviceType_Mixed	TL - Mixed
DeviceType_Custom	TL - Custom
DeviceType_GEV	TL - GEV
DeviceType_CL	TL - CL
DeviceType_IIDC	TL - IIDC
DeviceType_UVC	TL - UVC
DeviceType_CXP	TL - CXP
DeviceType_CLHS	TL - CLHS
DeviceType_U3V	TL - U3V
DeviceType_ETHERNET	TL - ETHERNET
DeviceType_PCI	TL - PCI
NUMDEVICETYPE	

## 6.40.2.5 spinTLFilterDriverStatusEnums

```
enum spinTLFilterDriverStatusEnums
```

< Reports whether FLIR Light Weight Filter Driver is enabled or not.

## Enumerator

FilterDriverStatus_NotSupported	Not Supported
FilterDriverStatus_Disabled	FLIR Light Weight Filter Driver is disabled
FilterDriverStatus_Enabled	FLIR Light Weight Filter Driver is enabled
NUMFILTERDRIVERSTATUS	

## 6.40.2.6 spinTLGenICamXMLLocationEnums

```
enum spinTLGenICamXMLLocationEnums
```

< Sets the location to load GenICam XML.

## Enumerator

GenICamXMLLocation_Device	Load GenICam XML from device
GenICamXMLLocation_Host	Load GenICam XML from host
NUMGENICAMXMLLOCATION	

#### 6.40.2.7 spinTLGevCCPEnums

enum [spinTLGevCCPEnums](#)

< Controls the device access privilege of an application.

##### Enumerator

GevCCP_EnumEntry_GevCCP_OpenAccess	Open access privilege.
GevCCP_EnumEntry_GevCCP_ExclusiveAccess	Exclusive access privilege.
GevCCP_EnumEntry_GevCCP_ControlAccess	Control access privilege.
NUMGEVCCP	

#### 6.40.2.8 spinTLGUIXMLLocationEnums

enum [spinTLGUIXMLLocationEnums](#)

< Sets the location to load GUI XML.

##### Enumerator

GUIXMLLocation_Device	Load XML from device
GUIXMLLocation_Host	Load XML from host
NUMGUIXMLLOCATION	

#### 6.40.2.9 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	

#### 6.40.2.10 spinTLStreamBufferCountModeEnums

enum [spinTLStreamBufferCountModeEnums](#)



< Controls access to setting the number of buffers used for the stream. Locked to Manual mode on 32-bit Windows due to memory constraints.

#### Enumerator

StreamBufferCountMode_Manual	The number of buffers used for the stream are set by the user.
StreamBufferCountMode_Auto	The number of buffers used for the stream is automatically calculated based on the device frame rate.
NUMSTREAMBUFFERCOUNTMODE	

#### 6.40.2.11 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_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.
StreamBufferHandlingMode_NewestFirstOverwrite	DEPRECATED. This is replaced by NewestOnly.
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.
NUMSTREAMBUFFERHANDLINGMODE	

#### 6.40.2.12 spinTLStreamDefaultBufferCountModeEnums

```
enum spinTLStreamDefaultBufferCountModeEnums
```

< DEPRECATED; Replaced by StreamBufferCountMode. Controls access to setting the number of buffers used for the stream. Locked to Manual mode on 32-bit Windows due to memory constraints.

#### Enumerator

StreamDefaultBufferCountMode_Manual	DEPRECATED. The number of buffers used for the stream are set by the user.
StreamDefaultBufferCountMode_Auto	DEPRECATED. The number of buffers used for the stream is automatically calculated.
NUMSTREAMDEFAULTBUFFERCOUNTMODE	

#### 6.40.2.13 spinTLStreamTypeEnums

enum [spinTLStreamTypeEnums](#)

The enumeration definitions for transport layer nodes.

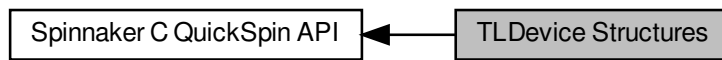
< Stream type of the device.

#### Enumerator

StreamType_Mixed	Stream Type - Mixed
StreamType_Custom	Stream Type - Custom
StreamType_GEV	Stream Type - GEV
StreamType_CL	Stream Type - CL
StreamType_IIDC	Stream Type - IIDC
StreamType_UVC	Stream Type - UVC
StreamType_CXP	Stream Type - CXP
StreamType_CLHS	Stream Type - CLHS
StreamType_U3V	Stream Type - U3V
StreamType_ETHERNET	Stream Type - ETHERNET
StreamType_PCI	Stream Type - PCI
NUMSTREAMTYPE	

## 6.41 TLDevice Structures

Collaboration diagram for TLDevice Structures:



### Data Structures

- struct [quickSpinTLDevice](#)

#### 6.41.1 Detailed Description

## 6.42 TLInterface Structures

Collaboration diagram for TLInterface Structures:



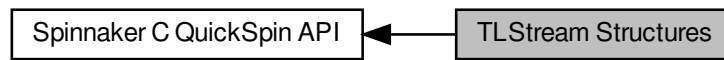
### Data Structures

- struct [quickSpinTLInterface](#)

#### 6.42.1 Detailed Description

## 6.43 TLStream Structures

Collaboration diagram for TLStream Structures:



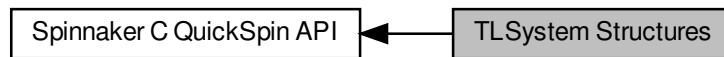
### Data Structures

- struct [quickSpinTLStream](#)

#### 6.43.1 Detailed Description

## 6.44 TLSystem Structures

Collaboration diagram for TLSystem Structures:



### Data Structures

- struct [quickSpinTLSystem](#)

#### 6.44.1 Detailed Description

## Chapter 7

# Data Structure Documentation

### 7.1 actionCommandResult Struct Reference

Action Command Result.

#### Data Fields

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

#### 7.1.1 Detailed Description

Action Command Result.

#### 7.1.2 Field Documentation

##### 7.1.2.1 DeviceAddress

```
unsigned int DeviceAddress
```

##### 7.1.2.2 Status

```
actionCommandStatus Status
```

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

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

## 7.2 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](#)
- [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](#)
- [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 ChunkInferenceResult](#)
- [quickSpinFloatNode ChunkInferenceConfidence](#)

## 7.2.1 Field Documentation

### 7.2.1.1 AasRoiEnable

[quickSpinBooleanNode](#) AasRoiEnable

### 7.2.1.2 AasRoiHeight

[quickSpinIntegerNode](#) AasRoiHeight

### 7.2.1.3 AasRoiOffsetX

[quickSpinIntegerNode](#) AasRoiOffsetX

### 7.2.1.4 AasRoiOffsetY

[quickSpinIntegerNode](#) AasRoiOffsetY

### 7.2.1.5 AasRoiWidth

[quickSpinIntegerNode](#) AasRoiWidth

#### 7.2.1.6 AcquisitionAbort

[quickSpinCommandNode](#) AcquisitionAbort

#### 7.2.1.7 AcquisitionArm

[quickSpinCommandNode](#) AcquisitionArm

#### 7.2.1.8 AcquisitionBurstFrameCount

[quickSpinIntegerNode](#) AcquisitionBurstFrameCount

#### 7.2.1.9 AcquisitionFrameCount

[quickSpinIntegerNode](#) AcquisitionFrameCount

#### 7.2.1.10 AcquisitionFrameRate

[quickSpinFloatNode](#) AcquisitionFrameRate

#### 7.2.1.11 AcquisitionFrameRateEnable

[quickSpinBooleanNode](#) AcquisitionFrameRateEnable

#### 7.2.1.12 AcquisitionLineRate

[quickSpinFloatNode](#) AcquisitionLineRate

#### 7.2.1.13 AcquisitionMode

[quickSpinEnumerationNode](#) AcquisitionMode

#### 7.2.1.14 AcquisitionResultingFrameRate

`quickSpinFloatNode` AcquisitionResultingFrameRate

#### 7.2.1.15 AcquisitionStart

`quickSpinCommandNode` AcquisitionStart

#### 7.2.1.16 AcquisitionStatus

`quickSpinBooleanNode` AcquisitionStatus

#### 7.2.1.17 AcquisitionStatusSelector

`quickSpinEnumerationNode` AcquisitionStatusSelector

#### 7.2.1.18 AcquisitionStop

`quickSpinCommandNode` AcquisitionStop

#### 7.2.1.19 ActionDeviceKey

`quickSpinIntegerNode` ActionDeviceKey

#### 7.2.1.20 ActionGroupKey

`quickSpinIntegerNode` ActionGroupKey

#### 7.2.1.21 ActionGroupMask

`quickSpinIntegerNode` ActionGroupMask

#### 7.2.1.22 ActionQueueSize

[quickSpinIntegerNode](#) ActionQueueSize

#### 7.2.1.23 ActionSelector

[quickSpinIntegerNode](#) ActionSelector

#### 7.2.1.24 ActionUnconditionalMode

[quickSpinEnumerationNode](#) ActionUnconditionalMode

#### 7.2.1.25 AdaptiveCompressionEnable

[quickSpinBooleanNode](#) AdaptiveCompressionEnable

#### 7.2.1.26 AdcBitDepth

[quickSpinEnumerationNode](#) AdcBitDepth

#### 7.2.1.27 aPAUSEMACCtrlFramesReceived

[quickSpinIntegerNode](#) aPAUSEMACCtrlFramesReceived

#### 7.2.1.28 aPAUSEMACCtrlFramesTransmitted

[quickSpinIntegerNode](#) aPAUSEMACCtrlFramesTransmitted

#### 7.2.1.29 AutoAlgorithmSelector

[quickSpinEnumerationNode](#) AutoAlgorithmSelector

#### 7.2.1.30 AutoExposureControlLoopDamping

`quickSpinFloatNode` AutoExposureControlLoopDamping

#### 7.2.1.31 AutoExposureControlPriority

`quickSpinEnumerationNode` AutoExposureControlPriority

#### 7.2.1.32 AutoExposureEVCompensation

`quickSpinFloatNode` AutoExposureEVCompensation

#### 7.2.1.33 AutoExposureExposureTimeLowerLimit

`quickSpinFloatNode` AutoExposureExposureTimeLowerLimit

#### 7.2.1.34 AutoExposureExposureTimeUpperLimit

`quickSpinFloatNode` AutoExposureExposureTimeUpperLimit

#### 7.2.1.35 AutoExposureGainLowerLimit

`quickSpinFloatNode` AutoExposureGainLowerLimit

#### 7.2.1.36 AutoExposureGainUpperLimit

`quickSpinFloatNode` AutoExposureGainUpperLimit

#### 7.2.1.37 AutoExposureGreyValueLowerLimit

`quickSpinFloatNode` AutoExposureGreyValueLowerLimit



### 7.2.1.38 AutoExposureGreyValueUpperLimit

`quickSpinFloatNode` AutoExposureGreyValueUpperLimit

### 7.2.1.39 AutoExposureLightingMode

`quickSpinEnumerationNode` AutoExposureLightingMode

### 7.2.1.40 AutoExposureMeteringMode

`quickSpinEnumerationNode` AutoExposureMeteringMode

### 7.2.1.41 AutoExposureTargetGreyValue

`quickSpinFloatNode` AutoExposureTargetGreyValue

### 7.2.1.42 AutoExposureTargetGreyValueAuto

`quickSpinEnumerationNode` AutoExposureTargetGreyValueAuto

### 7.2.1.43 BalanceRatio

`quickSpinFloatNode` BalanceRatio

### 7.2.1.44 BalanceRatioSelector

`quickSpinEnumerationNode` BalanceRatioSelector

### 7.2.1.45 BalanceWhiteAuto

`quickSpinEnumerationNode` BalanceWhiteAuto

**7.2.1.46 BalanceWhiteAutoDamping**

`quickSpinFloatNode` `BalanceWhiteAutoDamping`

**7.2.1.47 BalanceWhiteAutoLowerLimit**

`quickSpinFloatNode` `BalanceWhiteAutoLowerLimit`

**7.2.1.48 BalanceWhiteAutoProfile**

`quickSpinEnumerationNode` `BalanceWhiteAutoProfile`

**7.2.1.49 BalanceWhiteAutoUpperLimit**

`quickSpinFloatNode` `BalanceWhiteAutoUpperLimit`

**7.2.1.50 BinningHorizontal**

`quickSpinIntegerNode` `BinningHorizontal`

**7.2.1.51 BinningHorizontalMode**

`quickSpinEnumerationNode` `BinningHorizontalMode`

**7.2.1.52 BinningSelector**

`quickSpinEnumerationNode` `BinningSelector`

**7.2.1.53 BinningVertical**

`quickSpinIntegerNode` `BinningVertical`

#### 7.2.1.54 BinningVerticalMode

`quickSpinEnumerationNode` BinningVerticalMode

#### 7.2.1.55 BlackLevel

`quickSpinFloatNode` BlackLevel

#### 7.2.1.56 BlackLevelAuto

`quickSpinEnumerationNode` BlackLevelAuto

#### 7.2.1.57 BlackLevelAutoBalance

`quickSpinEnumerationNode` BlackLevelAutoBalance

#### 7.2.1.58 BlackLevelClampingEnable

`quickSpinBooleanNode` BlackLevelClampingEnable

#### 7.2.1.59 BlackLevelRaw

`quickSpinIntegerNode` BlackLevelRaw

#### 7.2.1.60 BlackLevelSelector

`quickSpinEnumerationNode` BlackLevelSelector

#### 7.2.1.61 ChunkBlackLevel

`quickSpinFloatNode` ChunkBlackLevel

#### 7.2.1.62 ChunkBlackLevelSelector

[quickSpinEnumerationNode](#) ChunkBlackLevelSelector

#### 7.2.1.63 ChunkCounterSelector

[quickSpinEnumerationNode](#) ChunkCounterSelector

#### 7.2.1.64 ChunkCounterValue

[quickSpinIntegerNode](#) ChunkCounterValue

#### 7.2.1.65 ChunkCRC

[quickSpinIntegerNode](#) ChunkCRC

#### 7.2.1.66 ChunkEnable

[quickSpinBooleanNode](#) ChunkEnable

#### 7.2.1.67 ChunkEncoderSelector

[quickSpinEnumerationNode](#) ChunkEncoderSelector

#### 7.2.1.68 ChunkEncoderStatus

[quickSpinEnumerationNode](#) ChunkEncoderStatus

#### 7.2.1.69 ChunkEncoderValue

[quickSpinIntegerNode](#) ChunkEncoderValue

### 7.2.1.70 ChunkExposureEndLineStatusAll

`quickSpinIntegerNode` ChunkExposureEndLineStatusAll

### 7.2.1.71 ChunkExposureTime

`quickSpinFloatNode` ChunkExposureTime

### 7.2.1.72 ChunkExposureTimeSelector

`quickSpinEnumerationNode` ChunkExposureTimeSelector

### 7.2.1.73 ChunkFrameID

`quickSpinIntegerNode` ChunkFrameID

### 7.2.1.74 ChunkGain

`quickSpinFloatNode` ChunkGain

### 7.2.1.75 ChunkGainSelector

`quickSpinEnumerationNode` ChunkGainSelector

### 7.2.1.76 ChunkHeight

`quickSpinIntegerNode` ChunkHeight

### 7.2.1.77 ChunkImage

`quickSpinIntegerNode` ChunkImage

**7.2.1.78 ChunkImageComponent**

[quickSpinEnumerationNode](#) ChunkImageComponent

**7.2.1.79 ChunkInferenceConfidence**

[quickSpinFloatNode](#) ChunkInferenceConfidence

**7.2.1.80 ChunkInferenceResult**

[quickSpinIntegerNode](#) ChunkInferenceResult

**7.2.1.81 ChunkLinePitch**

[quickSpinIntegerNode](#) ChunkLinePitch

**7.2.1.82 ChunkLineStatusAll**

[quickSpinIntegerNode](#) ChunkLineStatusAll

**7.2.1.83 ChunkModeActive**

[quickSpinBooleanNode](#) ChunkModeActive

**7.2.1.84 ChunkOffsetX**

[quickSpinIntegerNode](#) ChunkOffsetX

**7.2.1.85 ChunkOffsetY**

[quickSpinIntegerNode](#) ChunkOffsetY

### 7.2.1.86 ChunkPartSelector

[quickSpinIntegerNode](#) ChunkPartSelector

### 7.2.1.87 ChunkPixelDynamicRangeMax

[quickSpinIntegerNode](#) ChunkPixelDynamicRangeMax

### 7.2.1.88 ChunkPixelDynamicRangeMin

[quickSpinIntegerNode](#) ChunkPixelDynamicRangeMin

### 7.2.1.89 ChunkPixelFormat

[quickSpinEnumerationNode](#) ChunkPixelFormat

### 7.2.1.90 ChunkRegionID

[quickSpinEnumerationNode](#) ChunkRegionID

### 7.2.1.91 ChunkScan3dAxisMax

[quickSpinFloatNode](#) ChunkScan3dAxisMax

### 7.2.1.92 ChunkScan3dAxisMin

[quickSpinFloatNode](#) ChunkScan3dAxisMin

### 7.2.1.93 ChunkScan3dCoordinateOffset

[quickSpinFloatNode](#) ChunkScan3dCoordinateOffset

#### 7.2.1.94 ChunkScan3dCoordinateReferenceSelector

[quickSpinEnumerationNode](#) ChunkScan3dCoordinateReferenceSelector

#### 7.2.1.95 ChunkScan3dCoordinateReferenceValue

[quickSpinFloatNode](#) ChunkScan3dCoordinateReferenceValue

#### 7.2.1.96 ChunkScan3dCoordinateScale

[quickSpinFloatNode](#) ChunkScan3dCoordinateScale

#### 7.2.1.97 ChunkScan3dCoordinateSelector

[quickSpinEnumerationNode](#) ChunkScan3dCoordinateSelector

#### 7.2.1.98 ChunkScan3dCoordinateSystem

[quickSpinEnumerationNode](#) ChunkScan3dCoordinateSystem

#### 7.2.1.99 ChunkScan3dCoordinateSystemReference

[quickSpinEnumerationNode](#) ChunkScan3dCoordinateSystemReference

#### 7.2.1.100 ChunkScan3dCoordinateTransformSelector

[quickSpinEnumerationNode](#) ChunkScan3dCoordinateTransformSelector

#### 7.2.1.101 ChunkScan3dDistanceUnit

[quickSpinEnumerationNode](#) ChunkScan3dDistanceUnit



**7.2.1.102 ChunkScan3dInvalidDataFlag**

[quickSpinBooleanNode](#) ChunkScan3dInvalidDataFlag

**7.2.1.103 ChunkScan3dInvalidDataValue**

[quickSpinFloatNode](#) ChunkScan3dInvalidDataValue

**7.2.1.104 ChunkScan3dOutputMode**

[quickSpinEnumerationNode](#) ChunkScan3dOutputMode

**7.2.1.105 ChunkScan3dTransformValue**

[quickSpinFloatNode](#) ChunkScan3dTransformValue

**7.2.1.106 ChunkScanLineSelector**

[quickSpinIntegerNode](#) ChunkScanLineSelector

**7.2.1.107 ChunkSelector**

[quickSpinEnumerationNode](#) ChunkSelector

**7.2.1.108 ChunkSequencerSetActive**

[quickSpinIntegerNode](#) ChunkSequencerSetActive

**7.2.1.109 ChunkSerialData**

[quickSpinStringNode](#) ChunkSerialData

**7.2.1.110 ChunkSerialDataLength**

[quickSpinIntegerNode](#) ChunkSerialDataLength

**7.2.1.111 ChunkSerialReceiveOverflow**

[quickSpinBooleanNode](#) ChunkSerialReceiveOverflow

**7.2.1.112 ChunkSourceID**

[quickSpinEnumerationNode](#) ChunkSourceID

**7.2.1.113 ChunkStreamChannelID**

[quickSpinIntegerNode](#) ChunkStreamChannelID

**7.2.1.114 ChunkTimerSelector**

[quickSpinEnumerationNode](#) ChunkTimerSelector

**7.2.1.115 ChunkTimerValue**

[quickSpinFloatNode](#) ChunkTimerValue

**7.2.1.116 ChunkTimestamp**

[quickSpinIntegerNode](#) ChunkTimestamp

**7.2.1.117 ChunkTimestampLatchValue**

[quickSpinIntegerNode](#) ChunkTimestampLatchValue

**7.2.1.118 ChunkTransferBlockID**

[quickSpinIntegerNode](#) ChunkTransferBlockID

**7.2.1.119 ChunkTransferQueueCurrentBlockCount**

[quickSpinIntegerNode](#) ChunkTransferQueueCurrentBlockCount

**7.2.1.120 ChunkTransferStreamID**

[quickSpinEnumerationNode](#) ChunkTransferStreamID

**7.2.1.121 ChunkWidth**

[quickSpinIntegerNode](#) ChunkWidth

**7.2.1.122 ClConfiguration**

[quickSpinEnumerationNode](#) ClConfiguration

**7.2.1.123 ClTimeSlotsCount**

[quickSpinEnumerationNode](#) ClTimeSlotsCount

**7.2.1.124 ColorTransformationEnable**

[quickSpinBooleanNode](#) ColorTransformationEnable

**7.2.1.125 ColorTransformationSelector**

[quickSpinEnumerationNode](#) ColorTransformationSelector

**7.2.1.126 ColorTransformationValue**

[quickSpinFloatNode](#) ColorTransformationValue

**7.2.1.127 ColorTransformationValueSelector**

[quickSpinEnumerationNode](#) ColorTransformationValueSelector

**7.2.1.128 CompressionRatio**

[quickSpinFloatNode](#) CompressionRatio

**7.2.1.129 CounterDelay**

[quickSpinIntegerNode](#) CounterDelay

**7.2.1.130 CounterDuration**

[quickSpinIntegerNode](#) CounterDuration

**7.2.1.131 CounterEventActivation**

[quickSpinEnumerationNode](#) CounterEventActivation

**7.2.1.132 CounterEventSource**

[quickSpinEnumerationNode](#) CounterEventSource

**7.2.1.133 CounterReset**

[quickSpinCommandNode](#) CounterReset

**7.2.1.134 CounterResetActivation**

[quickSpinEnumerationNode](#) CounterResetActivation

**7.2.1.135 CounterResetSource**

[quickSpinEnumerationNode](#) CounterResetSource

**7.2.1.136 CounterSelector**

[quickSpinEnumerationNode](#) CounterSelector

**7.2.1.137 CounterStatus**

[quickSpinEnumerationNode](#) CounterStatus

**7.2.1.138 CounterTriggerActivation**

[quickSpinEnumerationNode](#) CounterTriggerActivation

**7.2.1.139 CounterTriggerSource**

[quickSpinEnumerationNode](#) CounterTriggerSource

**7.2.1.140 CounterValue**

[quickSpinIntegerNode](#) CounterValue

**7.2.1.141 CounterValueAtReset**

[quickSpinIntegerNode](#) CounterValueAtReset

**7.2.1.142 CxpConnectionSelector**

[quickSpinIntegerNode](#) CxpConnectionSelector

**7.2.1.143 CxpConnectionTestErrorCount**

[quickSpinIntegerNode](#) CxpConnectionTestErrorCount

**7.2.1.144 CxpConnectionTestMode**

[quickSpinEnumerationNode](#) CxpConnectionTestMode

**7.2.1.145 CxpConnectionTestPacketCount**

[quickSpinIntegerNode](#) CxpConnectionTestPacketCount

**7.2.1.146 CxpLinkConfiguration**

[quickSpinEnumerationNode](#) CxpLinkConfiguration

**7.2.1.147 CxpLinkConfigurationPreferred**

[quickSpinEnumerationNode](#) CxpLinkConfigurationPreferred

**7.2.1.148 CxpLinkConfigurationStatus**

[quickSpinEnumerationNode](#) CxpLinkConfigurationStatus

**7.2.1.149 CxpPoCxpAuto**

[quickSpinCommandNode](#) CxpPoCxpAuto

**7.2.1.150 CxpPoCxpStatus**

[quickSpinEnumerationNode](#) CxpPoCxpStatus

**7.2.1.151 CxpPoCxpTripReset**

[quickSpinCommandNode](#) CxpPoCxpTripReset

**7.2.1.152 CxpPoCxpTurnOff**

[quickSpinCommandNode](#) CxpPoCxpTurnOff

**7.2.1.153 DecimationHorizontal**

[quickSpinIntegerNode](#) DecimationHorizontal

**7.2.1.154 DecimationHorizontalMode**

[quickSpinEnumerationNode](#) DecimationHorizontalMode

**7.2.1.155 DecimationSelector**

[quickSpinEnumerationNode](#) DecimationSelector

**7.2.1.156 DecimationVertical**

[quickSpinIntegerNode](#) DecimationVertical

**7.2.1.157 DecimationVerticalMode**

[quickSpinEnumerationNode](#) DecimationVerticalMode

**7.2.1.158 DefectCorrectionMode**

`quickSpinEnumerationNode` DefectCorrectionMode

**7.2.1.159 DefectCorrectStaticEnable**

`quickSpinBooleanNode` DefectCorrectStaticEnable

**7.2.1.160 DefectTableApply**

`quickSpinCommandNode` DefectTableApply

**7.2.1.161 DefectTableCoordinateX**

`quickSpinIntegerNode` DefectTableCoordinateX

**7.2.1.162 DefectTableCoordinateY**

`quickSpinIntegerNode` DefectTableCoordinateY

**7.2.1.163 DefectTableFactoryRestore**

`quickSpinCommandNode` DefectTableFactoryRestore

**7.2.1.164 DefectTableIndex**

`quickSpinIntegerNode` DefectTableIndex

**7.2.1.165 DefectTablePixelCount**

`quickSpinIntegerNode` DefectTablePixelCount



**7.2.1.166 DefectTableSave**

[quickSpinCommandNode](#) DefectTableSave

**7.2.1.167 Deinterlacing**

[quickSpinEnumerationNode](#) Deinterlacing

**7.2.1.168 DeviceCharacterSet**

[quickSpinEnumerationNode](#) DeviceCharacterSet

**7.2.1.169 DeviceClockFrequency**

[quickSpinFloatNode](#) DeviceClockFrequency

**7.2.1.170 DeviceClockSelector**

[quickSpinEnumerationNode](#) DeviceClockSelector

**7.2.1.171 DeviceConnectionSelector**

[quickSpinIntegerNode](#) DeviceConnectionSelector

**7.2.1.172 DeviceConnectionSpeed**

[quickSpinIntegerNode](#) DeviceConnectionSpeed

**7.2.1.173 DeviceConnectionStatus**

[quickSpinEnumerationNode](#) DeviceConnectionStatus

**7.2.1.174 DeviceEventChannelCount**

[quickSpinIntegerNode](#) DeviceEventChannelCount

**7.2.1.175 DeviceFamilyName**

[quickSpinStringNode](#) DeviceFamilyName

**7.2.1.176 DeviceFeaturePersistenceEnd**

[quickSpinCommandNode](#) DeviceFeaturePersistenceEnd

**7.2.1.177 DeviceFeaturePersistenceStart**

[quickSpinCommandNode](#) DeviceFeaturePersistenceStart

**7.2.1.178 DeviceFirmwareVersion**

[quickSpinStringNode](#) DeviceFirmwareVersion

**7.2.1.179 DeviceGenCPVersionMajor**

[quickSpinIntegerNode](#) DeviceGenCPVersionMajor

**7.2.1.180 DeviceGenCPVersionMinor**

[quickSpinIntegerNode](#) DeviceGenCPVersionMinor

**7.2.1.181 DeviceID**

[quickSpinStringNode](#) DeviceID

**7.2.1.182 DeviceIndicatorMode**

[quickSpinEnumerationNode](#) DeviceIndicatorMode

**7.2.1.183 DeviceLinkBandwidthReserve**

[quickSpinFloatNode](#) DeviceLinkBandwidthReserve

**7.2.1.184 DeviceLinkCommandTimeout**

[quickSpinFloatNode](#) DeviceLinkCommandTimeout

**7.2.1.185 DeviceLinkConnectionCount**

[quickSpinIntegerNode](#) DeviceLinkConnectionCount

**7.2.1.186 DeviceLinkCurrentThroughput**

[quickSpinIntegerNode](#) DeviceLinkCurrentThroughput

**7.2.1.187 DeviceLinkHeartbeatMode**

[quickSpinEnumerationNode](#) DeviceLinkHeartbeatMode

**7.2.1.188 DeviceLinkHeartbeatTimeout**

[quickSpinFloatNode](#) DeviceLinkHeartbeatTimeout

**7.2.1.189 DeviceLinkSelector**

[quickSpinIntegerNode](#) DeviceLinkSelector

**7.2.1.190 DeviceLinkSpeed**

[quickSpinIntegerNode](#) DeviceLinkSpeed

**7.2.1.191 DeviceLinkThroughputLimit**

[quickSpinIntegerNode](#) DeviceLinkThroughputLimit

**7.2.1.192 DeviceLinkThroughputLimitMode**

[quickSpinEnumerationNode](#) DeviceLinkThroughputLimitMode

**7.2.1.193 DeviceManifestEntrySelector**

[quickSpinIntegerNode](#) DeviceManifestEntrySelector

**7.2.1.194 DeviceManifestPrimaryURL**

[quickSpinStringNode](#) DeviceManifestPrimaryURL

**7.2.1.195 DeviceManifestSchemaMajorVersion**

[quickSpinIntegerNode](#) DeviceManifestSchemaMajorVersion

**7.2.1.196 DeviceManifestSchemaMinorVersion**

[quickSpinIntegerNode](#) DeviceManifestSchemaMinorVersion

**7.2.1.197 DeviceManifestSecondaryURL**

[quickSpinStringNode](#) DeviceManifestSecondaryURL

**7.2.1.198 DeviceManifestXMLMajorVersion**

[quickSpinIntegerNode](#) DeviceManifestXMLMajorVersion

**7.2.1.199 DeviceManifestXMLMinorVersion**

[quickSpinIntegerNode](#) DeviceManifestXMLMinorVersion

**7.2.1.200 DeviceManifestXMLSubMinorVersion**

[quickSpinIntegerNode](#) DeviceManifestXMLSubMinorVersion

**7.2.1.201 DeviceManufacturerInfo**

[quickSpinStringNode](#) DeviceManufacturerInfo

**7.2.1.202 DeviceMaxThroughput**

[quickSpinIntegerNode](#) DeviceMaxThroughput

**7.2.1.203 DeviceModelName**

[quickSpinStringNode](#) DeviceModelName

**7.2.1.204 DevicePowerSupplySelector**

[quickSpinEnumerationNode](#) DevicePowerSupplySelector

**7.2.1.205 DeviceRegistersCheck**

[quickSpinCommandNode](#) DeviceRegistersCheck

**7.2.1.206 DeviceRegistersEndianness**

[quickSpinEnumerationNode](#) DeviceRegistersEndianness

**7.2.1.207 DeviceRegistersStreamingEnd**

[quickSpinCommandNode](#) DeviceRegistersStreamingEnd

**7.2.1.208 DeviceRegistersStreamingStart**

[quickSpinCommandNode](#) DeviceRegistersStreamingStart

**7.2.1.209 DeviceRegistersValid**

[quickSpinBooleanNode](#) DeviceRegistersValid

**7.2.1.210 DeviceReset**

[quickSpinCommandNode](#) DeviceReset

**7.2.1.211 DeviceScanType**

[quickSpinEnumerationNode](#) DeviceScanType

**7.2.1.212 DeviceSerialNumber**

[quickSpinStringNode](#) DeviceSerialNumber

**7.2.1.213 DeviceSerialPortBaudRate**

[quickSpinEnumerationNode](#) DeviceSerialPortBaudRate

**7.2.1.214 DeviceSerialPortSelector**

`quickSpinEnumerationNode` DeviceSerialPortSelector

**7.2.1.215 DeviceSFNCVersionMajor**

`quickSpinIntegerNode` DeviceSFNCVersionMajor

**7.2.1.216 DeviceSFNCVersionMinor**

`quickSpinIntegerNode` DeviceSFNCVersionMinor

**7.2.1.217 DeviceSFNCVersionSubMinor**

`quickSpinIntegerNode` DeviceSFNCVersionSubMinor

**7.2.1.218 DeviceStreamChannelCount**

`quickSpinIntegerNode` DeviceStreamChannelCount

**7.2.1.219 DeviceStreamChannelEndianness**

`quickSpinEnumerationNode` DeviceStreamChannelEndianness

**7.2.1.220 DeviceStreamChannelLink**

`quickSpinIntegerNode` DeviceStreamChannelLink

**7.2.1.221 DeviceStreamChannelPacketSize**

`quickSpinIntegerNode` DeviceStreamChannelPacketSize

**7.2.1.222 DeviceStreamChannelSelector**

[quickSpinIntegerNode](#) DeviceStreamChannelSelector

**7.2.1.223 DeviceStreamChannelType**

[quickSpinEnumerationNode](#) DeviceStreamChannelType

**7.2.1.224 DeviceTapGeometry**

[quickSpinEnumerationNode](#) DeviceTapGeometry

**7.2.1.225 DeviceTemperature**

[quickSpinFloatNode](#) DeviceTemperature

**7.2.1.226 DeviceTemperatureSelector**

[quickSpinEnumerationNode](#) DeviceTemperatureSelector

**7.2.1.227 DeviceTLType**

[quickSpinEnumerationNode](#) DeviceTLType

**7.2.1.228 DeviceTLVersionMajor**

[quickSpinIntegerNode](#) DeviceTLVersionMajor

**7.2.1.229 DeviceTLVersionMinor**

[quickSpinIntegerNode](#) DeviceTLVersionMinor



**7.2.1.230 DeviceTLVersionSubMinor**

[quickSpinIntegerNode](#) DeviceTLVersionSubMinor

**7.2.1.231 DeviceType**

[quickSpinEnumerationNode](#) DeviceType

**7.2.1.232 DeviceUptime**

[quickSpinIntegerNode](#) DeviceUptime

**7.2.1.233 DeviceUserID**

[quickSpinStringNode](#) DeviceUserID

**7.2.1.234 DeviceVendorName**

[quickSpinStringNode](#) DeviceVendorName

**7.2.1.235 DeviceVersion**

[quickSpinStringNode](#) DeviceVersion

**7.2.1.236 EncoderDivider**

[quickSpinIntegerNode](#) EncoderDivider

**7.2.1.237 EncoderMode**

[quickSpinEnumerationNode](#) EncoderMode

**7.2.1.238 EncoderOutputMode**

[quickSpinEnumerationNode](#) EncoderOutputMode

**7.2.1.239 EncoderReset**

[quickSpinCommandNode](#) EncoderReset

**7.2.1.240 EncoderResetActivation**

[quickSpinEnumerationNode](#) EncoderResetActivation

**7.2.1.241 EncoderResetSource**

[quickSpinEnumerationNode](#) EncoderResetSource

**7.2.1.242 EncoderSelector**

[quickSpinEnumerationNode](#) EncoderSelector

**7.2.1.243 EncoderSourceA**

[quickSpinEnumerationNode](#) EncoderSourceA

**7.2.1.244 EncoderSourceB**

[quickSpinEnumerationNode](#) EncoderSourceB

**7.2.1.245 EncoderStatus**

[quickSpinEnumerationNode](#) EncoderStatus

**7.2.1.246 EncoderTimeout**

[quickSpinFloatNode](#) EncoderTimeout

**7.2.1.247 EncoderValue**

[quickSpinIntegerNode](#) EncoderValue

**7.2.1.248 EncoderValueAtReset**

[quickSpinIntegerNode](#) EncoderValueAtReset

**7.2.1.249 EnumerationCount**

[quickSpinIntegerNode](#) EnumerationCount

**7.2.1.250 EventAcquisitionEnd**

[quickSpinIntegerNode](#) EventAcquisitionEnd

**7.2.1.251 EventAcquisitionEndFrameID**

[quickSpinIntegerNode](#) EventAcquisitionEndFrameID

**7.2.1.252 EventAcquisitionEndTimestamp**

[quickSpinIntegerNode](#) EventAcquisitionEndTimestamp

**7.2.1.253 EventAcquisitionError**

[quickSpinIntegerNode](#) EventAcquisitionError

**7.2.1.254 EventAcquisitionErrorFrameID**

`quickSpinIntegerNode` EventAcquisitionErrorFrameID

**7.2.1.255 EventAcquisitionErrorTimestamp**

`quickSpinIntegerNode` EventAcquisitionErrorTimestamp

**7.2.1.256 EventAcquisitionStart**

`quickSpinIntegerNode` EventAcquisitionStart

**7.2.1.257 EventAcquisitionStartFrameID**

`quickSpinIntegerNode` EventAcquisitionStartFrameID

**7.2.1.258 EventAcquisitionStartTimestamp**

`quickSpinIntegerNode` EventAcquisitionStartTimestamp

**7.2.1.259 EventAcquisitionTransferEnd**

`quickSpinIntegerNode` EventAcquisitionTransferEnd

**7.2.1.260 EventAcquisitionTransferEndFrameID**

`quickSpinIntegerNode` EventAcquisitionTransferEndFrameID

**7.2.1.261 EventAcquisitionTransferEndTimestamp**

`quickSpinIntegerNode` EventAcquisitionTransferEndTimestamp

**7.2.1.262 EventAcquisitionTransferStart**

[quickSpinIntegerNode](#) EventAcquisitionTransferStart

**7.2.1.263 EventAcquisitionTransferStartFrameID**

[quickSpinIntegerNode](#) EventAcquisitionTransferStartFrameID

**7.2.1.264 EventAcquisitionTransferStartTimestamp**

[quickSpinIntegerNode](#) EventAcquisitionTransferStartTimestamp

**7.2.1.265 EventAcquisitionTrigger**

[quickSpinIntegerNode](#) EventAcquisitionTrigger

**7.2.1.266 EventAcquisitionTriggerFrameID**

[quickSpinIntegerNode](#) EventAcquisitionTriggerFrameID

**7.2.1.267 EventAcquisitionTriggerTimestamp**

[quickSpinIntegerNode](#) EventAcquisitionTriggerTimestamp

**7.2.1.268 EventActionLate**

[quickSpinIntegerNode](#) EventActionLate

**7.2.1.269 EventActionLateFrameID**

[quickSpinIntegerNode](#) EventActionLateFrameID

**7.2.1.270 EventActionLateTimestamp**

[quickSpinIntegerNode](#) EventActionLateTimestamp

**7.2.1.271 EventCounter0End**

[quickSpinIntegerNode](#) EventCounter0End

**7.2.1.272 EventCounter0EndFrameID**

[quickSpinIntegerNode](#) EventCounter0EndFrameID

**7.2.1.273 EventCounter0EndTimestamp**

[quickSpinIntegerNode](#) EventCounter0EndTimestamp

**7.2.1.274 EventCounter0Start**

[quickSpinIntegerNode](#) EventCounter0Start

**7.2.1.275 EventCounter0StartFrameID**

[quickSpinIntegerNode](#) EventCounter0StartFrameID

**7.2.1.276 EventCounter0StartTimestamp**

[quickSpinIntegerNode](#) EventCounter0StartTimestamp

**7.2.1.277 EventCounter1End**

[quickSpinIntegerNode](#) EventCounter1End

**7.2.1.278 EventCounter1EndFrameID**

`quickSpinIntegerNode` EventCounter1EndFrameID

**7.2.1.279 EventCounter1EndTimestamp**

`quickSpinIntegerNode` EventCounter1EndTimestamp

**7.2.1.280 EventCounter1Start**

`quickSpinIntegerNode` EventCounter1Start

**7.2.1.281 EventCounter1StartFrameID**

`quickSpinIntegerNode` EventCounter1StartFrameID

**7.2.1.282 EventCounter1StartTimestamp**

`quickSpinIntegerNode` EventCounter1StartTimestamp

**7.2.1.283 EventEncoder0Restarted**

`quickSpinIntegerNode` EventEncoder0Restarted

**7.2.1.284 EventEncoder0RestartedFrameID**

`quickSpinIntegerNode` EventEncoder0RestartedFrameID

**7.2.1.285 EventEncoder0RestartedTimestamp**

`quickSpinIntegerNode` EventEncoder0RestartedTimestamp

**7.2.1.286 EventEncoder0Stopped**

[quickSpinIntegerNode](#) EventEncoder0Stopped

**7.2.1.287 EventEncoder0StoppedFrameID**

[quickSpinIntegerNode](#) EventEncoder0StoppedFrameID

**7.2.1.288 EventEncoder0StoppedTimestamp**

[quickSpinIntegerNode](#) EventEncoder0StoppedTimestamp

**7.2.1.289 EventEncoder1Restarted**

[quickSpinIntegerNode](#) EventEncoder1Restarted

**7.2.1.290 EventEncoder1RestartedFrameID**

[quickSpinIntegerNode](#) EventEncoder1RestartedFrameID

**7.2.1.291 EventEncoder1RestartedTimestamp**

[quickSpinIntegerNode](#) EventEncoder1RestartedTimestamp

**7.2.1.292 EventEncoder1Stopped**

[quickSpinIntegerNode](#) EventEncoder1Stopped

**7.2.1.293 EventEncoder1StoppedFrameID**

[quickSpinIntegerNode](#) EventEncoder1StoppedFrameID



**7.2.1.294 EventEncoder1StoppedTimestamp**

[quickSpinIntegerNode](#) EventEncoder1StoppedTimestamp

**7.2.1.295 EventError**

[quickSpinIntegerNode](#) EventError

**7.2.1.296 EventErrorCode**

[quickSpinIntegerNode](#) EventErrorCode

**7.2.1.297 EventErrorFrameID**

[quickSpinIntegerNode](#) EventErrorFrameID

**7.2.1.298 EventErrorTimestamp**

[quickSpinIntegerNode](#) EventErrorTimestamp

**7.2.1.299 EventExposureEnd**

[quickSpinIntegerNode](#) EventExposureEnd

**7.2.1.300 EventExposureEndFrameID**

[quickSpinIntegerNode](#) EventExposureEndFrameID

**7.2.1.301 EventExposureEndTimestamp**

[quickSpinIntegerNode](#) EventExposureEndTimestamp

**7.2.1.302 EventExposureStart**

[quickSpinIntegerNode](#) EventExposureStart

**7.2.1.303 EventExposureStartFrameID**

[quickSpinIntegerNode](#) EventExposureStartFrameID

**7.2.1.304 EventExposureStartTimestamp**

[quickSpinIntegerNode](#) EventExposureStartTimestamp

**7.2.1.305 EventFrameBurstEnd**

[quickSpinIntegerNode](#) EventFrameBurstEnd

**7.2.1.306 EventFrameBurstEndFrameID**

[quickSpinIntegerNode](#) EventFrameBurstEndFrameID

**7.2.1.307 EventFrameBurstEndTimestamp**

[quickSpinIntegerNode](#) EventFrameBurstEndTimestamp

**7.2.1.308 EventFrameBurstStart**

[quickSpinIntegerNode](#) EventFrameBurstStart

**7.2.1.309 EventFrameBurstStartFrameID**

[quickSpinIntegerNode](#) EventFrameBurstStartFrameID

**7.2.1.310 EventFrameBurstStartTimestamp**

`quickSpinIntegerNode` EventFrameBurstStartTimestamp

**7.2.1.311 EventFrameEnd**

`quickSpinIntegerNode` EventFrameEnd

**7.2.1.312 EventFrameEndFrameID**

`quickSpinIntegerNode` EventFrameEndFrameID

**7.2.1.313 EventFrameEndTimestamp**

`quickSpinIntegerNode` EventFrameEndTimestamp

**7.2.1.314 EventFrameStart**

`quickSpinIntegerNode` EventFrameStart

**7.2.1.315 EventFrameStartFrameID**

`quickSpinIntegerNode` EventFrameStartFrameID

**7.2.1.316 EventFrameStartTimestamp**

`quickSpinIntegerNode` EventFrameStartTimestamp

**7.2.1.317 EventFrameTransferEnd**

`quickSpinIntegerNode` EventFrameTransferEnd

**7.2.1.318 EventFrameTransferEndFrameID**

[quickSpinIntegerNode](#) EventFrameTransferEndFrameID

**7.2.1.319 EventFrameTransferEndTimestamp**

[quickSpinIntegerNode](#) EventFrameTransferEndTimestamp

**7.2.1.320 EventFrameTransferStart**

[quickSpinIntegerNode](#) EventFrameTransferStart

**7.2.1.321 EventFrameTransferStartFrameID**

[quickSpinIntegerNode](#) EventFrameTransferStartFrameID

**7.2.1.322 EventFrameTransferStartTimestamp**

[quickSpinIntegerNode](#) EventFrameTransferStartTimestamp

**7.2.1.323 EventFrameTrigger**

[quickSpinIntegerNode](#) EventFrameTrigger

**7.2.1.324 EventFrameTriggerFrameID**

[quickSpinIntegerNode](#) EventFrameTriggerFrameID

**7.2.1.325 EventFrameTriggerTimestamp**

[quickSpinIntegerNode](#) EventFrameTriggerTimestamp

### 7.2.1.326 EventLine0AnyEdge

[quickSpinIntegerNode](#) EventLine0AnyEdge

### 7.2.1.327 EventLine0AnyEdgeFrameID

[quickSpinIntegerNode](#) EventLine0AnyEdgeFrameID

### 7.2.1.328 EventLine0AnyEdgeTimestamp

[quickSpinIntegerNode](#) EventLine0AnyEdgeTimestamp

### 7.2.1.329 EventLine0FallingEdge

[quickSpinIntegerNode](#) EventLine0FallingEdge

### 7.2.1.330 EventLine0FallingEdgeFrameID

[quickSpinIntegerNode](#) EventLine0FallingEdgeFrameID

### 7.2.1.331 EventLine0FallingEdgeTimestamp

[quickSpinIntegerNode](#) EventLine0FallingEdgeTimestamp

### 7.2.1.332 EventLine0RisingEdge

[quickSpinIntegerNode](#) EventLine0RisingEdge

### 7.2.1.333 EventLine0RisingEdgeFrameID

[quickSpinIntegerNode](#) EventLine0RisingEdgeFrameID

**7.2.1.334 EventLine0RisingEdgeTimestamp**

[quickSpinIntegerNode](#) EventLine0RisingEdgeTimestamp

**7.2.1.335 EventLine1AnyEdge**

[quickSpinIntegerNode](#) EventLine1AnyEdge

**7.2.1.336 EventLine1AnyEdgeFrameID**

[quickSpinIntegerNode](#) EventLine1AnyEdgeFrameID

**7.2.1.337 EventLine1AnyEdgeTimestamp**

[quickSpinIntegerNode](#) EventLine1AnyEdgeTimestamp

**7.2.1.338 EventLine1FallingEdge**

[quickSpinIntegerNode](#) EventLine1FallingEdge

**7.2.1.339 EventLine1FallingEdgeFrameID**

[quickSpinIntegerNode](#) EventLine1FallingEdgeFrameID

**7.2.1.340 EventLine1FallingEdgeTimestamp**

[quickSpinIntegerNode](#) EventLine1FallingEdgeTimestamp

**7.2.1.341 EventLine1RisingEdge**

[quickSpinIntegerNode](#) EventLine1RisingEdge

**7.2.1.342 EventLine1RisingEdgeFrameID**

[quickSpinIntegerNode](#) EventLine1RisingEdgeFrameID

**7.2.1.343 EventLine1RisingEdgeTimestamp**

[quickSpinIntegerNode](#) EventLine1RisingEdgeTimestamp

**7.2.1.344 EventLinkSpeedChange**

[quickSpinIntegerNode](#) EventLinkSpeedChange

**7.2.1.345 EventLinkSpeedChangeFrameID**

[quickSpinIntegerNode](#) EventLinkSpeedChangeFrameID

**7.2.1.346 EventLinkSpeedChangeTimestamp**

[quickSpinIntegerNode](#) EventLinkSpeedChangeTimestamp

**7.2.1.347 EventLinkTrigger0**

[quickSpinIntegerNode](#) EventLinkTrigger0

**7.2.1.348 EventLinkTrigger0FrameID**

[quickSpinIntegerNode](#) EventLinkTrigger0FrameID

**7.2.1.349 EventLinkTrigger0Timestamp**

[quickSpinIntegerNode](#) EventLinkTrigger0Timestamp

**7.2.1.350 EventLinkTrigger1**

`quickSpinIntegerNode` EventLinkTrigger1

**7.2.1.351 EventLinkTrigger1FrameID**

`quickSpinIntegerNode` EventLinkTrigger1FrameID

**7.2.1.352 EventLinkTrigger1Timestamp**

`quickSpinIntegerNode` EventLinkTrigger1Timestamp

**7.2.1.353 EventNotification**

`quickSpinEnumerationNode` EventNotification

**7.2.1.354 EventSelector**

`quickSpinEnumerationNode` EventSelector

**7.2.1.355 EventSequencerSetChange**

`quickSpinIntegerNode` EventSequencerSetChange

**7.2.1.356 EventSequencerSetChangeFrameID**

`quickSpinIntegerNode` EventSequencerSetChangeFrameID

**7.2.1.357 EventSequencerSetChangeTimestamp**

`quickSpinIntegerNode` EventSequencerSetChangeTimestamp



**7.2.1.358 EventSerialData**

[quickSpinStringNode](#) EventSerialData

**7.2.1.359 EventSerialDataLength**

[quickSpinIntegerNode](#) EventSerialDataLength

**7.2.1.360 EventSerialPortReceive**

[quickSpinIntegerNode](#) EventSerialPortReceive

**7.2.1.361 EventSerialPortReceiveTimestamp**

[quickSpinIntegerNode](#) EventSerialPortReceiveTimestamp

**7.2.1.362 EventSerialReceiveOverflow**

[quickSpinBooleanNode](#) EventSerialReceiveOverflow

**7.2.1.363 EventStream0TransferBlockEnd**

[quickSpinIntegerNode](#) EventStream0TransferBlockEnd

**7.2.1.364 EventStream0TransferBlockEndFrameID**

[quickSpinIntegerNode](#) EventStream0TransferBlockEndFrameID

**7.2.1.365 EventStream0TransferBlockEndTimestamp**

[quickSpinIntegerNode](#) EventStream0TransferBlockEndTimestamp

**7.2.1.366 EventStream0TransferBlockStart**

[quickSpinIntegerNode](#) EventStream0TransferBlockStart

**7.2.1.367 EventStream0TransferBlockStartFrameID**

[quickSpinIntegerNode](#) EventStream0TransferBlockStartFrameID

**7.2.1.368 EventStream0TransferBlockStartTimestamp**

[quickSpinIntegerNode](#) EventStream0TransferBlockStartTimestamp

**7.2.1.369 EventStream0TransferBlockTrigger**

[quickSpinIntegerNode](#) EventStream0TransferBlockTrigger

**7.2.1.370 EventStream0TransferBlockTriggerFrameID**

[quickSpinIntegerNode](#) EventStream0TransferBlockTriggerFrameID

**7.2.1.371 EventStream0TransferBlockTriggerTimestamp**

[quickSpinIntegerNode](#) EventStream0TransferBlockTriggerTimestamp

**7.2.1.372 EventStream0TransferBurstEnd**

[quickSpinIntegerNode](#) EventStream0TransferBurstEnd

**7.2.1.373 EventStream0TransferBurstEndFrameID**

[quickSpinIntegerNode](#) EventStream0TransferBurstEndFrameID

**7.2.1.374 EventStream0TransferBurstEndTimestamp**

[quickSpinIntegerNode](#) EventStream0TransferBurstEndTimestamp

**7.2.1.375 EventStream0TransferBurstStart**

[quickSpinIntegerNode](#) EventStream0TransferBurstStart

**7.2.1.376 EventStream0TransferBurstStartFrameID**

[quickSpinIntegerNode](#) EventStream0TransferBurstStartFrameID

**7.2.1.377 EventStream0TransferBurstStartTimestamp**

[quickSpinIntegerNode](#) EventStream0TransferBurstStartTimestamp

**7.2.1.378 EventStream0TransferEnd**

[quickSpinIntegerNode](#) EventStream0TransferEnd

**7.2.1.379 EventStream0TransferEndFrameID**

[quickSpinIntegerNode](#) EventStream0TransferEndFrameID

**7.2.1.380 EventStream0TransferEndTimestamp**

[quickSpinIntegerNode](#) EventStream0TransferEndTimestamp

**7.2.1.381 EventStream0TransferOverflow**

[quickSpinIntegerNode](#) EventStream0TransferOverflow

**7.2.1.382 EventStream0TransferOverflowFrameID**

[quickSpinIntegerNode](#) EventStream0TransferOverflowFrameID

**7.2.1.383 EventStream0TransferOverflowTimestamp**

[quickSpinIntegerNode](#) EventStream0TransferOverflowTimestamp

**7.2.1.384 EventStream0TransferPause**

[quickSpinIntegerNode](#) EventStream0TransferPause

**7.2.1.385 EventStream0TransferPauseFrameID**

[quickSpinIntegerNode](#) EventStream0TransferPauseFrameID

**7.2.1.386 EventStream0TransferPauseTimestamp**

[quickSpinIntegerNode](#) EventStream0TransferPauseTimestamp

**7.2.1.387 EventStream0TransferResume**

[quickSpinIntegerNode](#) EventStream0TransferResume

**7.2.1.388 EventStream0TransferResumeFrameID**

[quickSpinIntegerNode](#) EventStream0TransferResumeFrameID

**7.2.1.389 EventStream0TransferResumeTimestamp**

[quickSpinIntegerNode](#) EventStream0TransferResumeTimestamp

**7.2.1.390 EventStream0TransferStart**

[quickSpinIntegerNode](#) EventStream0TransferStart

**7.2.1.391 EventStream0TransferStartFrameID**

[quickSpinIntegerNode](#) EventStream0TransferStartFrameID

**7.2.1.392 EventStream0TransferStartTimestamp**

[quickSpinIntegerNode](#) EventStream0TransferStartTimestamp

**7.2.1.393 EventTest**

[quickSpinIntegerNode](#) EventTest

**7.2.1.394 EventTestTimestamp**

[quickSpinIntegerNode](#) EventTestTimestamp

**7.2.1.395 EventTimer0End**

[quickSpinIntegerNode](#) EventTimer0End

**7.2.1.396 EventTimer0EndFrameID**

[quickSpinIntegerNode](#) EventTimer0EndFrameID

**7.2.1.397 EventTimer0EndTimestamp**

[quickSpinIntegerNode](#) EventTimer0EndTimestamp

**7.2.1.398 EventTimer0Start**

[quickSpinIntegerNode](#) EventTimer0Start

**7.2.1.399 EventTimer0StartFrameID**

[quickSpinIntegerNode](#) EventTimer0StartFrameID

**7.2.1.400 EventTimer0StartTimestamp**

[quickSpinIntegerNode](#) EventTimer0StartTimestamp

**7.2.1.401 EventTimer1End**

[quickSpinIntegerNode](#) EventTimer1End

**7.2.1.402 EventTimer1EndFrameID**

[quickSpinIntegerNode](#) EventTimer1EndFrameID

**7.2.1.403 EventTimer1EndTimestamp**

[quickSpinIntegerNode](#) EventTimer1EndTimestamp

**7.2.1.404 EventTimer1Start**

[quickSpinIntegerNode](#) EventTimer1Start

**7.2.1.405 EventTimer1StartFrameID**

[quickSpinIntegerNode](#) EventTimer1StartFrameID

**7.2.1.406 EventTimer1StartTimestamp**

`quickSpinIntegerNode` EventTimer1StartTimestamp

**7.2.1.407 ExposureActiveMode**

`quickSpinEnumerationNode` ExposureActiveMode

**7.2.1.408 ExposureAuto**

`quickSpinEnumerationNode` ExposureAuto

**7.2.1.409 ExposureMode**

`quickSpinEnumerationNode` ExposureMode

**7.2.1.410 ExposureTime**

`quickSpinFloatNode` ExposureTime

**7.2.1.411 ExposureTimeMode**

`quickSpinEnumerationNode` ExposureTimeMode

**7.2.1.412 ExposureTimeSelector**

`quickSpinEnumerationNode` ExposureTimeSelector

**7.2.1.413 FactoryReset**

`quickSpinCommandNode` FactoryReset

**7.2.1.414 FileAccessBuffer**

[quickSpinRegisterNode](#) FileAccessBuffer

**7.2.1.415 FileAccessLength**

[quickSpinIntegerNode](#) FileAccessLength

**7.2.1.416 FileAccessOffset**

[quickSpinIntegerNode](#) FileAccessOffset

**7.2.1.417 FileOpenMode**

[quickSpinEnumerationNode](#) FileOpenMode

**7.2.1.418 FileOperationExecute**

[quickSpinCommandNode](#) FileOperationExecute

**7.2.1.419 FileOperationResult**

[quickSpinIntegerNode](#) FileOperationResult

**7.2.1.420 FileOperationSelector**

[quickSpinEnumerationNode](#) FileOperationSelector

**7.2.1.421 FileOperationStatus**

[quickSpinEnumerationNode](#) FileOperationStatus



**7.2.1.422 FileSelector**

`quickSpinEnumerationNode` FileSelector

**7.2.1.423 FileSize**

`quickSpinIntegerNode` FileSize

**7.2.1.424 Gain**

`quickSpinFloatNode` Gain

**7.2.1.425 GainAuto**

`quickSpinEnumerationNode` GainAuto

**7.2.1.426 GainAutoBalance**

`quickSpinEnumerationNode` GainAutoBalance

**7.2.1.427 GainSelector**

`quickSpinEnumerationNode` GainSelector

**7.2.1.428 Gamma**

`quickSpinFloatNode` Gamma

**7.2.1.429 GammaEnable**

`quickSpinBooleanNode` GammaEnable

**7.2.1.430   GevActiveLinkCount**

`quickSpinIntegerNode`   GevActiveLinkCount

**7.2.1.431   GevCCP**

`quickSpinEnumerationNode`   GevCCP

**7.2.1.432   GevCurrentDefaultGateway**

`quickSpinIntegerNode`   GevCurrentDefaultGateway

**7.2.1.433   GevCurrentIPAddress**

`quickSpinIntegerNode`   GevCurrentIPAddress

**7.2.1.434   GevCurrentIPConfigurationDHCP**

`quickSpinBooleanNode`   GevCurrentIPConfigurationDHCP

**7.2.1.435   GevCurrentIPConfigurationLLA**

`quickSpinBooleanNode`   GevCurrentIPConfigurationLLA

**7.2.1.436   GevCurrentIPConfigurationPersistentIP**

`quickSpinBooleanNode`   GevCurrentIPConfigurationPersistentIP

**7.2.1.437   GevCurrentPhysicalLinkConfiguration**

`quickSpinEnumerationNode`   GevCurrentPhysicalLinkConfiguration

**7.2.1.438   GevCurrentSubnetMask**

[quickSpinIntegerNode](#)   GevCurrentSubnetMask

**7.2.1.439   GevDiscoveryAckDelay**

[quickSpinIntegerNode](#)   GevDiscoveryAckDelay

**7.2.1.440   GevFirstURL**

[quickSpinStringNode](#)   GevFirstURL

**7.2.1.441   GevGVCPExtendedStatusCodes**

[quickSpinBooleanNode](#)   GevGVCPExtendedStatusCodes

**7.2.1.442   GevGVCPExtendedStatusCodesSelector**

[quickSpinEnumerationNode](#)   GevGVCPExtendedStatusCodesSelector

**7.2.1.443   GevGVCPHeartbeatDisable**

[quickSpinBooleanNode](#)   GevGVCPHeartbeatDisable

**7.2.1.444   GevGVCPPendingAck**

[quickSpinBooleanNode](#)   GevGVCPPendingAck

**7.2.1.445   GevGVCPPendingTimeout**

[quickSpinIntegerNode](#)   GevGVCPPendingTimeout

**7.2.1.446   GevGVSPExtendedIDMode**

[quickSpinEnumerationNode](#)   GevGVSPExtendedIDMode

**7.2.1.447   GevHeartbeatTimeout**

[quickSpinIntegerNode](#)   GevHeartbeatTimeout

**7.2.1.448   GevIEEE1588**

[quickSpinBooleanNode](#)   GevIEEE1588

**7.2.1.449   GevIEEE1588ClockAccuracy**

[quickSpinEnumerationNode](#)   GevIEEE1588ClockAccuracy

**7.2.1.450   GevIEEE1588Mode**

[quickSpinEnumerationNode](#)   GevIEEE1588Mode

**7.2.1.451   GevIEEE1588Status**

[quickSpinEnumerationNode](#)   GevIEEE1588Status

**7.2.1.452   GevInterfaceSelector**

[quickSpinIntegerNode](#)   GevInterfaceSelector

**7.2.1.453   GevIPConfigurationStatus**

[quickSpinEnumerationNode](#)   GevIPConfigurationStatus

**7.2.1.454   GevMACAddress**

[quickSpinIntegerNode](#)   GevMACAddress

**7.2.1.455   GevMCDA**

[quickSpinIntegerNode](#)   GevMCDA

**7.2.1.456   GevMCPHostPort**

[quickSpinIntegerNode](#)   GevMCPHostPort

**7.2.1.457   GevMCRC**

[quickSpinIntegerNode](#)   GevMCRC

**7.2.1.458   GevMCSP**

[quickSpinIntegerNode](#)   GevMCSP

**7.2.1.459   GevMCTT**

[quickSpinIntegerNode](#)   GevMCTT

**7.2.1.460   GevNumberOfInterfaces**

[quickSpinIntegerNode](#)   GevNumberOfInterfaces

**7.2.1.461   GevPAUSEFrameReception**

[quickSpinBooleanNode](#)   GevPAUSEFrameReception

**7.2.1.462   GevPAUSEFrameTransmission**

`quickSpinBooleanNode`   GevPAUSEFrameTransmission

**7.2.1.463   GevPersistentDefaultGateway**

`quickSpinIntegerNode`   GevPersistentDefaultGateway

**7.2.1.464   GevPersistentIPAddress**

`quickSpinIntegerNode`   GevPersistentIPAddress

**7.2.1.465   GevPersistentSubnetMask**

`quickSpinIntegerNode`   GevPersistentSubnetMask

**7.2.1.466   GevPhysicalLinkConfiguration**

`quickSpinEnumerationNode`   GevPhysicalLinkConfiguration

**7.2.1.467   GevPrimaryApplicationIPAddress**

`quickSpinIntegerNode`   GevPrimaryApplicationIPAddress

**7.2.1.468   GevPrimaryApplicationSocket**

`quickSpinIntegerNode`   GevPrimaryApplicationSocket

**7.2.1.469   GevPrimaryApplicationSwitchoverKey**

`quickSpinIntegerNode`   GevPrimaryApplicationSwitchoverKey

**7.2.1.470   GevSCCFGAllInTransmission**

`quickSpinBooleanNode` `GevSCCFGAllInTransmission`

**7.2.1.471   GevSCCFGExtendedChunkData**

`quickSpinBooleanNode` `GevSCCFGExtendedChunkData`

**7.2.1.472   GevSCCFGPacketResendDestination**

`quickSpinBooleanNode` `GevSCCFGPacketResendDestination`

**7.2.1.473   GevSCCFGUnconditionalStreaming**

`quickSpinBooleanNode` `GevSCCFGUnconditionalStreaming`

**7.2.1.474   GevSCDA**

`quickSpinIntegerNode` `GevSCDA`

**7.2.1.475   GevSCPD**

`quickSpinIntegerNode` `GevSCPD`

**7.2.1.476   GevSCPDDirection**

`quickSpinIntegerNode` `GevSCPDDirection`

**7.2.1.477   GevSCPHostPort**

`quickSpinIntegerNode` `GevSCPHostPort`

**7.2.1.478   GevSCPInterfaceIndex**

[quickSpinIntegerNode](#)   GevSCPInterfaceIndex

**7.2.1.479   GevSCPSBigEndian**

[quickSpinBooleanNode](#)   GevSCPSBigEndian

**7.2.1.480   GevSCPSPDoNotFragment**

[quickSpinBooleanNode](#)   GevSCPSPDoNotFragment

**7.2.1.481   GevSCPSFireTestPacket**

[quickSpinBooleanNode](#)   GevSCPSFireTestPacket

**7.2.1.482   GevSCPSPacketSize**

[quickSpinIntegerNode](#)   GevSCPSPacketSize

**7.2.1.483   GevSCSP**

[quickSpinIntegerNode](#)   GevSCSP

**7.2.1.484   GevSCZoneConfigurationLock**

[quickSpinBooleanNode](#)   GevSCZoneConfigurationLock

**7.2.1.485   GevSCZoneCount**

[quickSpinIntegerNode](#)   GevSCZoneCount



**7.2.1.486   GevSCZoneDirectionAll**

`quickSpinIntegerNode` `GevSCZoneDirectionAll`

**7.2.1.487   GevSecondURL**

`quickSpinStringNode` `GevSecondURL`

**7.2.1.488   GevStreamChannelSelector**

`quickSpinIntegerNode` `GevStreamChannelSelector`

**7.2.1.489   GevSupportedOption**

`quickSpinBooleanNode` `GevSupportedOption`

**7.2.1.490   GevSupportedOptionSelector**

`quickSpinEnumerationNode` `GevSupportedOptionSelector`

**7.2.1.491   GevTimestampTickFrequency**

`quickSpinIntegerNode` `GevTimestampTickFrequency`

**7.2.1.492   GuiXmlManifestAddress**

`quickSpinIntegerNode` `GuiXmlManifestAddress`

**7.2.1.493   Height**

`quickSpinIntegerNode` `Height`

**7.2.1.494 HeightMax**

[quickSpinIntegerNode](#) HeightMax

**7.2.1.495 ImageComponentEnable**

[quickSpinBooleanNode](#) ImageComponentEnable

**7.2.1.496 ImageComponentSelector**

[quickSpinEnumerationNode](#) ImageComponentSelector

**7.2.1.497 ImageCompressionBitrate**

[quickSpinFloatNode](#) ImageCompressionBitrate

**7.2.1.498 ImageCompressionJPEGFormatOption**

[quickSpinEnumerationNode](#) ImageCompressionJPEGFormatOption

**7.2.1.499 ImageCompressionMode**

[quickSpinEnumerationNode](#) ImageCompressionMode

**7.2.1.500 ImageCompressionQuality**

[quickSpinIntegerNode](#) ImageCompressionQuality

**7.2.1.501 ImageCompressionRateOption**

[quickSpinEnumerationNode](#) ImageCompressionRateOption

**7.2.1.502 IspEnable**

`quickSpinBooleanNode` IspEnable

**7.2.1.503 LineFilterWidth**

`quickSpinFloatNode` LineFilterWidth

**7.2.1.504 LineFormat**

`quickSpinEnumerationNode` LineFormat

**7.2.1.505 LineInputFilterSelector**

`quickSpinEnumerationNode` LineInputFilterSelector

**7.2.1.506 LineInverter**

`quickSpinBooleanNode` LineInverter

**7.2.1.507 LineMode**

`quickSpinEnumerationNode` LineMode

**7.2.1.508 LinePitch**

`quickSpinIntegerNode` LinePitch

**7.2.1.509 LineSelector**

`quickSpinEnumerationNode` LineSelector

**7.2.1.510 LineSource**

[quickSpinEnumerationNode](#) LineSource

**7.2.1.511 LineStatus**

[quickSpinBooleanNode](#) LineStatus

**7.2.1.512 LineStatusAll**

[quickSpinIntegerNode](#) LineStatusAll

**7.2.1.513 LinkErrorCount**

[quickSpinIntegerNode](#) LinkErrorCount

**7.2.1.514 LinkUptime**

[quickSpinIntegerNode](#) LinkUptime

**7.2.1.515 LogicBlockLUTInputActivation**

[quickSpinEnumerationNode](#) LogicBlockLUTInputActivation

**7.2.1.516 LogicBlockLUTInputSelector**

[quickSpinEnumerationNode](#) LogicBlockLUTInputSelector

**7.2.1.517 LogicBlockLUTInputSource**

[quickSpinEnumerationNode](#) LogicBlockLUTInputSource

**7.2.1.518 LogicBlockLUTOutputValue**

[quickSpinBooleanNode](#) LogicBlockLUTOutputValue

**7.2.1.519 LogicBlockLUTOutputValueAll**

[quickSpinIntegerNode](#) LogicBlockLUTOutputValueAll

**7.2.1.520 LogicBlockLUTRowIndex**

[quickSpinIntegerNode](#) LogicBlockLUTRowIndex

**7.2.1.521 LogicBlockLUTSelector**

[quickSpinEnumerationNode](#) LogicBlockLUTSelector

**7.2.1.522 LogicBlockSelector**

[quickSpinEnumerationNode](#) LogicBlockSelector

**7.2.1.523 LUTEnable**

[quickSpinBooleanNode](#) LUTEnable

**7.2.1.524 LUTIndex**

[quickSpinIntegerNode](#) LUTIndex

**7.2.1.525 LUTSelector**

[quickSpinEnumerationNode](#) LUTSelector

**7.2.1.526 LUTValue**

[quickSpinIntegerNode](#) LUTValue

**7.2.1.527 LUTValueAll**

[quickSpinRegisterNode](#) LUTValueAll

**7.2.1.528 MaxDeviceResetTime**

[quickSpinIntegerNode](#) MaxDeviceResetTime

**7.2.1.529 OffsetX**

[quickSpinIntegerNode](#) OffsetX

**7.2.1.530 OffsetY**

[quickSpinIntegerNode](#) OffsetY

**7.2.1.531 PacketResendRequestCount**

[quickSpinIntegerNode](#) PacketResendRequestCount

**7.2.1.532 PayloadSize**

[quickSpinIntegerNode](#) PayloadSize

**7.2.1.533 PixelColorFilter**

[quickSpinEnumerationNode](#) PixelColorFilter

**7.2.1.534 PixelDynamicRangeMax**

[quickSpinIntegerNode](#) PixelDynamicRangeMax

**7.2.1.535 PixelDynamicRangeMin**

[quickSpinIntegerNode](#) PixelDynamicRangeMin

**7.2.1.536 PixelFormat**

[quickSpinEnumerationNode](#) PixelFormat

**7.2.1.537 PixelFormatInfoID**

[quickSpinIntegerNode](#) PixelFormatInfoID

**7.2.1.538 PixelFormatInfoSelector**

[quickSpinEnumerationNode](#) PixelFormatInfoSelector

**7.2.1.539 PixelSize**

[quickSpinEnumerationNode](#) PixelSize

**7.2.1.540 PowerSupplyCurrent**

[quickSpinFloatNode](#) PowerSupplyCurrent

**7.2.1.541 PowerSupplyVoltage**

[quickSpinFloatNode](#) PowerSupplyVoltage

**7.2.1.542 RegionDestination**

[quickSpinEnumerationNode](#) RegionDestination

**7.2.1.543 RegionMode**

[quickSpinEnumerationNode](#) RegionMode

**7.2.1.544 RegionSelector**

[quickSpinEnumerationNode](#) RegionSelector

**7.2.1.545 ReverseX**

[quickSpinBooleanNode](#) ReverseX

**7.2.1.546 ReverseY**

[quickSpinBooleanNode](#) ReverseY

**7.2.1.547 RgbTransformLightSource**

[quickSpinEnumerationNode](#) RgbTransformLightSource

**7.2.1.548 Saturation**

[quickSpinFloatNode](#) Saturation

**7.2.1.549 SaturationEnable**

[quickSpinBooleanNode](#) SaturationEnable



**7.2.1.550 Scan3dAxisMax**

`quickSpinFloatNode` Scan3dAxisMax

**7.2.1.551 Scan3dAxisMin**

`quickSpinFloatNode` Scan3dAxisMin

**7.2.1.552 Scan3dCoordinateOffset**

`quickSpinFloatNode` Scan3dCoordinateOffset

**7.2.1.553 Scan3dCoordinateReferenceSelector**

`quickSpinEnumerationNode` Scan3dCoordinateReferenceSelector

**7.2.1.554 Scan3dCoordinateReferenceValue**

`quickSpinFloatNode` Scan3dCoordinateReferenceValue

**7.2.1.555 Scan3dCoordinateScale**

`quickSpinFloatNode` Scan3dCoordinateScale

**7.2.1.556 Scan3dCoordinateSelector**

`quickSpinEnumerationNode` Scan3dCoordinateSelector

**7.2.1.557 Scan3dCoordinateSystem**

`quickSpinEnumerationNode` Scan3dCoordinateSystem

**7.2.1.558 Scan3dCoordinateSystemReference**

[quickSpinEnumerationNode](#) Scan3dCoordinateSystemReference

**7.2.1.559 Scan3dCoordinateTransformSelector**

[quickSpinEnumerationNode](#) Scan3dCoordinateTransformSelector

**7.2.1.560 Scan3dDistanceUnit**

[quickSpinEnumerationNode](#) Scan3dDistanceUnit

**7.2.1.561 Scan3dInvalidDataFlag**

[quickSpinBooleanNode](#) Scan3dInvalidDataFlag

**7.2.1.562 Scan3dInvalidDataValue**

[quickSpinFloatNode](#) Scan3dInvalidDataValue

**7.2.1.563 Scan3dOutputMode**

[quickSpinEnumerationNode](#) Scan3dOutputMode

**7.2.1.564 Scan3dTransformValue**

[quickSpinFloatNode](#) Scan3dTransformValue

**7.2.1.565 SensorDescription**

[quickSpinStringNode](#) SensorDescription

**7.2.1.566 SensorDigitizationTaps**

`quickSpinEnumerationNode` SensorDigitizationTaps

**7.2.1.567 SensorHeight**

`quickSpinIntegerNode` SensorHeight

**7.2.1.568 SensorShutterMode**

`quickSpinEnumerationNode` SensorShutterMode

**7.2.1.569 SensorTaps**

`quickSpinEnumerationNode` SensorTaps

**7.2.1.570 SensorWidth**

`quickSpinIntegerNode` SensorWidth

**7.2.1.571 SequencerConfigurationMode**

`quickSpinEnumerationNode` SequencerConfigurationMode

**7.2.1.572 SequencerConfigurationValid**

`quickSpinEnumerationNode` SequencerConfigurationValid

**7.2.1.573 SequencerFeatureEnable**

`quickSpinBooleanNode` SequencerFeatureEnable

**7.2.1.574 SequencerMode**

[quickSpinEnumerationNode](#) SequencerMode

**7.2.1.575 SequencerPathSelector**

[quickSpinIntegerNode](#) SequencerPathSelector

**7.2.1.576 SequencerSetActive**

[quickSpinIntegerNode](#) SequencerSetActive

**7.2.1.577 SequencerSetLoad**

[quickSpinCommandNode](#) SequencerSetLoad

**7.2.1.578 SequencerSetNext**

[quickSpinIntegerNode](#) SequencerSetNext

**7.2.1.579 SequencerSetSave**

[quickSpinCommandNode](#) SequencerSetSave

**7.2.1.580 SequencerSetSelector**

[quickSpinIntegerNode](#) SequencerSetSelector

**7.2.1.581 SequencerSetStart**

[quickSpinIntegerNode](#) SequencerSetStart

**7.2.1.582 SequencerSetValid**

[quickSpinEnumerationNode](#) SequencerSetValid

**7.2.1.583 SequencerTriggerActivation**

[quickSpinEnumerationNode](#) SequencerTriggerActivation

**7.2.1.584 SequencerTriggerSource**

[quickSpinEnumerationNode](#) SequencerTriggerSource

**7.2.1.585 SerialPortBaudRate**

[quickSpinEnumerationNode](#) SerialPortBaudRate

**7.2.1.586 SerialPortDataBits**

[quickSpinIntegerNode](#) SerialPortDataBits

**7.2.1.587 SerialPortParity**

[quickSpinEnumerationNode](#) SerialPortParity

**7.2.1.588 SerialPortSelector**

[quickSpinEnumerationNode](#) SerialPortSelector

**7.2.1.589 SerialPortSource**

[quickSpinEnumerationNode](#) SerialPortSource

**7.2.1.590 SerialPortStopBits**

`quickSpinEnumerationNode` SerialPortStopBits

**7.2.1.591 SerialReceiveFramingErrorCount**

`quickSpinIntegerNode` SerialReceiveFramingErrorCount

**7.2.1.592 SerialReceiveParityErrorCount**

`quickSpinIntegerNode` SerialReceiveParityErrorCount

**7.2.1.593 SerialReceiveQueueClear**

`quickSpinCommandNode` SerialReceiveQueueClear

**7.2.1.594 SerialReceiveQueueCurrentCharacterCount**

`quickSpinIntegerNode` SerialReceiveQueueCurrentCharacterCount

**7.2.1.595 SerialReceiveQueueMaxCharacterCount**

`quickSpinIntegerNode` SerialReceiveQueueMaxCharacterCount

**7.2.1.596 SerialTransmitQueueCurrentCharacterCount**

`quickSpinIntegerNode` SerialTransmitQueueCurrentCharacterCount

**7.2.1.597 SerialTransmitQueueMaxCharacterCount**

`quickSpinIntegerNode` SerialTransmitQueueMaxCharacterCount

**7.2.1.598 Sharpening**

`quickSpinFloatNode` Sharpening

**7.2.1.599 SharpeningAuto**

`quickSpinBooleanNode` SharpeningAuto

**7.2.1.600 SharpeningEnable**

`quickSpinBooleanNode` SharpeningEnable

**7.2.1.601 SharpeningThreshold**

`quickSpinFloatNode` SharpeningThreshold

**7.2.1.602 SoftwareSignalPulse**

`quickSpinCommandNode` SoftwareSignalPulse

**7.2.1.603 SoftwareSignalSelector**

`quickSpinEnumerationNode` SoftwareSignalSelector

**7.2.1.604 SourceCount**

`quickSpinIntegerNode` SourceCount

**7.2.1.605 SourceSelector**

`quickSpinEnumerationNode` SourceSelector

**7.2.1.606 Test0001**

`quickSpinIntegerNode` Test0001

**7.2.1.607 TestEventGenerate**

`quickSpinCommandNode` TestEventGenerate

**7.2.1.608 TestPattern**

`quickSpinEnumerationNode` TestPattern

**7.2.1.609 TestPatternGeneratorSelector**

`quickSpinEnumerationNode` TestPatternGeneratorSelector

**7.2.1.610 TestPendingAck**

`quickSpinIntegerNode` TestPendingAck

**7.2.1.611 TimerDelay**

`quickSpinFloatNode` TimerDelay

**7.2.1.612 TimerDuration**

`quickSpinFloatNode` TimerDuration

**7.2.1.613 TimerReset**

`quickSpinCommandNode` TimerReset



**7.2.1.614 TimerSelector**

`quickSpinEnumerationNode` TimerSelector

**7.2.1.615 TimerStatus**

`quickSpinEnumerationNode` TimerStatus

**7.2.1.616 TimerTriggerActivation**

`quickSpinEnumerationNode` TimerTriggerActivation

**7.2.1.617 TimerTriggerSource**

`quickSpinEnumerationNode` TimerTriggerSource

**7.2.1.618 TimerValue**

`quickSpinFloatNode` TimerValue

**7.2.1.619 Timestamp**

`quickSpinIntegerNode` Timestamp

**7.2.1.620 TimestampLatch**

`quickSpinCommandNode` TimestampLatch

**7.2.1.621 TimestampLatchValue**

`quickSpinIntegerNode` TimestampLatchValue

**7.2.1.622    TimestampReset**

[quickSpinCommandNode](#)    TimestampReset

**7.2.1.623    TLParamsLocked**

[quickSpinIntegerNode](#)    TLParamsLocked

**7.2.1.624    TransferAbort**

[quickSpinCommandNode](#)    TransferAbort

**7.2.1.625    TransferBlockCount**

[quickSpinIntegerNode](#)    TransferBlockCount

**7.2.1.626    TransferBurstCount**

[quickSpinIntegerNode](#)    TransferBurstCount

**7.2.1.627    TransferComponentSelector**

[quickSpinEnumerationNode](#)    TransferComponentSelector

**7.2.1.628    TransferControlMode**

[quickSpinEnumerationNode](#)    TransferControlMode

**7.2.1.629    TransferOperationMode**

[quickSpinEnumerationNode](#)    TransferOperationMode

**7.2.1.630 TransferPause**

[quickSpinCommandNode](#) TransferPause

**7.2.1.631 TransferQueueCurrentBlockCount**

[quickSpinIntegerNode](#) TransferQueueCurrentBlockCount

**7.2.1.632 TransferQueueMaxBlockCount**

[quickSpinIntegerNode](#) TransferQueueMaxBlockCount

**7.2.1.633 TransferQueueMode**

[quickSpinEnumerationNode](#) TransferQueueMode

**7.2.1.634 TransferQueueOverflowCount**

[quickSpinIntegerNode](#) TransferQueueOverflowCount

**7.2.1.635 TransferResume**

[quickSpinCommandNode](#) TransferResume

**7.2.1.636 TransferSelector**

[quickSpinEnumerationNode](#) TransferSelector

**7.2.1.637 TransferStart**

[quickSpinCommandNode](#) TransferStart

**7.2.1.638 TransferStatus**

[quickSpinBooleanNode](#) TransferStatus

**7.2.1.639 TransferStatusSelector**

[quickSpinEnumerationNode](#) TransferStatusSelector

**7.2.1.640 TransferStop**

[quickSpinCommandNode](#) TransferStop

**7.2.1.641 TransferStreamChannel**

[quickSpinIntegerNode](#) TransferStreamChannel

**7.2.1.642 TransferTriggerActivation**

[quickSpinEnumerationNode](#) TransferTriggerActivation

**7.2.1.643 TransferTriggerMode**

[quickSpinEnumerationNode](#) TransferTriggerMode

**7.2.1.644 TransferTriggerSelector**

[quickSpinEnumerationNode](#) TransferTriggerSelector

**7.2.1.645 TransferTriggerSource**

[quickSpinEnumerationNode](#) TransferTriggerSource

**7.2.1.646 TriggerActivation**

`quickSpinEnumerationNode` TriggerActivation

**7.2.1.647 TriggerDelay**

`quickSpinFloatNode` TriggerDelay

**7.2.1.648 TriggerDivider**

`quickSpinIntegerNode` TriggerDivider

**7.2.1.649 TriggerEventTest**

`quickSpinCommandNode` TriggerEventTest

**7.2.1.650 TriggerMode**

`quickSpinEnumerationNode` TriggerMode

**7.2.1.651 TriggerMultiplier**

`quickSpinIntegerNode` TriggerMultiplier

**7.2.1.652 TriggerOverlap**

`quickSpinEnumerationNode` TriggerOverlap

**7.2.1.653 TriggerSelector**

`quickSpinEnumerationNode` TriggerSelector

**7.2.1.654 TriggerSoftware**

`quickSpinCommandNode` TriggerSoftware

**7.2.1.655 TriggerSource**

`quickSpinEnumerationNode` TriggerSource

**7.2.1.656 UserOutputSelector**

`quickSpinEnumerationNode` UserOutputSelector

**7.2.1.657 UserOutputValue**

`quickSpinBooleanNode` UserOutputValue

**7.2.1.658 UserOutputValueAll**

`quickSpinIntegerNode` UserOutputValueAll

**7.2.1.659 UserOutputValueAllMask**

`quickSpinIntegerNode` UserOutputValueAllMask

**7.2.1.660 UserSetDefault**

`quickSpinEnumerationNode` UserSetDefault

**7.2.1.661 UserSetFeatureEnable**

`quickSpinBooleanNode` UserSetFeatureEnable

**7.2.1.662 UserSetLoad**

`quickSpinCommandNode` UserSetLoad

**7.2.1.663 UserSetSave**

`quickSpinCommandNode` UserSetSave

**7.2.1.664 UserSetSelector**

`quickSpinEnumerationNode` UserSetSelector

**7.2.1.665 V3\_3Enable**

`quickSpinBooleanNode` V3\_3Enable

**7.2.1.666 WhiteClip**

`quickSpinFloatNode` WhiteClip

**7.2.1.667 WhiteClipSelector**

`quickSpinEnumerationNode` WhiteClipSelector

**7.2.1.668 Width**

`quickSpinIntegerNode` Width

### 7.2.1.669 WidthMax

[quickSpinIntegerNode](#) WidthMax

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

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

## 7.3 quickSpinTLDevice Struct Reference

### Data Fields

- [quickSpinStringNode](#) DeviceID
- [quickSpinStringNode](#) DeviceSerialNumber
- [quickSpinStringNode](#) DeviceVendorName
- [quickSpinStringNode](#) DeviceModelName
- [quickSpinEnumerationNode](#) DeviceType
- [quickSpinStringNode](#) DeviceDisplayName
- [quickSpinEnumerationNode](#) DeviceAccessStatus
- [quickSpinStringNode](#) DeviceVersion
- [quickSpinStringNode](#) DeviceUserID
- [quickSpinStringNode](#) DeviceDriverVersion
- [quickSpinBooleanNode](#) DevicesUpdater
- [quickSpinEnumerationNode](#) GevCCP
- [quickSpinEnumerationNode](#) GUIXMLLocation
- [quickSpinStringNode](#) GUIXMLPath
- [quickSpinEnumerationNode](#) GenICamXMLLocation
- [quickSpinStringNode](#) GenICamXMLPath
- [quickSpinIntegerNode](#) GevDeviceIPAddress
- [quickSpinIntegerNode](#) GevDeviceSubnetMask
- [quickSpinIntegerNode](#) GevDeviceMACAddress
- [quickSpinIntegerNode](#) GevDeviceGateway
- [quickSpinIntegerNode](#) DeviceLinkSpeed
- [quickSpinIntegerNode](#) GevVersionMajor
- [quickSpinIntegerNode](#) GevVersionMinor
- [quickSpinBooleanNode](#) GevDeviceModelsBigEndian
- [quickSpinIntegerNode](#) GevDeviceReadAndWriteTimeout
- [quickSpinIntegerNode](#) GevDeviceMaximumRetryCount
- [quickSpinIntegerNode](#) GevDevicePort
- [quickSpinCommandNode](#) GevDeviceDiscoverMaximumPacketSize
- [quickSpinIntegerNode](#) GevDeviceMaximumPacketSize
- [quickSpinBooleanNode](#) GevDevicesWrongSubnet
- [quickSpinCommandNode](#) GevDeviceForceIP
- [quickSpinBooleanNode](#) DeviceMulticastMonitorMode
- [quickSpinEnumerationNode](#) DeviceEndiannessMechanism
- [quickSpinStringNode](#) DeviceInstanceId
- [quickSpinStringNode](#) DeviceLocation
- [quickSpinEnumerationNode](#) DeviceCurrentSpeed
- [quickSpinBooleanNode](#) DeviceU3VProtocol



### 7.3.1 Field Documentation

#### 7.3.1.1 DeviceAccessStatus

[quickSpinEnumerationNode](#) DeviceAccessStatus

#### 7.3.1.2 DeviceCurrentSpeed

[quickSpinEnumerationNode](#) DeviceCurrentSpeed

#### 7.3.1.3 DeviceDisplayName

[quickSpinStringNode](#) DeviceDisplayName

#### 7.3.1.4 DeviceDriverVersion

[quickSpinStringNode](#) DeviceDriverVersion

#### 7.3.1.5 DeviceEndiannessMechanism

[quickSpinEnumerationNode](#) DeviceEndiannessMechanism

#### 7.3.1.6 DeviceID

[quickSpinStringNode](#) DeviceID

#### 7.3.1.7 DeviceInstanceId

[quickSpinStringNode](#) DeviceInstanceId

#### 7.3.1.8 DeviceIsUpdater

[quickSpinBooleanNode](#) DeviceIsUpdater

#### 7.3.1.9 DeviceLinkSpeed

[quickSpinIntegerNode](#) DeviceLinkSpeed

#### 7.3.1.10 DeviceLocation

[quickSpinStringNode](#) DeviceLocation

#### 7.3.1.11 DeviceModelName

[quickSpinStringNode](#) DeviceModelName

#### 7.3.1.12 DeviceMulticastMonitorMode

[quickSpinBooleanNode](#) DeviceMulticastMonitorMode

#### 7.3.1.13 DeviceSerialNumber

[quickSpinStringNode](#) DeviceSerialNumber

#### 7.3.1.14 DeviceType

[quickSpinEnumerationNode](#) DeviceType

#### 7.3.1.15 DeviceU3VProtocol

[quickSpinBooleanNode](#) DeviceU3VProtocol

#### 7.3.1.16 DeviceUserID

[quickSpinStringNode](#) DeviceUserID

#### 7.3.1.17 DeviceVendorName

[quickSpinStringNode](#) DeviceVendorName

#### 7.3.1.18 DeviceVersion

[quickSpinStringNode](#) DeviceVersion

#### 7.3.1.19 GenICamXMLLocation

[quickSpinEnumerationNode](#) GenICamXMLLocation

#### 7.3.1.20 GenICamXMLPath

[quickSpinStringNode](#) GenICamXMLPath

#### 7.3.1.21 GevCCP

[quickSpinEnumerationNode](#) GevCCP

#### 7.3.1.22 GevDeviceDiscoverMaximumPacketSize

[quickSpinCommandNode](#) GevDeviceDiscoverMaximumPacketSize

#### 7.3.1.23 GevDeviceForceIP

[quickSpinCommandNode](#) GevDeviceForceIP

#### 7.3.1.24 **GevDeviceGateway**

[quickSpinIntegerNode](#) `GevDeviceGateway`

#### 7.3.1.25 **GevDeviceIPAddress**

[quickSpinIntegerNode](#) `GevDeviceIPAddress`

#### 7.3.1.26 **GevDeviceIsWrongSubnet**

[quickSpinBooleanNode](#) `GevDeviceIsWrongSubnet`

#### 7.3.1.27 **GevDeviceMACAddress**

[quickSpinIntegerNode](#) `GevDeviceMACAddress`

#### 7.3.1.28 **GevDeviceMaximumPacketSize**

[quickSpinIntegerNode](#) `GevDeviceMaximumPacketSize`

#### 7.3.1.29 **GevDeviceMaximumRetryCount**

[quickSpinIntegerNode](#) `GevDeviceMaximumRetryCount`

#### 7.3.1.30 **GevDeviceModelsBigEndian**

[quickSpinBooleanNode](#) `GevDeviceModeIsBigEndian`

#### 7.3.1.31 **GevDevicePort**

[quickSpinIntegerNode](#) `GevDevicePort`

#### 7.3.1.32 `GevDeviceReadAndWriteTimeout`

`quickSpinIntegerNode` `GevDeviceReadAndWriteTimeout`

#### 7.3.1.33 `GevDeviceSubnetMask`

`quickSpinIntegerNode` `GevDeviceSubnetMask`

#### 7.3.1.34 `GevVersionMajor`

`quickSpinIntegerNode` `GevVersionMajor`

#### 7.3.1.35 `GevVersionMinor`

`quickSpinIntegerNode` `GevVersionMinor`

#### 7.3.1.36 `GUIXMLLocation`

`quickSpinEnumerationNode` `GUIXMLLocation`

#### 7.3.1.37 `GUIXMLPath`

`quickSpinStringNode` `GUIXMLPath`

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

- `include/spinc/TransportLayerDeviceC.h`

## 7.4 quickSpinTLInterface Struct Reference

### Data Fields

- [quickSpinStringNode InterfaceID](#)
- [quickSpinStringNode InterfaceDisplayName](#)
- [quickSpinStringNode InterfaceType](#)
- [quickSpinIntegerNode GevInterfaceGateway](#)
- [quickSpinIntegerNode GevInterfaceMACAddress](#)
- [quickSpinIntegerNode GevInterfaceIPAddress](#)
- [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](#)
- [quickSpinEnumerationNode DeviceAccessStatus](#)
- [quickSpinIntegerNode GevDeviceIPAddress](#)
- [quickSpinIntegerNode GevDeviceSubnetMask](#)
- [quickSpinIntegerNode GevDeviceMACAddress](#)
- [quickSpinCommandNode AutoForceIP](#)
- [quickSpinIntegerNode IncompatibleDeviceCount](#)
- [quickSpinIntegerNode IncompatibleDeviceSelector](#)
- [quickSpinStringNode IncompatibleDeviceID](#)
- [quickSpinStringNode IncompatibleDeviceVendorName](#)
- [quickSpinStringNode IncompatibleDeviceModelName](#)
- [quickSpinIntegerNode IncompatibleGevDeviceIPAddress](#)
- [quickSpinIntegerNode IncompatibleGevDeviceSubnetMask](#)
- [quickSpinIntegerNode IncompatibleGevDeviceMACAddress](#)
- [quickSpinStringNode HostAdapterName](#)
- [quickSpinStringNode HostAdapterVendor](#)
- [quickSpinStringNode HostAdapterDriverVersion](#)

### 7.4.1 Field Documentation

#### 7.4.1.1 ActionCommand

[quickSpinCommandNode](#) ActionCommand

#### 7.4.1.2 AutoForceIP

[quickSpinCommandNode](#) AutoForceIP

#### 7.4.1.3 DeviceAccessStatus

[quickSpinEnumerationNode](#) DeviceAccessStatus

#### 7.4.1.4 DeviceCount

[quickSpinIntegerNode](#) DeviceCount

#### 7.4.1.5 DeviceID

[quickSpinStringNode](#) DeviceID

#### 7.4.1.6 DeviceModelName

[quickSpinStringNode](#) DeviceModelName

#### 7.4.1.7 DeviceSelector

[quickSpinIntegerNode](#) DeviceSelector

#### 7.4.1.8 DeviceUnlock

[quickSpinStringNode](#) DeviceUnlock

#### 7.4.1.9 DeviceUpdateList

[quickSpinCommandNode](#) DeviceUpdateList

#### 7.4.1.10 DeviceVendorName

[quickSpinStringNode](#) DeviceVendorName

#### 7.4.1.11 FilterDriverStatus

[quickSpinEnumerationNode](#) FilterDriverStatus

#### 7.4.1.12 GevActionDeviceKey

[quickSpinIntegerNode](#) GevActionDeviceKey

#### 7.4.1.13 GevActionGroupKey

[quickSpinIntegerNode](#) GevActionGroupKey

#### 7.4.1.14 GevActionGroupMask

[quickSpinIntegerNode](#) GevActionGroupMask

#### 7.4.1.15 GevActionTime

[quickSpinIntegerNode](#) GevActionTime

#### 7.4.1.16 GevDeviceIPAddress

[quickSpinIntegerNode](#) GevDeviceIPAddress



#### 7.4.1.17 `GevDeviceMACAddress`

`quickSpinIntegerNode` `GevDeviceMACAddress`

#### 7.4.1.18 `GevDeviceSubnetMask`

`quickSpinIntegerNode` `GevDeviceSubnetMask`

#### 7.4.1.19 `GevInterfaceGateway`

`quickSpinIntegerNode` `GevInterfaceGateway`

#### 7.4.1.20 `GevInterfaceIPAddress`

`quickSpinIntegerNode` `GevInterfaceIPAddress`

#### 7.4.1.21 `GevInterfaceMACAddress`

`quickSpinIntegerNode` `GevInterfaceMACAddress`

#### 7.4.1.22 `GevInterfaceMTU`

`quickSpinIntegerNode` `GevInterfaceMTU`

#### 7.4.1.23 `GevInterfaceReceiveLinkSpeed`

`quickSpinIntegerNode` `GevInterfaceReceiveLinkSpeed`

#### 7.4.1.24 `GevInterfaceSubnetMask`

`quickSpinIntegerNode` `GevInterfaceSubnetMask`

**7.4.1.25   GevInterfaceTransmitLinkSpeed**

`quickSpinIntegerNode`   GevInterfaceTransmitLinkSpeed

**7.4.1.26   HostAdapterDriverVersion**

`quickSpinStringNode`   HostAdapterDriverVersion

**7.4.1.27   HostAdapterName**

`quickSpinStringNode`   HostAdapterName

**7.4.1.28   HostAdapterVendor**

`quickSpinStringNode`   HostAdapterVendor

**7.4.1.29   IncompatibleDeviceCount**

`quickSpinIntegerNode`   IncompatibleDeviceCount

**7.4.1.30   IncompatibleDeviceID**

`quickSpinStringNode`   IncompatibleDeviceID

**7.4.1.31   IncompatibleDeviceModelName**

`quickSpinStringNode`   IncompatibleDeviceModelName

**7.4.1.32   IncompatibleDeviceSelector**

`quickSpinIntegerNode`   IncompatibleDeviceSelector

#### 7.4.1.33 IncompatibleDeviceVendorName

`quickSpinStringNode` IncompatibleDeviceVendorName

#### 7.4.1.34 IncompatibleGevDeviceIPAddress

`quickSpinIntegerNode` IncompatibleGevDeviceIPAddress

#### 7.4.1.35 IncompatibleGevDeviceMACAddress

`quickSpinIntegerNode` IncompatibleGevDeviceMACAddress

#### 7.4.1.36 IncompatibleGevDeviceSubnetMask

`quickSpinIntegerNode` IncompatibleGevDeviceSubnetMask

#### 7.4.1.37 InterfaceDisplayName

`quickSpinStringNode` InterfaceDisplayName

#### 7.4.1.38 InterfaceID

`quickSpinStringNode` InterfaceID

#### 7.4.1.39 InterfaceType

`quickSpinStringNode` InterfaceType

#### 7.4.1.40 POEStatus

[quickSpinEnumerationNode](#) POEStatus

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

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

## 7.5 quickSpinTLStream Struct Reference

### Data Fields

- [quickSpinStringNode](#) StreamID
- [quickSpinEnumerationNode](#) StreamType
- [quickSpinIntegerNode](#) StreamTotalBufferCount
- [quickSpinIntegerNode](#) StreamDefaultBufferCount
- [quickSpinIntegerNode](#) StreamDefaultBufferCountMax
- [quickSpinEnumerationNode](#) StreamDefaultBufferCountMode
- [quickSpinIntegerNode](#) StreamBufferCountManual
- [quickSpinIntegerNode](#) StreamBufferCountResult
- [quickSpinIntegerNode](#) StreamBufferCountMax
- [quickSpinEnumerationNode](#) StreamBufferCountMode
- [quickSpinEnumerationNode](#) StreamBufferHandlingMode
- [quickSpinBooleanNode](#) StreamCRCCheckEnable
- [quickSpinBooleanNode](#) GevPacketResendMode
- [quickSpinIntegerNode](#) GevMaximumNumberResendRequests
- [quickSpinIntegerNode](#) GevPacketResendTimeout
- [quickSpinIntegerNode](#) GevMaximumNumberResendBuffers
- [quickSpinIntegerNode](#) GevTotalPacketCount
- [quickSpinIntegerNode](#) GevFailedPacketCount
- [quickSpinIntegerNode](#) GevResendPacketCount
- [quickSpinIntegerNode](#) StreamFailedBufferCount
- [quickSpinIntegerNode](#) StreamBufferUnderrunCount
- [quickSpinIntegerNode](#) GevResendRequestCount
- [quickSpinIntegerNode](#) StreamBlockTransferSize

### 7.5.1 Field Documentation

#### 7.5.1.1 GevFailedPacketCount

[quickSpinIntegerNode](#) GevFailedPacketCount

#### 7.5.1.2 `GevMaximumNumberResendBuffers`

`quickSpinIntegerNode` `GevMaximumNumberResendBuffers`

#### 7.5.1.3 `GevMaximumNumberResendRequests`

`quickSpinIntegerNode` `GevMaximumNumberResendRequests`

#### 7.5.1.4 `GevPacketResendMode`

`quickSpinBooleanNode` `GevPacketResendMode`

#### 7.5.1.5 `GevPacketResendTimeout`

`quickSpinIntegerNode` `GevPacketResendTimeout`

#### 7.5.1.6 `GevResendPacketCount`

`quickSpinIntegerNode` `GevResendPacketCount`

#### 7.5.1.7 `GevResendRequestCount`

`quickSpinIntegerNode` `GevResendRequestCount`

#### 7.5.1.8 `GevTotalPacketCount`

`quickSpinIntegerNode` `GevTotalPacketCount`

#### 7.5.1.9 `StreamBlockTransferSize`

`quickSpinIntegerNode` `StreamBlockTransferSize`

#### 7.5.1.10 StreamBufferCountManual

[quickSpinIntegerNode](#) StreamBufferCountManual

#### 7.5.1.11 StreamBufferCountMax

[quickSpinIntegerNode](#) StreamBufferCountMax

#### 7.5.1.12 StreamBufferCountMode

[quickSpinEnumerationNode](#) StreamBufferCountMode

#### 7.5.1.13 StreamBufferCountResult

[quickSpinIntegerNode](#) StreamBufferCountResult

#### 7.5.1.14 StreamBufferHandlingMode

[quickSpinEnumerationNode](#) StreamBufferHandlingMode

#### 7.5.1.15 StreamBufferUnderrunCount

[quickSpinIntegerNode](#) StreamBufferUnderrunCount

#### 7.5.1.16 StreamCRCCheckEnable

[quickSpinBooleanNode](#) StreamCRCCheckEnable

#### 7.5.1.17 StreamDefaultBufferCount

[quickSpinIntegerNode](#) StreamDefaultBufferCount

#### 7.5.1.18 StreamDefaultBufferCountMax

[quickSpinIntegerNode](#) StreamDefaultBufferCountMax

#### 7.5.1.19 StreamDefaultBufferCountMode

[quickSpinEnumerationNode](#) StreamDefaultBufferCountMode

#### 7.5.1.20 StreamFailedBufferCount

[quickSpinIntegerNode](#) StreamFailedBufferCount

#### 7.5.1.21 StreamID

[quickSpinStringNode](#) StreamID

#### 7.5.1.22 StreamTotalBufferCount

[quickSpinIntegerNode](#) StreamTotalBufferCount

#### 7.5.1.23 StreamType

[quickSpinEnumerationNode](#) StreamType

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

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

## 7.6 quickSpinTLSystem Struct Reference

### Data Fields

- [quickSpinBooleanNode EnumerateGEVInterfaces](#)

## 7.6.1 Field Documentation

### 7.6.1.1 EnumerateGEVInterfaces

`quickSpinBooleanNode` EnumerateGEVInterfaces

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

- `include/spinc/TransportLayerSystemC.h`

## 7.7 spinAVIOption Struct Reference

Options for saving uncompressed videos.

### Data Fields

- float `frameRate`  
*Frame rate of the stream.*
- unsigned int `reserved` [256]  
*Reserved for future use.*

### 7.7.1 Detailed Description

Options for saving uncompressed videos.

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

### 7.7.2 Field Documentation

#### 7.7.2.1 frameRate

`float frameRate`

Frame rate of the stream.



### 7.7.2.2 reserved

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

Reserved for future use.

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

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

## 7.8 spinBMPOption Struct Reference

Options for saving BMP images.

### Data Fields

- [bool8\\_t indexedColor\\_8bit](#)
- unsigned int [reserved](#) [16]

*Reserved for future use.*

### 7.8.1 Detailed Description

Options for saving BMP images.

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

### 7.8.2 Field Documentation

#### 7.8.2.1 indexedColor\_8bit

```
bool8_t indexedColor_8bit
```

#### 7.8.2.2 reserved

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

Reserved for future use.

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

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

## 7.9 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\\_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\\_inferenceResult](#)
- double [m\\_inferenceConfidence](#)

### 7.9.1 Detailed Description

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

### 7.9.2 Field Documentation

#### 7.9.2.1 m\_blackLevel

double m\_blackLevel

#### 7.9.2.2 m\_counterValue

int64\_t m\_counterValue

#### 7.9.2.3 m\_cRC

int64\_t m\_cRC

#### 7.9.2.4 m\_encoderValue

int64\_t m\_encoderValue

#### 7.9.2.5 m\_exposureEndLineStatusAll

int64\_t m\_exposureEndLineStatusAll

#### 7.9.2.6 m\_exposureTime

double m\_exposureTime

#### 7.9.2.7 m\_frameID

int64\_t m\_frameID

#### 7.9.2.8 m\_gain

double m\_gain

**7.9.2.9 m\_height**

```
int64_t m_height
```

**7.9.2.10 m\_image**

```
int64_t m_image
```

**7.9.2.11 m\_inferenceConfidence**

```
double m_inferenceConfidence
```

**7.9.2.12 m\_inferenceResult**

```
int64_t m_inferenceResult
```

**7.9.2.13 m\_linePitch**

```
int64_t m_linePitch
```

**7.9.2.14 m\_lineStatusAll**

```
int64_t m_lineStatusAll
```

**7.9.2.15 m\_offsetX**

```
int64_t m_offsetX
```

**7.9.2.16 m\_offsetY**

```
int64_t m_offsetY
```

**7.9.2.17 m\_partSelector**

```
int64_t m_partSelector
```

**7.9.2.18 m\_pixelDynamicRangeMax**

```
int64_t m_pixelDynamicRangeMax
```

**7.9.2.19 m\_pixelDynamicRangeMin**

```
int64_t m_pixelDynamicRangeMin
```

**7.9.2.20 m\_scan3dAxisMax**

```
double m_scan3dAxisMax
```

**7.9.2.21 m\_scan3dAxisMin**

```
double m_scan3dAxisMin
```

**7.9.2.22 m\_scan3dCoordinateOffset**

```
double m_scan3dCoordinateOffset
```

**7.9.2.23 m\_scan3dCoordinateReferenceValue**

```
double m_scan3dCoordinateReferenceValue
```

**7.9.2.24 m\_scan3dCoordinateScale**

```
double m_scan3dCoordinateScale
```

**7.9.2.25 m\_scan3dInvalidDataValue**

```
double m_scan3dInvalidDataValue
```

**7.9.2.26 m\_scan3dTransformValue**

```
double m_scan3dTransformValue
```

**7.9.2.27 m\_scanLineSelector**

```
int64_t m_scanLineSelector
```

**7.9.2.28 m\_sequencerSetActive**

```
int64_t m_sequencerSetActive
```

**7.9.2.29 m\_serialDataLength**

```
int64_t m_serialDataLength
```

**7.9.2.30 m\_streamChannelID**

```
int64_t m_streamChannelID
```

**7.9.2.31 m\_timerValue**

```
double m_timerValue
```

**7.9.2.32 m\_timestamp**

```
int64_t m_timestamp
```

### 7.9.2.33 m\_timestampLatchValue

int64\_t m\_timestampLatchValue

### 7.9.2.34 m\_transferBlockID

int64\_t m\_transferBlockID

### 7.9.2.35 m\_transferQueueCurrentBlockCount

int64\_t m\_transferQueueCurrentBlockCount

### 7.9.2.36 m\_width

int64\_t m\_width

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

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

## 7.10 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.*

### 7.10.1 Detailed Description

Options for saving H264 videos.

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

### 7.10.2 Field Documentation

#### 7.10.2.1 bitrate

```
unsigned int bitrate
```

Bitrate to encode at.

#### 7.10.2.2 frameRate

```
float frameRate
```

Frame rate of the stream.

#### 7.10.2.3 height

```
unsigned int height
```

Height of source image.

#### 7.10.2.4 reserved

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

Reserved for future use.

#### 7.10.2.5 width

```
unsigned int width
```

Width of source image.

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

- `include/spinc/SpinnakerDefsC.h`



## 7.11 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.*

### 7.11.1 Detailed Description

Options for saving JPEG images.

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

### 7.11.2 Field Documentation

#### 7.11.2.1 progressive

`bool8_t progressive`

Whether to save as a progressive JPEG file.

#### 7.11.2.2 quality

`unsigned int quality`

JPEG image quality in range (0-100).

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

### 7.11.2.3 reserved

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

Reserved for future use.

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

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

## 7.12 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.*

### 7.12.1 Detailed Description

Options for saving JPEG 2000 images.

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

### 7.12.2 Field Documentation

#### 7.12.2.1 quality

```
unsigned int quality
```

JPEG saving quality in range (1-512).

#### 7.12.2.2 reserved

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

Reserved for future use.

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

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

## 7.13 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.*

### 7.13.1 Detailed Description

Provides easier access to the current version of Spinnaker.

### 7.13.2 Field Documentation

#### 7.13.2.1 build

```
unsigned int build
```

Build number of the library.

#### 7.13.2.2 major

```
unsigned int major
```

Major version of the library.

#### 7.13.2.3 minor

```
unsigned int minor
```

Minor version of the library.

#### 7.13.2.4 type

`unsigned int type`

Version type of the library.

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

- `include/spinc/SpinnakerDefsC.h`

## 7.14 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 `reserved` [256]

### 7.14.1 Detailed Description

Options for saving MJPG videos.

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

### 7.14.2 Field Documentation

#### 7.14.2.1 frameRate

`float frameRate`

Frame rate of the stream.

#### 7.14.2.2 quality

`unsigned int quality`

Image quality (1-100)

### 7.14.2.3 reserved

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

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

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

## 7.15 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.*

### 7.15.1 Detailed Description

Options for saving PGM images.

### 7.15.2 Field Documentation

#### 7.15.2.1 binaryFile

```
bool8_t binaryFile
```

Whether to save the PPM as a binary file.

#### 7.15.2.2 reserved

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

Reserved for future use.

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

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

## 7.16 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.*

### 7.16.1 Detailed Description

Options for saving PNG images.

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

### 7.16.2 Field Documentation

#### 7.16.2.1 compressionLevel

```
unsigned int compressionLevel
```

Compression level (0-9).

0 is no compression, 9 is best compression.

#### 7.16.2.2 interlaced

```
bool8_t interlaced
```

Whether to save the PNG as interlaced.

#### 7.16.2.3 reserved

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

Reserved for future use.

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

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

## 7.17 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.*

### 7.17.1 Detailed Description

Options for saving PPM images.

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

### 7.17.2 Field Documentation

#### 7.17.2.1 binaryFile

`bool8_t binaryFile`

Whether to save the PPM as a binary file.

#### 7.17.2.2 reserved

`unsigned int reserved[16]`

Reserved for future use.

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

- `include/spinc/SpinnakerDefsC.h`

## 7.18 spinTIFFOption Struct Reference

Options for saving TIFF images.

## Data Fields

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

### 7.18.1 Detailed Description

Options for saving TIFF images.

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

### 7.18.2 Field Documentation

#### 7.18.2.1 [compression](#)

[spinCompressionMethod](#) *compression*

Compression method to use for encoding TIFF images.

#### 7.18.2.2 [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 8

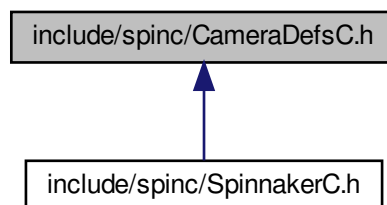
# File Documentation

### 8.1 doc/Doxygen/spindocs/C/Licensing.dox File Reference

### 8.2 doc/Doxygen/spindocs/C/MainPage.dox File Reference

### 8.3 include/spinc/CameraDefsC.h File Reference

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



#### Enumerations

- enum `spinLUTSelectorEnums` {  
    `LUTSelector_LUT1`,  
    `NUM_LUTSELECTOR` }

*The enum definitions for camera nodes.*

- enum `spinExposureModeEnums` {  
    `ExposureMode_Timed`,  
    `ExposureMode_TriggerWidth`,  
    `NUM_EXPOSUREMODE` }

- enum `spinAcquisitionModeEnums` {  
    `AcquisitionMode_Continuous`,  
    `AcquisitionMode_SingleFrame`,  
    `AcquisitionMode_MultiFrame`,  
    `NUM_ACQUISITIONMODE` }
- enum `spinTriggerSourceEnums` {  
    `TriggerSource_Software`,  
    `TriggerSource_Line0`,  
    `TriggerSource_Line1`,  
    `TriggerSource_Line2`,  
    `TriggerSource_Line3`,  
    `TriggerSource_UserOutput0`,  
    `TriggerSource_UserOutput1`,  
    `TriggerSource_UserOutput2`,  
    `TriggerSource_UserOutput3`,  
    `TriggerSource_Counter0Start`,  
    `TriggerSource_Counter1Start`,  
    `TriggerSource_Counter0End`,  
    `TriggerSource_Counter1End`,  
    `TriggerSource_LogicBlock0`,  
    `TriggerSource_LogicBlock1`,  
    `TriggerSource_Action0`,  
    `NUM_TRIGGERSOURCE` }
- enum `spinTriggerActivationEnums` {  
    `TriggerActivation_LevelLow`,  
    `TriggerActivation_LevelHigh`,  
    `TriggerActivation_FallingEdge`,  
    `TriggerActivation_RisingEdge`,  
    `TriggerActivation_AnyEdge`,  
    `NUM_TRIGGERACTIVATION` }
- enum `spinSensorShutterModeEnums` {  
    `SensorShutterMode_Global`,  
    `SensorShutterMode_Rolling`,  
    `SensorShutterMode_GlobalReset`,  
    `NUM_SENSORSHUTTERMODE` }
- enum `spinTriggerModeEnums` {  
    `TriggerMode_Off`,  
    `TriggerMode_On`,  
    `NUM_TRIGGERMODE` }
- enum `spinTriggerOverlapEnums` {  
    `TriggerOverlap_Off`,  
    `TriggerOverlap_ReadOut`,  
    `TriggerOverlap_PreviousFrame`,  
    `NUM_TRIGGEROVERLAP` }
- enum `spinTriggerSelectorEnums` {  
    `TriggerSelector_AcquisitionStart`,  
    `TriggerSelector_FrameStart`,  
    `TriggerSelector_FrameBurstStart`,  
    `NUM_TRIGGERSELECTOR` }
- enum `spinExposureAutoEnums` {  
    `ExposureAuto_Off`,  
    `ExposureAuto_Once`,  
    `ExposureAuto_Continuous`,  
    `NUM_EXPOSUREAUTO` }
- enum `spinEventSelectorEnums` {  
    `EventSelector_Error`,  
    `EventSelector_ExposureEnd`,

```

    EventSelector_SerialPortReceive,
    NUM_EVENTSELECTOR }
• enum spinEventNotificationEnums {
    EventNotification_On,
    EventNotification_Off,
    NUM_EVENTNOTIFICATION }
• enum spinLogicBlockSelectorEnums {
    LogicBlockSelector_LogicBlock0,
    LogicBlockSelector_LogicBlock1,
    NUM_LOGICBLOCKSELECTOR }
• enum spinLogicBlockLUTInputActivationEnums {
    LogicBlockLUTInputActivation_LevelLow,
    LogicBlockLUTInputActivation_LevelHigh,
    LogicBlockLUTInputActivation_FallingEdge,
    LogicBlockLUTInputActivation_RisingEdge,
    LogicBlockLUTInputActivation_AnyEdge,
    NUM_LOGICBLOCKLUTINPUTACTIVATION }
• enum spinLogicBlockLUTInputSelectorEnums {
    LogicBlockLUTInputSelector_Input0,
    LogicBlockLUTInputSelector_Input1,
    LogicBlockLUTInputSelector_Input2,
    LogicBlockLUTInputSelector_Input3,
    NUM_LOGICBLOCKLUTINPUTSELECTOR }
• enum spinLogicBlockLUTInputSourceEnums {
    LogicBlockLUTInputSource_Zero,
    LogicBlockLUTInputSource_Line0,
    LogicBlockLUTInputSource_Line1,
    LogicBlockLUTInputSource_Line2,
    LogicBlockLUTInputSource_Line3,
    LogicBlockLUTInputSource_UserOutput0,
    LogicBlockLUTInputSource_UserOutput1,
    LogicBlockLUTInputSource_UserOutput2,
    LogicBlockLUTInputSource_UserOutput3,
    LogicBlockLUTInputSource_Counter0Start,
    LogicBlockLUTInputSource_Counter1Start,
    LogicBlockLUTInputSource_Counter0End,
    LogicBlockLUTInputSource_Counter1End,
    LogicBlockLUTInputSource_LogicBlock0,
    LogicBlockLUTInputSource_LogicBlock1,
    LogicBlockLUTInputSource_ExposureStart,
    LogicBlockLUTInputSource_ExposureEnd,
    LogicBlockLUTInputSource_FrameTriggerWait,
    LogicBlockLUTInputSource_AcquisitionActive,
    NUM_LOGICBLOCKLUTINPUTSOURCE }
• enum spinLogicBlockLUTSelectorEnums {
    LogicBlockLUTSelector_Value,
    LogicBlockLUTSelector_Enable,
    NUM_LOGICBLOCKLUTSELECTOR }
• enum spinColorTransformationSelectorEnums {
    ColorTransformationSelector_RGBtoRGB,
    ColorTransformationSelector_RGBtoYUV,
    NUM_COLORTRANSFORMATIONSELECTOR }
• enum spinRgbTransformLightSourceEnums {
    RgbTransformLightSource_General,
    RgbTransformLightSource_Tungsten2800K,
    RgbTransformLightSource_WarmFluorescent3000K,
    RgbTransformLightSource_CoolFluorescent4000K,
    RgbTransformLightSource_Daylight5000K,

```

```

    RgbTransformLightSource_Cloudy6500K,
    RgbTransformLightSource_Shade8000K,
    RgbTransformLightSource_Custom,
    NUM_RGBTRANSFORMLIGHTSOURCE }

• enum spinColorTransformationValueSelectorEnums {
    ColorTransformationValueSelector_Gain00,
    ColorTransformationValueSelector_Gain01,
    ColorTransformationValueSelector_Gain02,
    ColorTransformationValueSelector_Gain10,
    ColorTransformationValueSelector_Gain11,
    ColorTransformationValueSelector_Gain12,
    ColorTransformationValueSelector_Gain20,
    ColorTransformationValueSelector_Gain21,
    ColorTransformationValueSelector_Gain22,
    ColorTransformationValueSelector_Offset0,
    ColorTransformationValueSelector_Offset1,
    ColorTransformationValueSelector_Offset2,
    NUM_COLORTRANSFORMATIONVALUESELECTOR }

• enum spinDeviceRegistersEndiannessEnums {
    DeviceRegistersEndianness_Little,
    DeviceRegistersEndianness_Big,
    NUM_DEVICEREGISTERSENDIANNES }

• enum spinDeviceScanTypeEnums {
    DeviceScanType_Areascan,
    NUM_DEVICESCANTYPE }

• enum spinDeviceCharacterSetEnums {
    DeviceCharacterSet_UTF8,
    DeviceCharacterSet_ASCII,
    NUM_DEVICECHARACTERSET }

• enum spinDeviceTLTypeEnums {
    DeviceTLType_GigEVision,
    DeviceTLType_CameraLink,
    DeviceTLType_CameraLinkHS,
    DeviceTLType_CoaXPress,
    DeviceTLType_USB3Vision,
    DeviceTLType_Custom,
    NUM_DEVICETLTYPE }

• enum spinDevicePowerSupplySelectorEnums {
    DevicePowerSupplySelector_External,
    NUM_DEVICEPOWERSUPPLYSELECTOR }

• enum spinDeviceTemperatureSelectorEnums {
    DeviceTemperatureSelector_Sensor,
    NUM_DEVICETEMPERATURESELECTOR }

• enum spinDeviceIndicatorModeEnums {
    DeviceIndicatorMode_Inactive,
    DeviceIndicatorMode_Active,
    DeviceIndicatorMode_ErrorStatus,
    NUM_DEVICEINDICATORMODE }

• enum spinAutoExposureControlPriorityEnums {
    AutoExposureControlPriority_Gain,
    AutoExposureControlPriority_ExposureTime,
    NUM_AUTOEXPOSURECONTROLPRIORITY }

• enum spinAutoExposureMeteringModeEnums {
    AutoExposureMeteringMode_Average,
    AutoExposureMeteringMode_Spot,
    AutoExposureMeteringMode_Partial,
    AutoExposureMeteringMode_CenterWeighted,

```

- AutoExposureMeteringMode\_HistogramPeak,  
NUM\_AUTOEXPOSUREMETERINGMODE }
- enum spinBalanceWhiteAutoProfileEnums {  
BalanceWhiteAutoProfile\_Indoor,  
BalanceWhiteAutoProfile\_Outdoor,  
NUM\_BALANCEWHITEAUTOPROFILE }
- enum spinAutoAlgorithmSelectorEnums {  
AutoAlgorithmSelector\_Awb,  
AutoAlgorithmSelector\_Ae,  
NUM\_AUTOALGORITHMSELECTOR }
- enum spinAutoExposureTargetGreyValueAutoEnums {  
AutoExposureTargetGreyValueAuto\_Off,  
AutoExposureTargetGreyValueAuto\_Continuous,  
NUM\_AUTOEXPOSURETARGETGREYVALUEAUTO }
- enum spinAutoExposureLightingModeEnums {  
AutoExposureLightingMode\_AutoDetect,  
AutoExposureLightingMode\_Backlight,  
AutoExposureLightingMode\_Frontlight,  
AutoExposureLightingMode\_Normal,  
NUM\_AUTOEXPOSURELIGHTINGMODE }
- enum spinGevIEEE1588StatusEnums {  
GevIEEE1588Status\_Initializing,  
GevIEEE1588Status\_Faulty,  
GevIEEE1588Status\_Disabled,  
GevIEEE1588Status\_Listening,  
GevIEEE1588Status\_PreMaster,  
GevIEEE1588Status\_Master,  
GevIEEE1588Status\_Passive,  
GevIEEE1588Status\_Uncalibrated,  
GevIEEE1588Status\_Slave,  
NUM\_GEVIEEE1588STATUS }
- enum spinGevIEEE1588ModeEnums {  
GevIEEE1588Mode\_Auto,  
GevIEEE1588Mode\_SlaveOnly,  
NUM\_GEVIEEE1588MODE }
- enum spinGevIEEE1588ClockAccuracyEnums {  
GevIEEE1588ClockAccuracy\_Unknown,  
NUM\_GEVIEEE1588CLOCKACCURACY }
- enum spinGevCCPEnums {  
GevCCP\_OpenAccess,  
GevCCP\_ExclusiveAccess,  
GevCCP\_ControlAccess,  
NUM\_GEVCCP }
- enum spinGevSupportedOptionSelectorEnums {  
GevSupportedOptionSelector\_UserDefinedName,  
GevSupportedOptionSelector\_SerialNumber,  
GevSupportedOptionSelector\_HeartbeatDisable,  
GevSupportedOptionSelector\_LinkSpeed,  
GevSupportedOptionSelector\_CCPApplicationSocket,  
GevSupportedOptionSelector\_ManifestTable,  
GevSupportedOptionSelector\_TestData,  
GevSupportedOptionSelector\_DiscoveryAckDelay,  
GevSupportedOptionSelector\_DiscoveryAckDelayWritable,  
GevSupportedOptionSelector\_ExtendedStatusCodes,  
GevSupportedOptionSelector\_Action,  
GevSupportedOptionSelector\_PendingAck,  
GevSupportedOptionSelector\_EventData,  
GevSupportedOptionSelector\_Event,

```

GevSupportedOptionSelector_PacketResend,
GevSupportedOptionSelector_WriteMem,
GevSupportedOptionSelector_CommandsConcatenation,
GevSupportedOptionSelector_IPConfigurationLLA,
GevSupportedOptionSelector_IPConfigurationDHCP,
GevSupportedOptionSelector_IPConfigurationPersistentIP,
GevSupportedOptionSelector_StreamChannelSourceSocket,
GevSupportedOptionSelector_MessageChannelSourceSocket,
NUM_GEVSUPPORTEDOPTIONSELECTOR }
• enum spinBlackLevelSelectorEnums {
    BlackLevelSelector_All,
    BlackLevelSelector_Analog,
    BlackLevelSelector_Digital,
    NUM_BLACKLEVELSELECTOR }
• enum spinBalanceWhiteAutoEnums {
    BalanceWhiteAuto_Off,
    BalanceWhiteAuto_Once,
    BalanceWhiteAuto_Continuous,
    NUM_BALANCEWHITEAUTO }
• enum spinGainAutoEnums {
    GainAuto_Off,
    GainAuto_Once,
    GainAuto_Continuous,
    NUM_GAINAUTO }
• enum spinBalanceRatioSelectorEnums {
    BalanceRatioSelector_Red,
    BalanceRatioSelector_Blue,
    NUM_BALANCERATIOSELECTOR }
• enum spinGainSelectorEnums {
    GainSelector_All,
    NUM_GAINSELECTOR }
• enum spinDefectCorrectionModeEnums {
    DefectCorrectionMode_Average,
    DefectCorrectionMode_Highlight,
    DefectCorrectionMode_Zero,
    NUM_DEFECTCORRECTIONMODE }
• enum spinUserSetSelectorEnums {
    UserSetSelector_Default,
    UserSetSelector_UserSet0,
    UserSetSelector_UserSet1,
    NUM_USERSETSELECTOR }
• enum spinUserSetDefaultEnums {
    UserSetDefault_Default,
    UserSetDefault_UserSet0,
    UserSetDefault_UserSet1,
    NUM_USERSETDEFAULT }
• enum spinSerialPortBaudRateEnums {
    SerialPortBaudRate_Baud300,
    SerialPortBaudRate_Baud600,
    SerialPortBaudRate_Baud1200,
    SerialPortBaudRate_Baud2400,
    SerialPortBaudRate_Baud4800,
    SerialPortBaudRate_Baud9600,
    SerialPortBaudRate_Baud14400,
    SerialPortBaudRate_Baud19200,
    SerialPortBaudRate_Baud38400,
    SerialPortBaudRate_Baud57600,
    SerialPortBaudRate_Baud115200,

```

- SerialPortBaudRate\_Baud230400,
- SerialPortBaudRate\_Baud460800,
- SerialPortBaudRate\_Baud921600,
- NUM\_SERIALPORTBAUDRATE }
- enum spinSerialPortParityEnums {
  - SerialPortParity\_None,
  - SerialPortParity\_Odd,
  - SerialPortParity\_Even,
  - SerialPortParity\_Mark,
  - SerialPortParity\_Space,
  - NUM\_SERIALPORTPARITY }
- enum spinSerialPortSelectorEnums {
  - SerialPortSelector\_SerialPort0,
  - NUM\_SERIALPORTSELECTOR }
- enum spinSerialPortStopBitsEnums {
  - SerialPortStopBits\_Bits1,
  - SerialPortStopBits\_Bits1AndAHalf,
  - SerialPortStopBits\_Bits2,
  - NUM\_SERIALPORTSTOPBITS }
- enum spinSerialPortSourceEnums {
  - SerialPortSource\_Line0,
  - SerialPortSource\_Line1,
  - SerialPortSource\_Line2,
  - SerialPortSource\_Line3,
  - SerialPortSource\_Off,
  - NUM\_SERIALPORTSOURCE }
- enum spinSequencerModeEnums {
  - SequencerMode\_Off,
  - SequencerMode\_On,
  - NUM\_SEQUENCERMODE }
- enum spinSequencerConfigurationValidEnums {
  - SequencerConfigurationValid\_No,
  - SequencerConfigurationValid\_Yes,
  - NUM\_SEQUENCERCONFIGURATIONVALID }
- enum spinSequencerSetValidEnums {
  - SequencerSetValid\_No,
  - SequencerSetValid\_Yes,
  - NUM\_SEQUENCERSETVALID }
- enum spinSequencerTriggerActivationEnums {
  - SequencerTriggerActivation\_RisingEdge,
  - SequencerTriggerActivation\_FallingEdge,
  - SequencerTriggerActivation\_AnyEdge,
  - SequencerTriggerActivation\_LevelHigh,
  - SequencerTriggerActivation\_LevelLow,
  - NUM\_SEQUENCERTRIGGERACTIVATION }
- enum spinSequencerConfigurationModeEnums {
  - SequencerConfigurationMode\_Off,
  - SequencerConfigurationMode\_On,
  - NUM\_SEQUENCERCONFIGURATIONMODE }
- enum spinSequencerTriggerSourceEnums {
  - SequencerTriggerSource\_Off,
  - SequencerTriggerSource\_FrameStart,
  - NUM\_SEQUENCERTRIGGERSOURCE }
- enum spinTransferQueueModeEnums {
  - TransferQueueMode\_FirstInFirstOut,
  - NUM\_TRANSFERQUEUEMODE }
- enum spinTransferOperationModeEnums {
  - TransferOperationMode\_Continuous,

```

TransferOperationMode_MultiBlock,
NUM_TRANSFEROPERATIONMODE }
• enum spinTransferControlModeEnums {
TransferControlMode_Basic,
TransferControlMode_Automatic,
TransferControlMode_UserControlled,
NUM_TRANSFERCONTROLMODE }
• enum spinChunkGainSelectorEnums {
ChunkGainSelector_All,
ChunkGainSelector_Red,
ChunkGainSelector_Green,
ChunkGainSelector_Blue,
NUM_CHUNKGAINSELECTOR }
• enum spinChunkSelectorEnums {
ChunkSelector_Image,
ChunkSelector_CRC,
ChunkSelector_FrameID,
ChunkSelector_OffsetX,
ChunkSelector_OffsetY,
ChunkSelector_Width,
ChunkSelector_Height,
ChunkSelector_ExposureTime,
ChunkSelector_Gain,
ChunkSelector_BlackLevel,
ChunkSelector_PixelFormat,
ChunkSelector_Timestamp,
ChunkSelector_SequencerSetActive,
ChunkSelector_SerialData,
ChunkSelector_ExposureEndLineStatusAll,
NUM_CHUNKSELECTOR }
• enum spinChunkBlackLevelSelectorEnums {
ChunkBlackLevelSelector_All,
NUM_CHUNKBLACKLEVELSELECTOR }
• enum spinChunkPixelFormatEnums {
ChunkPixelFormat_Mono8,
ChunkPixelFormat_Mono12Packed,
ChunkPixelFormat_Mono16,
ChunkPixelFormat_RGB8Packed,
ChunkPixelFormat_YUV422Packed,
ChunkPixelFormat_BayerGR8,
ChunkPixelFormat_BayerRG8,
ChunkPixelFormat_BayerGB8,
ChunkPixelFormat_BayerBG8,
ChunkPixelFormat_YCbCr601_422_8_CbYCrY,
NUM_CHUNKPIXELFORMAT }
• enum spinFileOperationStatusEnums {
FileOperationStatus_Success,
FileOperationStatus_Failure,
FileOperationStatus_Overflow,
NUM_FILEOPERATIONSTATUS }
• enum spinFileOpenModeEnums {
FileOpenMode_Read,
FileOpenMode_Write,
FileOpenMode_ReadWrite,
NUM_FILEOPENMODE }
• enum spinFileOperationSelectorEnums {
FileOperationSelector_Open,
FileOperationSelector_Close,

```



```
FileOperationSelector_Read,  
FileOperationSelector_Write,  
FileOperationSelector_Delete,  
NUM_FILEOPERATIONSELECTOR }  
  
• enum spinFileSelectorEnums {  
FileSelector_UserSetDefault,  
FileSelector_UserSet0,  
FileSelector_UserSet1,  
FileSelector_UserFile1,  
FileSelector_SerialPort0,  
NUM_FILESELECTOR }  
  
• enum spinBinningSelectorEnums {  
BinningSelector_All,  
BinningSelector_Sensor,  
BinningSelector_ISP,  
NUM_BINNINGSELECTOR }  
  
• enum spinTestPatternGeneratorSelectorEnums {  
TestPatternGeneratorSelector_Sensor,  
TestPatternGeneratorSelector_PipelineStart,  
NUM_TESTPATTERNGENERATORSELECTOR }  
  
• enum spinTestPatternEnums {  
TestPattern_Off,  
TestPattern_Increment,  
TestPattern_SensorTestPattern,  
NUM_TESTPATTERN }  
  
• enum spinPixelColorFilterEnums {  
PixelColorFilter_None,  
PixelColorFilter_BayerRG,  
PixelColorFilter_BayerGB,  
PixelColorFilter_BayerGR,  
PixelColorFilter_BayerBG,  
NUM_PIXELCOLORFILTER }  
  
• enum spinAdcBitDepthEnums {  
AdcBitDepth_Bit8,  
AdcBitDepth_Bit10,  
AdcBitDepth_Bit12,  
AdcBitDepth_Bit14,  
NUM_ADCBITDEPTH }  
  
• enum spinDecimationHorizontalModeEnums {  
DecimationHorizontalMode_Discard,  
NUM_DECIMATIONHORIZONTALMODE }  
  
• enum spinBinningVerticalModeEnums {  
BinningVerticalMode_Sum,  
BinningVerticalMode_Average,  
NUM_BINNINGVERTICALMODE }  
  
• enum spinPixelSizeEnums {  
PixelSize_Bpp1,  
PixelSize_Bpp2,  
PixelSize_Bpp4,  
PixelSize_Bpp8,  
PixelSize_Bpp10,  
PixelSize_Bpp12,  
PixelSize_Bpp14,  
PixelSize_Bpp16,  
PixelSize_Bpp20,  
PixelSize_Bpp24,  
PixelSize_Bpp30,  
PixelSize_Bpp32,
```

```
PixelSize_Bpp36,  
PixelSize_Bpp48,  
PixelSize_Bpp64,  
PixelSize_Bpp96,  
NUM_PIXELSIZE }  
  
• enum spinDecimationSelectorEnums {  
    DecimationSelector_All,  
    DecimationSelector_Sensor,  
    NUM_DECIMATIONSELECTOR }  
  
• enum spinImageCompressionModeEnums {  
    ImageCompressionMode_Off,  
    ImageCompressionMode_Lossless,  
    NUM_IMAGECOMPRESSIONMODE }  
  
• enum spinBinningHorizontalModeEnums {  
    BinningHorizontalMode_Sum,  
    BinningHorizontalMode_Average,  
    NUM_BINNINGHORIZONTALMODE }  
  
• enum spinPixelFormatEnums {  
    PixelFormat_Mono8,  
    PixelFormat_Mono16,  
    PixelFormat_RGB8Packed,  
    PixelFormat_BayerGR8,  
    PixelFormat_BayerRG8,  
    PixelFormat_BayerGB8,  
    PixelFormat_BayerBG8,  
    PixelFormat_BayerGR16,  
    PixelFormat_BayerRG16,  
    PixelFormat_BayerGB16,  
    PixelFormat_BayerBG16,  
    PixelFormat_Mono12Packed,  
    PixelFormat_BayerGR12Packed,  
    PixelFormat_BayerRG12Packed,  
    PixelFormat_BayerGB12Packed,  
    PixelFormat_BayerBG12Packed,  
    PixelFormat_YUV411Packed,  
    PixelFormat_YUV422Packed,  
    PixelFormat_YUV444Packed,  
    PixelFormat_Mono12p,  
    PixelFormat_BayerGR12p,  
    PixelFormat_BayerRG12p,  
    PixelFormat_BayerGB12p,  
    PixelFormat_BayerBG12p,  
    PixelFormat_YCbCr8,  
    PixelFormat_YCbCr422_8,  
    PixelFormat_YCbCr411_8,  
    PixelFormat_BGR8,  
    PixelFormat_BGRa8,  
    PixelFormat_Mono10Packed,  
    PixelFormat_BayerGR10Packed,  
    PixelFormat_BayerRG10Packed,  
    PixelFormat_BayerGB10Packed,  
    PixelFormat_BayerBG10Packed,  
    PixelFormat_Mono10p,  
    PixelFormat_BayerGR10p,  
    PixelFormat_BayerRG10p,  
    PixelFormat_BayerGB10p,  
    PixelFormat_BayerBG10p,  
    PixelFormat_Mono1p,
```

[PixelFormat\\_Mono2p,](#)  
[PixelFormat\\_Mono4p,](#)  
[PixelFormat\\_Mono8s,](#)  
[PixelFormat\\_Mono10,](#)  
[PixelFormat\\_Mono12,](#)  
[PixelFormat\\_Mono14,](#)  
[PixelFormat\\_Mono16s,](#)  
[PixelFormat\\_Mono32f,](#)  
[PixelFormat\\_BayerBG10,](#)  
[PixelFormat\\_BayerBG12,](#)  
[PixelFormat\\_BayerGB10,](#)  
[PixelFormat\\_BayerGB12,](#)  
[PixelFormat\\_BayerGR10,](#)  
[PixelFormat\\_BayerGR12,](#)  
[PixelFormat\\_BayerRG10,](#)  
[PixelFormat\\_BayerRG12,](#)  
[PixelFormat\\_RGBa8,](#)  
[PixelFormat\\_RGBa10,](#)  
[PixelFormat\\_RGBa10p,](#)  
[PixelFormat\\_RGBa12,](#)  
[PixelFormat\\_RGBa12p,](#)  
[PixelFormat\\_RGBa14,](#)  
[PixelFormat\\_RGBa16,](#)  
[PixelFormat\\_RGB8,](#)  
[PixelFormat\\_RGB8\\_Planar,](#)  
[PixelFormat\\_RGB10,](#)  
[PixelFormat\\_RGB10\\_Planar,](#)  
[PixelFormat\\_RGB10p,](#)  
[PixelFormat\\_RGB10p32,](#)  
[PixelFormat\\_RGB12,](#)  
[PixelFormat\\_RGB12\\_Planar,](#)  
[PixelFormat\\_RGB12p,](#)  
[PixelFormat\\_RGB14,](#)  
[PixelFormat\\_RGB16,](#)  
[PixelFormat\\_RGB16s,](#)  
[PixelFormat\\_RGB32f,](#)  
[PixelFormat\\_RGB16\\_Planar,](#)  
[PixelFormat\\_RGB565p,](#)  
[PixelFormat\\_BGRa10,](#)  
[PixelFormat\\_BGRa10p,](#)  
[PixelFormat\\_BGRa12,](#)  
[PixelFormat\\_BGRa12p,](#)  
[PixelFormat\\_BGRa14,](#)  
[PixelFormat\\_BGRa16,](#)  
[PixelFormat\\_RGBa32f,](#)  
[PixelFormat\\_BGR10,](#)  
[PixelFormat\\_BGR10p,](#)  
[PixelFormat\\_BGR12,](#)  
[PixelFormat\\_BGR12p,](#)  
[PixelFormat\\_BGR14,](#)  
[PixelFormat\\_BGR16,](#)  
[PixelFormat\\_BGR565p,](#)  
[PixelFormat\\_R8,](#)  
[PixelFormat\\_R10,](#)  
[PixelFormat\\_R12,](#)  
[PixelFormat\\_R16,](#)  
[PixelFormat\\_G8,](#)  
[PixelFormat\\_G10,](#)

PixelFormat\_G12,  
PixelFormat\_G16,  
PixelFormat\_B8,  
PixelFormat\_B10,  
PixelFormat\_B12,  
PixelFormat\_B16,  
PixelFormat\_Coord3D\_ABC8,  
PixelFormat\_Coord3D\_ABC8\_Planar,  
PixelFormat\_Coord3D\_ABC10p,  
PixelFormat\_Coord3D\_ABC10p\_Planar,  
PixelFormat\_Coord3D\_ABC12p,  
PixelFormat\_Coord3D\_ABC12p\_Planar,  
PixelFormat\_Coord3D\_ABC16,  
PixelFormat\_Coord3D\_ABC16\_Planar,  
PixelFormat\_Coord3D\_ABC32f,  
PixelFormat\_Coord3D\_ABC32f\_Planar,  
PixelFormat\_Coord3D\_AC8,  
PixelFormat\_Coord3D\_AC8\_Planar,  
PixelFormat\_Coord3D\_AC10p,  
PixelFormat\_Coord3D\_AC10p\_Planar,  
PixelFormat\_Coord3D\_AC12p,  
PixelFormat\_Coord3D\_AC12p\_Planar,  
PixelFormat\_Coord3D\_AC16,  
PixelFormat\_Coord3D\_AC16\_Planar,  
PixelFormat\_Coord3D\_AC32f,  
PixelFormat\_Coord3D\_AC32f\_Planar,  
PixelFormat\_Coord3D\_A8,  
PixelFormat\_Coord3D\_A10p,  
PixelFormat\_Coord3D\_A12p,  
PixelFormat\_Coord3D\_A16,  
PixelFormat\_Coord3D\_A32f,  
PixelFormat\_Coord3D\_B8,  
PixelFormat\_Coord3D\_B10p,  
PixelFormat\_Coord3D\_B12p,  
PixelFormat\_Coord3D\_B16,  
PixelFormat\_Coord3D\_B32f,  
PixelFormat\_Coord3D\_C8,  
PixelFormat\_Coord3D\_C10p,  
PixelFormat\_Coord3D\_C12p,  
PixelFormat\_Coord3D\_C16,  
PixelFormat\_Coord3D\_C32f,  
PixelFormat\_Confidence1,  
PixelFormat\_Confidence1p,  
PixelFormat\_Confidence8,  
PixelFormat\_Confidence16,  
PixelFormat\_Confidence32f,  
PixelFormat\_BiColorBGRG8,  
PixelFormat\_BiColorBGRG10,  
PixelFormat\_BiColorBGRG10p,  
PixelFormat\_BiColorBGRG12,  
PixelFormat\_BiColorBGRG12p,  
PixelFormat\_BiColorRGBG8,  
PixelFormat\_BiColorRGBG10,  
PixelFormat\_BiColorRGBG10p,  
PixelFormat\_BiColorRGBG12,  
PixelFormat\_BiColorRGBG12p,  
PixelFormat\_SCF1WBWG8,  
PixelFormat\_SCF1WBWG10,

PixelFormat\_SCF1WBWG10p,  
PixelFormat\_SCF1WBWG12,  
PixelFormat\_SCF1WBWG12p,  
PixelFormat\_SCF1WBWG14,  
PixelFormat\_SCF1WBWG16,  
PixelFormat\_SCF1WGWB8,  
PixelFormat\_SCF1WGWB10,  
PixelFormat\_SCF1WGWB10p,  
PixelFormat\_SCF1WGWB12,  
PixelFormat\_SCF1WGWB12p,  
PixelFormat\_SCF1WGWB14,  
PixelFormat\_SCF1WGWB16,  
PixelFormat\_SCF1WGWR8,  
PixelFormat\_SCF1WGWR10,  
PixelFormat\_SCF1WGWR10p,  
PixelFormat\_SCF1WGWR12,  
PixelFormat\_SCF1WGWR12p,  
PixelFormat\_SCF1WGWR14,  
PixelFormat\_SCF1WGWR16,  
PixelFormat\_SCF1WRWG8,  
PixelFormat\_SCF1WRWG10,  
PixelFormat\_SCF1WRWG10p,  
PixelFormat\_SCF1WRWG12,  
PixelFormat\_SCF1WRWG12p,  
PixelFormat\_SCF1WRWG14,  
PixelFormat\_SCF1WRWG16,  
PixelFormat\_YCbCr8\_CbYCr,  
PixelFormat\_YCbCr10\_CbYCr,  
PixelFormat\_YCbCr10p\_CbYCr,  
PixelFormat\_YCbCr12\_CbYCr,  
PixelFormat\_YCbCr12p\_CbYCr,  
PixelFormat\_YCbCr411\_8\_CbYYCrYY,  
PixelFormat\_YCbCr422\_8\_CbYCrY,  
PixelFormat\_YCbCr422\_10,  
PixelFormat\_YCbCr422\_10\_CbYCrY,  
PixelFormat\_YCbCr422\_10p,  
PixelFormat\_YCbCr422\_10p\_CbYCrY,  
PixelFormat\_YCbCr422\_12,  
PixelFormat\_YCbCr422\_12\_CbYCrY,  
PixelFormat\_YCbCr422\_12p,  
PixelFormat\_YCbCr422\_12p\_CbYCrY,  
PixelFormat\_YCbCr601\_8\_CbYCr,  
PixelFormat\_YCbCr601\_10\_CbYCr,  
PixelFormat\_YCbCr601\_10p\_CbYCr,  
PixelFormat\_YCbCr601\_12\_CbYCr,  
PixelFormat\_YCbCr601\_12p\_CbYCr,  
PixelFormat\_YCbCr601\_411\_8\_CbYYCrYY,  
PixelFormat\_YCbCr601\_422\_8,  
PixelFormat\_YCbCr601\_422\_8\_CbYCrY,  
PixelFormat\_YCbCr601\_422\_10,  
PixelFormat\_YCbCr601\_422\_10\_CbYCrY,  
PixelFormat\_YCbCr601\_422\_10p,  
PixelFormat\_YCbCr601\_422\_10p\_CbYCrY,  
PixelFormat\_YCbCr601\_422\_12,  
PixelFormat\_YCbCr601\_422\_12\_CbYCrY,  
PixelFormat\_YCbCr601\_422\_12p,  
PixelFormat\_YCbCr601\_422\_12p\_CbYCrY,  
PixelFormat\_YCbCr709\_8\_CbYCr,

```

PixelFormat_YCbCr709_10_CbYCr,
PixelFormat_YCbCr709_10p_CbYCr,
PixelFormat_YCbCr709_12_CbYCr,
PixelFormat_YCbCr709_12p_CbYCr,
PixelFormat_YCbCr709_411_8_CbYYCrYY,
PixelFormat_YCbCr709_422_8,
PixelFormat_YCbCr709_422_8_CbYCrY,
PixelFormat_YCbCr709_422_10,
PixelFormat_YCbCr709_422_10_CbYCrY,
PixelFormat_YCbCr709_422_10p,
PixelFormat_YCbCr709_422_10p_CbYCrY,
PixelFormat_YCbCr709_422_12,
PixelFormat_YCbCr709_422_12_CbYCrY,
PixelFormat_YCbCr709_422_12p,
PixelFormat_YCbCr709_422_12p_CbYCrY,
PixelFormat_YUV8_UYV,
PixelFormat_YUV411_8_UYYVYY,
PixelFormat_YUV422_8,
PixelFormat_YUV422_8_UYVY,
PixelFormat_Polarized8,
PixelFormat_Polarized10p,
PixelFormat_Polarized12p,
PixelFormat_Polarized16,
PixelFormat_BayerRGPolarized8,
PixelFormat_BayerRGPolarized10p,
PixelFormat_BayerRGPolarized12p,
PixelFormat_BayerRGPolarized16,
PixelFormat_LLCMono8,
PixelFormat_LLCBayerRG8,
PixelFormat_JPEGMono8,
PixelFormat_JPEGColor8,
PixelFormat_Raw16,
PixelFormat_Raw8,
PixelFormat_R12_Jpeg,
PixelFormat_GR12_Jpeg,
PixelFormat_GB12_Jpeg,
PixelFormat_B12_Jpeg,
UNKNOWN_PIXELFORMAT,
NUM_PIXELFORMAT }

• enum spinDecimationVerticalModeEnums {
    DecimationVerticalMode_Discard,
    NUM_DECIMATIONVERTICALMODE }

• enum spinLineModeEnums {
    LineMode_Input,
    LineMode_Output,
    NUM_LINEMODE }

• enum spinLineSourceEnums {
    LineSource_Off,
    LineSource_Line0,
    LineSource_Line1,
    LineSource_Line2,
    LineSource_Line3,
    LineSource_UserOutput0,
    LineSource_UserOutput1,
    LineSource_UserOutput2,
    LineSource_UserOutput3,
    LineSource_Counter0Active,
    LineSource_Counter1Active,

```

```

LineSource_LogicBlock0,
LineSource_LogicBlock1,
LineSource_ExposureActive,
LineSource_FrameTriggerWait,
LineSource_SerialPort0,
LineSource_PPSSignal,
LineSource_AllPixel,
LineSource_AnyPixel,
NUM_LINESOURCE }

• enum spinLineInputFilterSelectorEnums {
    LineInputFilterSelector_Deglintch,
    LineInputFilterSelector_Debounce,
    NUM_LINEINPUTFILTERSELECTOR }

• enum spinUserOutputSelectorEnums {
    UserOutputSelector_UserOutput0,
    UserOutputSelector_UserOutput1,
    UserOutputSelector_UserOutput2,
    UserOutputSelector_UserOutput3,
    NUM_USEROUTPUTSELECTOR }

• enum spinLineFormatEnums {
    LineFormat_NoConnect,
    LineFormat_TriState,
    LineFormat_TTL,
    LineFormat_LVDS,
    LineFormat_RS422,
    LineFormat_OptoCoupled,
    LineFormat_OpenDrain,
    NUM_LINEFORMAT }

• enum spinLineSelectorEnums {
    LineSelector_Line0,
    LineSelector_Line1,
    LineSelector_Line2,
    LineSelector_Line3,
    NUM_LINESELECTOR }

• enum spinExposureActiveModeEnums {
    ExposureActiveMode_Line1,
    ExposureActiveMode_AnyPixels,
    ExposureActiveMode_AllPixels,
    NUM_EXPOSUREACTIVEMODE }

• enum spinCounterTriggerActivationEnums {
    CounterTriggerActivation_LevelLow,
    CounterTriggerActivation_LevelHigh,
    CounterTriggerActivation_FallingEdge,
    CounterTriggerActivation_RisingEdge,
    CounterTriggerActivation_AnyEdge,
    NUM_COUNTERTRIGGERACTIVATION }

• enum spinCounterSelectorEnums {
    CounterSelector_Counter0,
    CounterSelector_Counter1,
    NUM_COUNTERSELECTOR }

• enum spinCounterStatusEnums {
    CounterStatus_CounterIdle,
    CounterStatus_CounterTriggerWait,
    CounterStatus_CounterActive,
    CounterStatus_CounterCompleted,
    CounterStatus_CounterOverflow,
    NUM_COUNTERSTATUS }

```

- `enum spinCounterTriggerSourceEnums {`  
    `CounterTriggerSource_Off,`  
    `CounterTriggerSource_Line0,`  
    `CounterTriggerSource_Line1,`  
    `CounterTriggerSource_Line2,`  
    `CounterTriggerSource_Line3,`  
    `CounterTriggerSource_UserOutput0,`  
    `CounterTriggerSource_UserOutput1,`  
    `CounterTriggerSource_UserOutput2,`  
    `CounterTriggerSource_UserOutput3,`  
    `CounterTriggerSource_Counter0Start,`  
    `CounterTriggerSource_Counter1Start,`  
    `CounterTriggerSource_Counter0End,`  
    `CounterTriggerSource_Counter1End,`  
    `CounterTriggerSource_LogicBlock0,`  
    `CounterTriggerSource_LogicBlock1,`  
    `CounterTriggerSource_ExposureStart,`  
    `CounterTriggerSource_ExposureEnd,`  
    `CounterTriggerSource_FrameTriggerWait,`  
    `NUM_COUNTERTRIGGERSOURCE }`
- `enum spinCounterResetSourceEnums {`  
    `CounterResetSource_Off,`  
    `CounterResetSource_Line0,`  
    `CounterResetSource_Line1,`  
    `CounterResetSource_Line2,`  
    `CounterResetSource_Line3,`  
    `CounterResetSource_UserOutput0,`  
    `CounterResetSource_UserOutput1,`  
    `CounterResetSource_UserOutput2,`  
    `CounterResetSource_UserOutput3,`  
    `CounterResetSource_Counter0Start,`  
    `CounterResetSource_Counter1Start,`  
    `CounterResetSource_Counter0End,`  
    `CounterResetSource_Counter1End,`  
    `CounterResetSource_LogicBlock0,`  
    `CounterResetSource_LogicBlock1,`  
    `CounterResetSource_ExposureStart,`  
    `CounterResetSource_ExposureEnd,`  
    `CounterResetSource_FrameTriggerWait,`  
    `NUM_COUNTERRESETSOURCE }`
- `enum spinCounterEventSourceEnums {`  
    `CounterEventSource_Off,`  
    `CounterEventSource_MHzTick,`  
    `CounterEventSource_Line0,`  
    `CounterEventSource_Line1,`  
    `CounterEventSource_Line2,`  
    `CounterEventSource_Line3,`  
    `CounterEventSource_UserOutput0,`  
    `CounterEventSource_UserOutput1,`  
    `CounterEventSource_UserOutput2,`  
    `CounterEventSource_UserOutput3,`  
    `CounterEventSource_Counter0Start,`  
    `CounterEventSource_Counter1Start,`  
    `CounterEventSource_Counter0End,`  
    `CounterEventSource_Counter1End,`  
    `CounterEventSource_LogicBlock0,`  
    `CounterEventSource_LogicBlock1,`  
    `CounterEventSource_ExposureStart,`



```

CounterEventSource_ExposureEnd,
CounterEventSource_FrameTriggerWait,
NUM_COUNTEREVENTSOURCE }
• enum spinCounterEventActivationEnums {
CounterEventActivation_LevelLow,
CounterEventActivation_LevelHigh,
CounterEventActivation_FallingEdge,
CounterEventActivation_RisingEdge,
CounterEventActivation_AnyEdge,
NUM_COUNTEREVENTACTIVATION }
• enum spinCounterResetActivationEnums {
CounterResetActivation_LevelLow,
CounterResetActivation_LevelHigh,
CounterResetActivation_FallingEdge,
CounterResetActivation_RisingEdge,
CounterResetActivation_AnyEdge,
NUM_COUNTERRESETACTIVATION }
• enum spinDeviceTypeEnums {
DeviceType_Transmitter,
DeviceType_Receiver,
DeviceType_Transceiver,
DeviceType_Peripheral,
NUM_DEVICETYPE }
• enum spinDeviceConnectionStatusEnums {
DeviceConnectionStatus_Active,
DeviceConnectionStatus_Inactive,
NUM_DEVICECONNECTIONSTATUS }
• enum spinDeviceLinkThroughputLimitModeEnums {
DeviceLinkThroughputLimitMode_On,
DeviceLinkThroughputLimitMode_Off,
NUM_DEVICELINKTHROUGHPUTLIMITMODE }
• enum spinDeviceLinkHeartbeatModeEnums {
DeviceLinkHeartbeatMode_On,
DeviceLinkHeartbeatMode_Off,
NUM_DEVICELINKHEARTBEATMODE }
• enum spinDeviceStreamChannelTypeEnums {
DeviceStreamChannelType_Transmitter,
DeviceStreamChannelType_Receiver,
NUM_DEVICESTREAMCHANNELTYPE }
• enum spinDeviceStreamChannelEndiannessEnums {
DeviceStreamChannelEndianness_Big,
DeviceStreamChannelEndianness_Little,
NUM_DEVICESTREAMCHANNELENDIANNESS }
• enum spinDeviceClockSelectorEnums {
DeviceClockSelector_Sensor,
DeviceClockSelector_SensorDigitization,
DeviceClockSelector_CameraLink,
NUM_DEVICECLOCKSELECTOR }
• enum spinDeviceSerialPortSelectorEnums {
DeviceSerialPortSelector_CameraLink,
NUM_DEVICESERIALPORTSELECTOR }
• enum spinDeviceSerialPortBaudRateEnums {
DeviceSerialPortBaudRate_Baud9600,
DeviceSerialPortBaudRate_Baud19200,
DeviceSerialPortBaudRate_Baud38400,
DeviceSerialPortBaudRate_Baud57600,
DeviceSerialPortBaudRate_Baud115200,
DeviceSerialPortBaudRate_Baud230400,

```

```

DeviceSerialPortBaudRate_Baud460800,
DeviceSerialPortBaudRate_Baud921600,
NUM_DEVICESERIALPORTBAUDRATE }

• enum spinSensorTapsEnums {
    SensorTaps_One,
    SensorTaps_Two,
    SensorTaps_Three,
    SensorTaps_Four,
    SensorTaps_Eight,
    SensorTaps_Ten,
    NUM_SENSORTAPS }

• enum spinSensorDigitizationTapsEnums {
    SensorDigitizationTaps_One,
    SensorDigitizationTaps_Two,
    SensorDigitizationTaps_Three,
    SensorDigitizationTaps_Four,
    SensorDigitizationTaps_Eight,
    SensorDigitizationTaps_Ten,
    NUM_SENSORDIGITIZATIONTAPS }

• enum spinRegionSelectorEnums {
    RegionSelector_Region0,
    RegionSelector_Region1,
    RegionSelector_Region2,
    RegionSelector_All,
    NUM_REGIONSELECTOR }

• enum spinRegionModeEnums {
    RegionMode_Off,
    RegionMode_On,
    NUM_REGIONMODE }

• enum spinRegionDestinationEnums {
    RegionDestination_Stream0,
    RegionDestination_Stream1,
    RegionDestination_Stream2,
    NUM_REGIONDESTINATION }

• enum spinImageComponentSelectorEnums {
    ImageComponentSelector_Intensity,
    ImageComponentSelector_Color,
    ImageComponentSelector_Infrared,
    ImageComponentSelector_Ultraviolet,
    ImageComponentSelector_Range,
    ImageComponentSelector_Disparity,
    ImageComponentSelector_Confidence,
    ImageComponentSelector_Scatter,
    NUM_IMAGECOMPONENTSELECTOR }

• enum spinPixelFormatInfoSelectorEnums {
    PixelFormatInfoSelector_Mono1p,
    PixelFormatInfoSelector_Mono2p,
    PixelFormatInfoSelector_Mono4p,
    PixelFormatInfoSelector_Mono8,
    PixelFormatInfoSelector_Mono8s,
    PixelFormatInfoSelector_Mono10,
    PixelFormatInfoSelector_Mono10p,
    PixelFormatInfoSelector_Mono12,
    PixelFormatInfoSelector_Mono12p,
    PixelFormatInfoSelector_Mono14,
    PixelFormatInfoSelector_Mono16,
    PixelFormatInfoSelector_Mono16s,
    PixelFormatInfoSelector_Mono32f,

```

[PixelFormatInfoSelector\\_BayerBG8,](#)  
[PixelFormatInfoSelector\\_BayerBG10,](#)  
[PixelFormatInfoSelector\\_BayerBG10p,](#)  
[PixelFormatInfoSelector\\_BayerBG12,](#)  
[PixelFormatInfoSelector\\_BayerBG12p,](#)  
[PixelFormatInfoSelector\\_BayerBG16,](#)  
[PixelFormatInfoSelector\\_BayerGB8,](#)  
[PixelFormatInfoSelector\\_BayerGB10,](#)  
[PixelFormatInfoSelector\\_BayerGB10p,](#)  
[PixelFormatInfoSelector\\_BayerGB12,](#)  
[PixelFormatInfoSelector\\_BayerGB12p,](#)  
[PixelFormatInfoSelector\\_BayerGB16,](#)  
[PixelFormatInfoSelector\\_BayerGR8,](#)  
[PixelFormatInfoSelector\\_BayerGR10,](#)  
[PixelFormatInfoSelector\\_BayerGR10p,](#)  
[PixelFormatInfoSelector\\_BayerGR12,](#)  
[PixelFormatInfoSelector\\_BayerGR12p,](#)  
[PixelFormatInfoSelector\\_BayerGR16,](#)  
[PixelFormatInfoSelector\\_BayerRG8,](#)  
[PixelFormatInfoSelector\\_BayerRG10,](#)  
[PixelFormatInfoSelector\\_BayerRG10p,](#)  
[PixelFormatInfoSelector\\_BayerRG12,](#)  
[PixelFormatInfoSelector\\_BayerRG12p,](#)  
[PixelFormatInfoSelector\\_BayerRG16,](#)  
[PixelFormatInfoSelector\\_RGBa8,](#)  
[PixelFormatInfoSelector\\_RGBa10,](#)  
[PixelFormatInfoSelector\\_RGBa10p,](#)  
[PixelFormatInfoSelector\\_RGBa12,](#)  
[PixelFormatInfoSelector\\_RGBa12p,](#)  
[PixelFormatInfoSelector\\_RGBa14,](#)  
[PixelFormatInfoSelector\\_RGBa16,](#)  
[PixelFormatInfoSelector\\_RGB8,](#)  
[PixelFormatInfoSelector\\_RGB8\\_Planar,](#)  
[PixelFormatInfoSelector\\_RGB10,](#)  
[PixelFormatInfoSelector\\_RGB10\\_Planar,](#)  
[PixelFormatInfoSelector\\_RGB10p,](#)  
[PixelFormatInfoSelector\\_RGB10p32,](#)  
[PixelFormatInfoSelector\\_RGB12,](#)  
[PixelFormatInfoSelector\\_RGB12\\_Planar,](#)  
[PixelFormatInfoSelector\\_RGB12p,](#)  
[PixelFormatInfoSelector\\_RGB14,](#)  
[PixelFormatInfoSelector\\_RGB16,](#)  
[PixelFormatInfoSelector\\_RGB16s,](#)  
[PixelFormatInfoSelector\\_RGB32f,](#)  
[PixelFormatInfoSelector\\_RGB16\\_Planar,](#)  
[PixelFormatInfoSelector\\_RGB565p,](#)  
[PixelFormatInfoSelector\\_BGRa8,](#)  
[PixelFormatInfoSelector\\_BGRa10,](#)  
[PixelFormatInfoSelector\\_BGRa10p,](#)  
[PixelFormatInfoSelector\\_BGRa12,](#)  
[PixelFormatInfoSelector\\_BGRa12p,](#)  
[PixelFormatInfoSelector\\_BGRa14,](#)  
[PixelFormatInfoSelector\\_BGRa16,](#)  
[PixelFormatInfoSelector\\_RGBa32f,](#)  
[PixelFormatInfoSelector\\_BGR8,](#)  
[PixelFormatInfoSelector\\_BGR10,](#)  
[PixelFormatInfoSelector\\_BGR10p,](#)  
[PixelFormatInfoSelector\\_BGR12,](#)

PixelFormatInfoSelector\_BGR12p,  
PixelFormatInfoSelector\_BGR14,  
PixelFormatInfoSelector\_BGR16,  
PixelFormatInfoSelector\_BGR565p,  
PixelFormatInfoSelector\_R8,  
PixelFormatInfoSelector\_R10,  
PixelFormatInfoSelector\_R12,  
PixelFormatInfoSelector\_R16,  
PixelFormatInfoSelector\_G8,  
PixelFormatInfoSelector\_G10,  
PixelFormatInfoSelector\_G12,  
PixelFormatInfoSelector\_G16,  
PixelFormatInfoSelector\_B8,  
PixelFormatInfoSelector\_B10,  
PixelFormatInfoSelector\_B12,  
PixelFormatInfoSelector\_B16,  
PixelFormatInfoSelector\_Coord3D\_ABC8,  
PixelFormatInfoSelector\_Coord3D\_ABC8\_Planar,  
PixelFormatInfoSelector\_Coord3D\_ABC10p,  
PixelFormatInfoSelector\_Coord3D\_ABC10p\_Planar,  
PixelFormatInfoSelector\_Coord3D\_ABC12p,  
PixelFormatInfoSelector\_Coord3D\_ABC12p\_Planar,  
PixelFormatInfoSelector\_Coord3D\_ABC16,  
PixelFormatInfoSelector\_Coord3D\_ABC16\_Planar,  
PixelFormatInfoSelector\_Coord3D\_ABC32f,  
PixelFormatInfoSelector\_Coord3D\_ABC32f\_Planar,  
PixelFormatInfoSelector\_Coord3D\_AC8,  
PixelFormatInfoSelector\_Coord3D\_AC8\_Planar,  
PixelFormatInfoSelector\_Coord3D\_AC10p,  
PixelFormatInfoSelector\_Coord3D\_AC10p\_Planar,  
PixelFormatInfoSelector\_Coord3D\_AC12p,  
PixelFormatInfoSelector\_Coord3D\_AC12p\_Planar,  
PixelFormatInfoSelector\_Coord3D\_AC16,  
PixelFormatInfoSelector\_Coord3D\_AC16\_Planar,  
PixelFormatInfoSelector\_Coord3D\_AC32f,  
PixelFormatInfoSelector\_Coord3D\_AC32f\_Planar,  
PixelFormatInfoSelector\_Coord3D\_A8,  
PixelFormatInfoSelector\_Coord3D\_A10p,  
PixelFormatInfoSelector\_Coord3D\_A12p,  
PixelFormatInfoSelector\_Coord3D\_A16,  
PixelFormatInfoSelector\_Coord3D\_A32f,  
PixelFormatInfoSelector\_Coord3D\_B8,  
PixelFormatInfoSelector\_Coord3D\_B10p,  
PixelFormatInfoSelector\_Coord3D\_B12p,  
PixelFormatInfoSelector\_Coord3D\_B16,  
PixelFormatInfoSelector\_Coord3D\_B32f,  
PixelFormatInfoSelector\_Coord3D\_C8,  
PixelFormatInfoSelector\_Coord3D\_C10p,  
PixelFormatInfoSelector\_Coord3D\_C12p,  
PixelFormatInfoSelector\_Coord3D\_C16,  
PixelFormatInfoSelector\_Coord3D\_C32f,  
PixelFormatInfoSelector\_Confidence1,  
PixelFormatInfoSelector\_Confidence1p,  
PixelFormatInfoSelector\_Confidence8,  
PixelFormatInfoSelector\_Confidence16,  
PixelFormatInfoSelector\_Confidence32f,  
PixelFormatInfoSelector\_BiColorBGRG8,  
PixelFormatInfoSelector\_BiColorBGRG10,

PixelFormatInfoSelector\_BiColorBGRG10p,  
PixelFormatInfoSelector\_BiColorBGRG12,  
PixelFormatInfoSelector\_BiColorBGRG12p,  
PixelFormatInfoSelector\_BiColorRGBG8,  
PixelFormatInfoSelector\_BiColorRGBG10,  
PixelFormatInfoSelector\_BiColorRGBG10p,  
PixelFormatInfoSelector\_BiColorRGBG12,  
PixelFormatInfoSelector\_BiColorRGBG12p,  
PixelFormatInfoSelector\_SCF1WBWG8,  
PixelFormatInfoSelector\_SCF1WBWG10,  
PixelFormatInfoSelector\_SCF1WBWG10p,  
PixelFormatInfoSelector\_SCF1WBWG12,  
PixelFormatInfoSelector\_SCF1WBWG12p,  
PixelFormatInfoSelector\_SCF1WBWG14,  
PixelFormatInfoSelector\_SCF1WBWG16,  
PixelFormatInfoSelector\_SCF1WGWB8,  
PixelFormatInfoSelector\_SCF1WGWB10,  
PixelFormatInfoSelector\_SCF1WGWB10p,  
PixelFormatInfoSelector\_SCF1WGWB12,  
PixelFormatInfoSelector\_SCF1WGWB12p,  
PixelFormatInfoSelector\_SCF1WGWB14,  
PixelFormatInfoSelector\_SCF1WGWB16,  
PixelFormatInfoSelector\_SCF1WGWR8,  
PixelFormatInfoSelector\_SCF1WGWR10,  
PixelFormatInfoSelector\_SCF1WGWR10p,  
PixelFormatInfoSelector\_SCF1WGWR12,  
PixelFormatInfoSelector\_SCF1WGWR12p,  
PixelFormatInfoSelector\_SCF1WGWR14,  
PixelFormatInfoSelector\_SCF1WGWR16,  
PixelFormatInfoSelector\_SCF1WRWG8,  
PixelFormatInfoSelector\_SCF1WRWG10,  
PixelFormatInfoSelector\_SCF1WRWG10p,  
PixelFormatInfoSelector\_SCF1WRWG12,  
PixelFormatInfoSelector\_SCF1WRWG12p,  
PixelFormatInfoSelector\_SCF1WRWG14,  
PixelFormatInfoSelector\_SCF1WRWG16,  
PixelFormatInfoSelector\_YCbCr8,  
PixelFormatInfoSelector\_YCbCr8\_CbYCr,  
PixelFormatInfoSelector\_YCbCr10\_CbYCr,  
PixelFormatInfoSelector\_YCbCr10p\_CbYCr,  
PixelFormatInfoSelector\_YCbCr12\_CbYCr,  
PixelFormatInfoSelector\_YCbCr12p\_CbYCr,  
PixelFormatInfoSelector\_YCbCr411\_8,  
PixelFormatInfoSelector\_YCbCr411\_8\_CbYYCrYY,  
PixelFormatInfoSelector\_YCbCr422\_8,  
PixelFormatInfoSelector\_YCbCr422\_8\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr422\_10,  
PixelFormatInfoSelector\_YCbCr422\_10\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr422\_10p,  
PixelFormatInfoSelector\_YCbCr422\_10p\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr422\_12,  
PixelFormatInfoSelector\_YCbCr422\_12\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr422\_12p,  
PixelFormatInfoSelector\_YCbCr422\_12p\_CbYCrY,  
PixelFormatInfoSelector\_YCbCr601\_8\_CbYCr,  
PixelFormatInfoSelector\_YCbCr601\_10\_CbYCr,  
PixelFormatInfoSelector\_YCbCr601\_10p\_CbYCr,  
PixelFormatInfoSelector\_YCbCr601\_12\_CbYCr,

```

PixelFormatInfoSelector_YCbCr601_12p_CbYCr,
PixelFormatInfoSelector_YCbCr601_411_8_CbYYCrYY,
PixelFormatInfoSelector_YCbCr601_422_8,
PixelFormatInfoSelector_YCbCr601_422_8_CbYCrY,
PixelFormatInfoSelector_YCbCr601_422_10,
PixelFormatInfoSelector_YCbCr601_422_10_CbYCrY,
PixelFormatInfoSelector_YCbCr601_422_10p,
PixelFormatInfoSelector_YCbCr601_422_10p_CbYCrY,
PixelFormatInfoSelector_YCbCr601_422_12,
PixelFormatInfoSelector_YCbCr601_422_12_CbYCrY,
PixelFormatInfoSelector_YCbCr601_422_12p,
PixelFormatInfoSelector_YCbCr601_422_12p_CbYCrY,
PixelFormatInfoSelector_YCbCr709_8_CbYCr,
PixelFormatInfoSelector_YCbCr709_10_CbYCr,
PixelFormatInfoSelector_YCbCr709_10p_CbYCr,
PixelFormatInfoSelector_YCbCr709_12_CbYCr,
PixelFormatInfoSelector_YCbCr709_12p_CbYCr,
PixelFormatInfoSelector_YCbCr709_411_8_CbYYCrYY,
PixelFormatInfoSelector_YCbCr709_422_8,
PixelFormatInfoSelector_YCbCr709_422_8_CbYCrY,
PixelFormatInfoSelector_YCbCr709_422_10,
PixelFormatInfoSelector_YCbCr709_422_10_CbYCrY,
PixelFormatInfoSelector_YCbCr709_422_10p,
PixelFormatInfoSelector_YCbCr709_422_10p_CbYCrY,
PixelFormatInfoSelector_YCbCr709_422_12,
PixelFormatInfoSelector_YCbCr709_422_12_CbYCrY,
PixelFormatInfoSelector_YCbCr709_422_12p,
PixelFormatInfoSelector_YCbCr709_422_12p_CbYCrY,
PixelFormatInfoSelector_YUV8_UYV,
PixelFormatInfoSelector_YUV411_8_UYYVYY,
PixelFormatInfoSelector_YUV422_8,
PixelFormatInfoSelector_YUV422_8_UYVY,
PixelFormatInfoSelector_Polarized8,
PixelFormatInfoSelector_Polarized10p,
PixelFormatInfoSelector_Polarized12p,
PixelFormatInfoSelector_Polarized16,
PixelFormatInfoSelector_BayerRGPolarized8,
PixelFormatInfoSelector_BayerRGPolarized10p,
PixelFormatInfoSelector_BayerRGPolarized12p,
PixelFormatInfoSelector_BayerRGPolarized16,
PixelFormatInfoSelector_LLCMono8,
PixelFormatInfoSelector_LLCBayerRG8,
PixelFormatInfoSelector_JPEGMono8,
PixelFormatInfoSelector_JPEGColor8,
NUM_PIXELFORMATINFOSELECTOR }

• enum spinDeinterlacingEnums {
    Deinterlacing_Off,
    Deinterlacing_LineDuplication,
    Deinterlacing_Weave,
    NUM_DEINTERLACING }

• enum spinImageCompressionRateOptionEnums {
    ImageCompressionRateOption_FixBitrate,
    ImageCompressionRateOption_FixQuality,
    NUM_IMAGECOMPRESSIONRATEOPTION }

• enum spinImageCompressionJPEGFormatOptionEnums {
    ImageCompressionJPEGFormatOption_Lossless,
    ImageCompressionJPEGFormatOption_BaselineStandard,
    ImageCompressionJPEGFormatOption_BaselineOptimized,

```

```

ImageCompressionJPEGFormatOption_Progressive,
NUM_IMAGECOMPRESSIONJPEGFORMATOPTION }
• enum spinAcquisitionStatusSelectorEnums {
  AcquisitionStatusSelector_AcquisitionTriggerWait,
  AcquisitionStatusSelector_AcquisitionActive,
  AcquisitionStatusSelector_AcquisitionTransfer,
  AcquisitionStatusSelector_FrameTriggerWait,
  AcquisitionStatusSelector_FrameActive,
  AcquisitionStatusSelector_ExposureActive,
  NUM_ACQUISITIONSTATUSSELECTOR }
• enum spinExposureTimeModeEnums {
  ExposureTimeMode_Common,
  ExposureTimeMode_Individual,
  NUM_EXPOSURETIMEMODE }
• enum spinExposureTimeSelectorEnums {
  ExposureTimeSelector_Common,
  ExposureTimeSelector_Red,
  ExposureTimeSelector_Green,
  ExposureTimeSelector_Blue,
  ExposureTimeSelector_Cyan,
  ExposureTimeSelector_Magenta,
  ExposureTimeSelector_Yellow,
  ExposureTimeSelector_Infrared,
  ExposureTimeSelector_Ultraviolet,
  ExposureTimeSelector_Stage1,
  ExposureTimeSelector_Stage2,
  NUM_EXPOSURETIMESELECTOR }
• enum spinGainAutoBalanceEnums {
  GainAutoBalance_Off,
  GainAutoBalance_Once,
  GainAutoBalance_Continuous,
  NUM_GAINAUTOBALANCE }
• enum spinBlackLevelAutoEnums {
  BlackLevelAuto_Off,
  BlackLevelAuto_Once,
  BlackLevelAuto_Continuous,
  NUM_BLACKLEVELAUTO }
• enum spinBlackLevelAutoBalanceEnums {
  BlackLevelAutoBalance_Off,
  BlackLevelAutoBalance_Once,
  BlackLevelAutoBalance_Continuous,
  NUM_BLACKLEVELAUTOBALANCE }
• enum spinWhiteClipSelectorEnums {
  WhiteClipSelector_All,
  WhiteClipSelector_Red,
  WhiteClipSelector_Green,
  WhiteClipSelector_Blue,
  WhiteClipSelector_Y,
  WhiteClipSelector_U,
  WhiteClipSelector_V,
  WhiteClipSelector_Tap1,
  WhiteClipSelector_Tap2,
  NUM_WHITECLIPSELECTOR }
• enum spinTimerSelectorEnums {
  TimerSelector_Timer0,
  TimerSelector_Timer1,
  TimerSelector_Timer2,
  NUM_TIMERSELECTOR }

```

- `enum spinTimerStatusEnums {`  
    [TimerStatus\\_TimerIdle](#),  
    [TimerStatus\\_TimerTriggerWait](#),  
    [TimerStatus\\_TimerActive](#),  
    [TimerStatus\\_TimerCompleted](#),  
    [NUM\\_TIMERSTATUS](#) }  
• `enum spinTimerTriggerSourceEnums {`  
    [TimerTriggerSource\\_Off](#),  
    [TimerTriggerSource\\_AcquisitionTrigger](#),  
    [TimerTriggerSource\\_AcquisitionStart](#),  
    [TimerTriggerSource\\_AcquisitionEnd](#),  
    [TimerTriggerSource\\_FrameTrigger](#),  
    [TimerTriggerSource\\_FrameStart](#),  
    [TimerTriggerSource\\_FrameEnd](#),  
    [TimerTriggerSource\\_FrameBurstStart](#),  
    [TimerTriggerSource\\_FrameBurstEnd](#),  
    [TimerTriggerSource\\_LineTrigger](#),  
    [TimerTriggerSource\\_LineStart](#),  
    [TimerTriggerSource\\_LineEnd](#),  
    [TimerTriggerSource\\_ExposureStart](#),  
    [TimerTriggerSource\\_ExposureEnd](#),  
    [TimerTriggerSource\\_Line0](#),  
    [TimerTriggerSource\\_Line1](#),  
    [TimerTriggerSource\\_Line2](#),  
    [TimerTriggerSource\\_UserOutput0](#),  
    [TimerTriggerSource\\_UserOutput1](#),  
    [TimerTriggerSource\\_UserOutput2](#),  
    [TimerTriggerSource\\_Counter0Start](#),  
    [TimerTriggerSource\\_Counter1Start](#),  
    [TimerTriggerSource\\_Counter2Start](#),  
    [TimerTriggerSource\\_Counter0End](#),  
    [TimerTriggerSource\\_Counter1End](#),  
    [TimerTriggerSource\\_Counter2End](#),  
    [TimerTriggerSource\\_Timer0Start](#),  
    [TimerTriggerSource\\_Timer1Start](#),  
    [TimerTriggerSource\\_Timer2Start](#),  
    [TimerTriggerSource\\_Timer0End](#),  
    [TimerTriggerSource\\_Timer1End](#),  
    [TimerTriggerSource\\_Timer2End](#),  
    [TimerTriggerSource\\_Encoder0](#),  
    [TimerTriggerSource\\_Encoder1](#),  
    [TimerTriggerSource\\_Encoder2](#),  
    [TimerTriggerSource\\_SoftwareSignal0](#),  
    [TimerTriggerSource\\_SoftwareSignal1](#),  
    [TimerTriggerSource\\_SoftwareSignal2](#),  
    [TimerTriggerSource\\_Action0](#),  
    [TimerTriggerSource\\_Action1](#),  
    [TimerTriggerSource\\_Action2](#),  
    [TimerTriggerSource\\_LinkTrigger0](#),  
    [TimerTriggerSource\\_LinkTrigger1](#),  
    [TimerTriggerSource\\_LinkTrigger2](#),  
    [NUM\\_TIMERTRIGGERSOURCE](#) }  
• `enum spinTimerTriggerActivationEnums {`  
    [TimerTriggerActivation\\_RisingEdge](#),  
    [TimerTriggerActivation\\_FallingEdge](#),  
    [TimerTriggerActivation\\_AnyEdge](#),  
    [TimerTriggerActivation\\_LevelHigh](#),  
    [TimerTriggerActivation\\_LevelLow](#),



```

    NUM_TIMERTRIGGERACTIVATION }

• enum spinEncoderSelectorEnums {
    EncoderSelector_Encoder0,
    EncoderSelector_Encoder1,
    EncoderSelector_Encoder2,
    NUM_ENCODERSELECTOR }

• enum spinEncoderSourceAEnums {
    EncoderSourceA_Off,
    EncoderSourceA_Line0,
    EncoderSourceA_Line1,
    EncoderSourceA_Line2,
    NUM_ENCODERSOURCEA }

• enum spinEncoderSourceBEnums {
    EncoderSourceB_Off,
    EncoderSourceB_Line0,
    EncoderSourceB_Line1,
    EncoderSourceB_Line2,
    NUM_ENCODERSOURCEB }

• enum spinEncoderModeEnums {
    EncoderMode_FourPhase,
    EncoderMode_HighResolution,
    NUM_ENCODERMODE }

• enum spinEncoderOutputModeEnums {
    EncoderOutputMode_Off,
    EncoderOutputMode_PositionUp,
    EncoderOutputMode_PositionDown,
    EncoderOutputMode_DirectionUp,
    EncoderOutputMode_DirectionDown,
    EncoderOutputMode_Motion,
    NUM_ENCODEROUTPUTMODE }

• enum spinEncoderStatusEnums {
    EncoderStatus_EncoderUp,
    EncoderStatus_EncoderDown,
    EncoderStatus_EncoderIdle,
    EncoderStatus_EncoderStatic,
    NUM_ENCODERSTATUS }

• enum spinEncoderResetSourceEnums {
    EncoderResetSource_Off,
    EncoderResetSource_AcquisitionTrigger,
    EncoderResetSource_AcquisitionStart,
    EncoderResetSource_AcquisitionEnd,
    EncoderResetSource_FrameTrigger,
    EncoderResetSource_FrameStart,
    EncoderResetSource_FrameEnd,
    EncoderResetSource_ExposureStart,
    EncoderResetSource_ExposureEnd,
    EncoderResetSource_Line0,
    EncoderResetSource_Line1,
    EncoderResetSource_Line2,
    EncoderResetSource_Counter0Start,
    EncoderResetSource_Counter1Start,
    EncoderResetSource_Counter2Start,
    EncoderResetSource_Counter0End,
    EncoderResetSource_Counter1End,
    EncoderResetSource_Counter2End,
    EncoderResetSource_Timer0Start,
    EncoderResetSource_Timer1Start,
    EncoderResetSource_Timer2Start,

```

```

EncoderResetSource_Timer0End,
EncoderResetSource_Timer1End,
EncoderResetSource_Timer2End,
EncoderResetSource_UserOutput0,
EncoderResetSource_UserOutput1,
EncoderResetSource_UserOutput2,
EncoderResetSource_SoftwareSignal0,
EncoderResetSource_SoftwareSignal1,
EncoderResetSource_SoftwareSignal2,
EncoderResetSource_Action0,
EncoderResetSource_Action1,
EncoderResetSource_Action2,
EncoderResetSource_LinkTrigger0,
EncoderResetSource_LinkTrigger1,
EncoderResetSource_LinkTrigger2,
NUM_ENCODERRESETSOURCE }

• enum spinEncoderResetActivationEnums {
EncoderResetActivation_RisingEdge,
EncoderResetActivation_FallingEdge,
EncoderResetActivation_AnyEdge,
EncoderResetActivation_LevelHigh,
EncoderResetActivation_LevelLow,
NUM_ENCODERRESETACTIVATION }

• enum spinSoftwareSignalSelectorEnums {
SoftwareSignalSelector_SoftwareSignal0,
SoftwareSignalSelector_SoftwareSignal1,
SoftwareSignalSelector_SoftwareSignal2,
NUM_SOFTWARESIGNALSELECTOR }

• enum spinActionUnconditionalModeEnums {
ActionUnconditionalMode_Off,
ActionUnconditionalMode_On,
NUM_ACTIONUNCONDITIONALMODE }

• enum spinSourceSelectorEnums {
SourceSelector_Source0,
SourceSelector_Source1,
SourceSelector_Source2,
SourceSelector_All,
NUM_SOURCESELECTOR }

• enum spinTransferSelectorEnums {
TransferSelector_Stream0,
TransferSelector_Stream1,
TransferSelector_Stream2,
TransferSelector_All,
NUM_TRANSFERSELECTOR }

• enum spinTransferTriggerSelectorEnums {
TransferTriggerSelector_TransferStart,
TransferTriggerSelector_TransferStop,
TransferTriggerSelector_TransferAbort,
TransferTriggerSelector_TransferPause,
TransferTriggerSelector_TransferResume,
TransferTriggerSelector_TransferActive,
TransferTriggerSelector_TransferBurstStart,
TransferTriggerSelector_TransferBurstStop,
NUM_TRANSFERTRIGGERSELECTOR }

• enum spinTransferTriggerModeEnums {
TransferTriggerMode_Off,
TransferTriggerMode_On,
NUM_TRANSFERTRIGGERMODE }

```

- enum [spinTransferTriggerSourceEnums](#) {  
    [TransferTriggerSource\\_Line0](#),  
    [TransferTriggerSource\\_Line1](#),  
    [TransferTriggerSource\\_Line2](#),  
    [TransferTriggerSource\\_Counter0Start](#),  
    [TransferTriggerSource\\_Counter1Start](#),  
    [TransferTriggerSource\\_Counter2Start](#),  
    [TransferTriggerSource\\_Counter0End](#),  
    [TransferTriggerSource\\_Counter1End](#),  
    [TransferTriggerSource\\_Counter2End](#),  
    [TransferTriggerSource\\_Timer0Start](#),  
    [TransferTriggerSource\\_Timer1Start](#),  
    [TransferTriggerSource\\_Timer2Start](#),  
    [TransferTriggerSource\\_Timer0End](#),  
    [TransferTriggerSource\\_Timer1End](#),  
    [TransferTriggerSource\\_Timer2End](#),  
    [TransferTriggerSource\\_SoftwareSignal0](#),  
    [TransferTriggerSource\\_SoftwareSignal1](#),  
    [TransferTriggerSource\\_SoftwareSignal2](#),  
    [TransferTriggerSource\\_Action0](#),  
    [TransferTriggerSource\\_Action1](#),  
    [TransferTriggerSource\\_Action2](#),  
    [NUM\\_TRANSFERTRIGGERSOURCE](#) }
- enum [spinTransferTriggerActivationEnums](#) {  
    [TransferTriggerActivation\\_RisingEdge](#),  
    [TransferTriggerActivation\\_FallingEdge](#),  
    [TransferTriggerActivation\\_AnyEdge](#),  
    [TransferTriggerActivation\\_LevelHigh](#),  
    [TransferTriggerActivation\\_LevelLow](#),  
    [NUM\\_TRANSFERTRIGGERACTIVATION](#) }
- enum [spinTransferStatusSelectorEnums](#) {  
    [TransferStatusSelector\\_Streaming](#),  
    [TransferStatusSelector\\_Paused](#),  
    [TransferStatusSelector\\_Stopping](#),  
    [TransferStatusSelector\\_Stopped](#),  
    [TransferStatusSelector\\_QueueOverflow](#),  
    [NUM\\_TRANSFERSTATUSSELECTOR](#) }
- enum [spinTransferComponentSelectorEnums](#) {  
    [TransferComponentSelector\\_Red](#),  
    [TransferComponentSelector\\_Green](#),  
    [TransferComponentSelector\\_Blue](#),  
    [TransferComponentSelector\\_All](#),  
    [NUM\\_TRANSFERCOMPONENTSELECTOR](#) }
- enum [spinScan3dDistanceUnitEnums](#) {  
    [Scan3dDistanceUnit\\_Millimeter](#),  
    [Scan3dDistanceUnit\\_Inch](#),  
    [NUM\\_SCAN3DDISTANCEUNIT](#) }
- enum [spinScan3dCoordinateSystemEnums](#) {  
    [Scan3dCoordinateSystem\\_Cartesian](#),  
    [Scan3dCoordinateSystem\\_Spherical](#),  
    [Scan3dCoordinateSystem\\_Cylindrical](#),  
    [NUM\\_SCAN3DCOORDINATESYSTEM](#) }
- enum [spinScan3dOutputModeEnums](#) {  
    [Scan3dOutputMode\\_UncalibratedC](#),  
    [Scan3dOutputMode\\_CalibratedABC\\_Grid](#),  
    [Scan3dOutputMode\\_CalibratedABC\\_PointCloud](#),  
    [Scan3dOutputMode\\_CalibratedAC](#),  
    [Scan3dOutputMode\\_CalibratedAC\\_Linescan](#),

```

Scan3dOutputMode_CalibratedC,
Scan3dOutputMode_CalibratedC_Linescan,
Scan3dOutputMode_RectifiedC,
Scan3dOutputMode_RectifiedC_Linescan,
Scan3dOutputMode_DisparityC,
Scan3dOutputMode_DisparityC_Linescan,
NUM_SCAN3DOUTPUTMODE }

• enum spinScan3dCoordinateSystemReferenceEnums {
    Scan3dCoordinateSystemReference_Anchor,
    Scan3dCoordinateSystemReference_Transformed,
    NUM_SCAN3DCOORDINATESYSTEMREFERENCE }

• enum spinScan3dCoordinateSelectorEnums {
    Scan3dCoordinateSelector_CoordinateA,
    Scan3dCoordinateSelector_CoordinateB,
    Scan3dCoordinateSelector_CoordinateC,
    NUM_SCAN3DCOORDINATESELECTOR }

• enum spinScan3dCoordinateTransformSelectorEnums {
    Scan3dCoordinateTransformSelector_RotationX,
    Scan3dCoordinateTransformSelector_RotationY,
    Scan3dCoordinateTransformSelector_RotationZ,
    Scan3dCoordinateTransformSelector_TranslationX,
    Scan3dCoordinateTransformSelector_TranslationY,
    Scan3dCoordinateTransformSelector_TranslationZ,
    NUM_SCAN3DCOORDINATETRANSFORMSELECTOR }

• enum spinScan3dCoordinateReferenceSelectorEnums {
    Scan3dCoordinateReferenceSelector_RotationX,
    Scan3dCoordinateReferenceSelector_RotationY,
    Scan3dCoordinateReferenceSelector_RotationZ,
    Scan3dCoordinateReferenceSelector_TranslationX,
    Scan3dCoordinateReferenceSelector_TranslationY,
    Scan3dCoordinateReferenceSelector_TranslationZ,
    NUM_SCAN3DCOORDINATEREFERENCESELECTOR }

• enum spinChunkImageComponentEnums {
    ChunkImageComponent_Intensity,
    ChunkImageComponent_Color,
    ChunkImageComponent_Infrared,
    ChunkImageComponent_Ultraviolet,
    ChunkImageComponent_Range,
    ChunkImageComponent_Disparity,
    ChunkImageComponent_Confidence,
    ChunkImageComponent_Scatter,
    NUM_CHUNKIMAGECOMPONENT }

• enum spinChunkCounterSelectorEnums {
    ChunkCounterSelector_Counter0,
    ChunkCounterSelector_Counter1,
    ChunkCounterSelector_Counter2,
    NUM_CHUNKCOUNTERSELECTOR }

• enum spinChunkTimerSelectorEnums {
    ChunkTimerSelector_Timer0,
    ChunkTimerSelector_Timer1,
    ChunkTimerSelector_Timer2,
    NUM_CHUNKTIMERSELECTOR }

• enum spinChunkEncoderSelectorEnums {
    ChunkEncoderSelector_Encoder0,
    ChunkEncoderSelector_Encoder1,
    ChunkEncoderSelector_Encoder2,
    NUM_CHUNKENCODERSELECTOR }

```

- enum [spinChunkEncoderStatusEnums](#) {  
[ChunkEncoderStatus\\_EncoderUp](#),  
[ChunkEncoderStatus\\_EncoderDown](#),  
[ChunkEncoderStatus\\_EncoderIdle](#),  
[ChunkEncoderStatus\\_EncoderStatic](#),  
[NUM\\_CHUNKENCODERSTATUS](#) }
- enum [spinChunkExposureTimeSelectorEnums](#) {  
[ChunkExposureTimeSelector\\_Common](#),  
[ChunkExposureTimeSelector\\_Red](#),  
[ChunkExposureTimeSelector\\_Green](#),  
[ChunkExposureTimeSelector\\_Blue](#),  
[ChunkExposureTimeSelector\\_Cyan](#),  
[ChunkExposureTimeSelector\\_Magenta](#),  
[ChunkExposureTimeSelector\\_Yellow](#),  
[ChunkExposureTimeSelector\\_Infrared](#),  
[ChunkExposureTimeSelector\\_Ultraviolet](#),  
[ChunkExposureTimeSelector\\_Stage1](#),  
[ChunkExposureTimeSelector\\_Stage2](#),  
[NUM\\_CHUNKEXPOSURETIMESELECTOR](#) }
- enum [spinChunkSourceIDEnums](#) {  
[ChunkSourceID\\_Source0](#),  
[ChunkSourceID\\_Source1](#),  
[ChunkSourceID\\_Source2](#),  
[NUM\\_CHUNKSOURCEID](#) }
- enum [spinChunkRegionIDEnums](#) {  
[ChunkRegionID\\_Region0](#),  
[ChunkRegionID\\_Region1](#),  
[ChunkRegionID\\_Region2](#),  
[NUM\\_CHUNKREGIONID](#) }
- enum [spinChunkTransferStreamIDEnums](#) {  
[ChunkTransferStreamID\\_Stream0](#),  
[ChunkTransferStreamID\\_Stream1](#),  
[ChunkTransferStreamID\\_Stream2](#),  
[ChunkTransferStreamID\\_Stream3](#),  
[NUM\\_CHUNKTRANSFERSTREAMID](#) }
- enum [spinChunkScan3dDistanceUnitEnums](#) {  
[ChunkScan3dDistanceUnit\\_Millimeter](#),  
[ChunkScan3dDistanceUnit\\_Inch](#),  
[NUM\\_CHUNKSCAN3DDISTANCEUNIT](#) }
- enum [spinChunkScan3dOutputModeEnums](#) {  
[ChunkScan3dOutputMode\\_UncalibratedC](#),  
[ChunkScan3dOutputMode\\_CalibratedABC\\_Grid](#),  
[ChunkScan3dOutputMode\\_CalibratedABC\\_PointCloud](#),  
[ChunkScan3dOutputMode\\_CalibratedAC](#),  
[ChunkScan3dOutputMode\\_CalibratedAC\\_Linescan](#),  
[ChunkScan3dOutputMode\\_CalibratedC](#),  
[ChunkScan3dOutputMode\\_CalibratedC\\_Linescan](#),  
[ChunkScan3dOutputMode\\_RectifiedC](#),  
[ChunkScan3dOutputMode\\_RectifiedC\\_Linescan](#),  
[ChunkScan3dOutputMode\\_DisparityC](#),  
[ChunkScan3dOutputMode\\_DisparityC\\_Linescan](#),  
[NUM\\_CHUNKSCAN3DOUTPUTMODE](#) }
- enum [spinChunkScan3dCoordinateSystemEnums](#) {  
[ChunkScan3dCoordinateSystem\\_Cartesian](#),  
[ChunkScan3dCoordinateSystem\\_Spherical](#),  
[ChunkScan3dCoordinateSystem\\_Cylindrical](#),  
[NUM\\_CHUNKSCAN3DCOORDINATESYSTEM](#) }

- `enum spinChunkScan3dCoordinateSystemReferenceEnums {`  
`ChunkScan3dCoordinateSystemReference_Anchor,`  
`ChunkScan3dCoordinateSystemReference_Transformed,`  
`NUM_CHUNKSCAN3DCOORDINATESYSTEMREFERENCE }`
- `enum spinChunkScan3dCoordinateSelectorEnums {`  
`ChunkScan3dCoordinateSelector_CoordinateA,`  
`ChunkScan3dCoordinateSelector_CoordinateB,`  
`ChunkScan3dCoordinateSelector_CoordinateC,`  
`NUM_CHUNKSCAN3DCOORDINATESELECTOR }`
- `enum spinChunkScan3dCoordinateTransformSelectorEnums {`  
`ChunkScan3dCoordinateTransformSelector_RotationX,`  
`ChunkScan3dCoordinateTransformSelector_RotationY,`  
`ChunkScan3dCoordinateTransformSelector_RotationZ,`  
`ChunkScan3dCoordinateTransformSelector_TranslationX,`  
`ChunkScan3dCoordinateTransformSelector_TranslationY,`  
`ChunkScan3dCoordinateTransformSelector_TranslationZ,`  
`NUM_CHUNKSCAN3DCOORDINATETRANSFORMSELECTOR }`
- `enum spinChunkScan3dCoordinateReferenceSelectorEnums {`  
`ChunkScan3dCoordinateReferenceSelector_RotationX,`  
`ChunkScan3dCoordinateReferenceSelector_RotationY,`  
`ChunkScan3dCoordinateReferenceSelector_RotationZ,`  
`ChunkScan3dCoordinateReferenceSelector_TranslationX,`  
`ChunkScan3dCoordinateReferenceSelector_TranslationY,`  
`ChunkScan3dCoordinateReferenceSelector_TranslationZ,`  
`NUM_CHUNKSCAN3DCOORDINATEREFERENCESELECTOR }`
- `enum spinDeviceTapGeometryEnums {`  
`DeviceTapGeometry_Geometry_1X_1Y,`  
`DeviceTapGeometry_Geometry_1X2_1Y,`  
`DeviceTapGeometry_Geometry_1X2_1Y2,`  
`DeviceTapGeometry_Geometry_2X_1Y,`  
`DeviceTapGeometry_Geometry_2X_1Y2Geometry_2XE_1Y,`  
`DeviceTapGeometry_Geometry_2XE_1Y2,`  
`DeviceTapGeometry_Geometry_2XM_1Y,`  
`DeviceTapGeometry_Geometry_2XM_1Y2,`  
`DeviceTapGeometry_Geometry_1X_1Y2,`  
`DeviceTapGeometry_Geometry_1X_2YE,`  
`DeviceTapGeometry_Geometry_1X3_1Y,`  
`DeviceTapGeometry_Geometry_3X_1Y,`  
`DeviceTapGeometry_Geometry_1X,`  
`DeviceTapGeometry_Geometry_1X2,`  
`DeviceTapGeometry_Geometry_2X,`  
`DeviceTapGeometry_Geometry_2XE,`  
`DeviceTapGeometry_Geometry_2XM,`  
`DeviceTapGeometry_Geometry_1X3,`  
`DeviceTapGeometry_Geometry_3X,`  
`DeviceTapGeometry_Geometry_1X4_1Y,`  
`DeviceTapGeometry_Geometry_4X_1Y,`  
`DeviceTapGeometry_Geometry_2X2_1Y,`  
`DeviceTapGeometry_Geometry_2X2E_1YGeometry_2X2M_1Y,`  
`DeviceTapGeometry_Geometry_1X2_2YE,`  
`DeviceTapGeometry_Geometry_2X_2YE,`  
`DeviceTapGeometry_Geometry_2XE_2YE,`  
`DeviceTapGeometry_Geometry_2XM_2YE,`  
`DeviceTapGeometry_Geometry_1X4,`  
`DeviceTapGeometry_Geometry_4X,`  
`DeviceTapGeometry_Geometry_2X2,`  
`DeviceTapGeometry_Geometry_2X2E,`  
`DeviceTapGeometry_Geometry_2X2M,`

```

DeviceTapGeometry_Geometry_1X8_1Y,
DeviceTapGeometry_Geometry_8X_1Y,
DeviceTapGeometry_Geometry_4X2_1Y,
DeviceTapGeometry_Geometry_2X2E_2YE,
DeviceTapGeometry_Geometry_1X8,
DeviceTapGeometry_Geometry_8X,
DeviceTapGeometry_Geometry_4X2,
DeviceTapGeometry_Geometry_4X2E,
DeviceTapGeometry_Geometry_4X2E_1Y,
DeviceTapGeometry_Geometry_1X10_1Y,
DeviceTapGeometry_Geometry_10X_1Y,
DeviceTapGeometry_Geometry_1X10,
DeviceTapGeometry_Geometry_10X,
NUM_DEVICETAPGEOMETRY }

• enum spinGevPhysicalLinkConfigurationEnums {
    GevPhysicalLinkConfiguration_SingleLink,
    GevPhysicalLinkConfiguration_MultiLink,
    GevPhysicalLinkConfiguration_StaticLAG,
    GevPhysicalLinkConfiguration_DynamicLAG,
    NUM_GEVPHYSICALLINKCONFIGURATION }

• enum spinGevCurrentPhysicalLinkConfigurationEnums {
    GevCurrentPhysicalLinkConfiguration_SingleLink,
    GevCurrentPhysicalLinkConfiguration_MultiLink,
    GevCurrentPhysicalLinkConfiguration_StaticLAG,
    GevCurrentPhysicalLinkConfiguration_DynamicLAG,
    NUM_GEVCURRENTPHYSICALLINKCONFIGURATION }

• enum spinGevIPConfigurationStatusEnums {
    GevIPConfigurationStatus_None,
    GevIPConfigurationStatus_PersistentIP,
    GevIPConfigurationStatus_DHCP,
    GevIPConfigurationStatus_LLA,
    GevIPConfigurationStatus_ForceIP,
    NUM_GEVIPCONFIGURATIONSTATUS }

• enum spinGevGVCPExtendedStatusCodesSelectorEnums {
    GevGVCPExtendedStatusCodesSelector_Version1_1,
    GevGVCPExtendedStatusCodesSelector_Version2_0,
    NUM_GEVGVCPEXTENDEDSTATUSCODESSELECTOR }

• enum spinGevGVSPExtendedIDModeEnums {
    GevGVSPExtendedIDMode_Off,
    GevGVSPExtendedIDMode_On,
    NUM_GEVGVSPEXTENDEDIDMODE }

• enum spinCIConfigurationEnums {
    CIConfiguration_Base,
    CIConfiguration_Medium,
    CIConfiguration_Full,
    CIConfiguration_DualBase,
    CIConfiguration_EightyBit,
    NUM_CLCONFIGURATION }

• enum spinCITimeSlotsCountEnums {
    CITimeSlotsCount_One,
    CITimeSlotsCount_Two,
    CITimeSlotsCount_Three,
    NUM_CLTIMESLOTSCOUNT }

• enum spinCxpLinkConfigurationStatusEnums {
    CxpLinkConfigurationStatus_None,
    CxpLinkConfigurationStatus_Pending,
    CxpLinkConfigurationStatus_CXP1_X1,
    CxpLinkConfigurationStatus_CXP2_X1,

```

```

CxpLinkConfigurationStatus_CXP3_X1,
CxpLinkConfigurationStatus_CXP5_X1,
CxpLinkConfigurationStatus_CXP6_X1,
CxpLinkConfigurationStatus_CXP1_X2,
CxpLinkConfigurationStatus_CXP2_X2,
CxpLinkConfigurationStatus_CXP3_X2,
CxpLinkConfigurationStatus_CXP5_X2,
CxpLinkConfigurationStatus_CXP6_X2,
CxpLinkConfigurationStatus_CXP1_X3,
CxpLinkConfigurationStatus_CXP2_X3,
CxpLinkConfigurationStatus_CXP3_X3,
CxpLinkConfigurationStatus_CXP5_X3,
CxpLinkConfigurationStatus_CXP6_X3,
CxpLinkConfigurationStatus_CXP1_X4,
CxpLinkConfigurationStatus_CXP2_X4,
CxpLinkConfigurationStatus_CXP3_X4,
CxpLinkConfigurationStatus_CXP5_X4,
CxpLinkConfigurationStatus_CXP6_X4,
CxpLinkConfigurationStatus_CXP1_X5,
CxpLinkConfigurationStatus_CXP2_X5,
CxpLinkConfigurationStatus_CXP3_X5,
CxpLinkConfigurationStatus_CXP5_X5,
CxpLinkConfigurationStatus_CXP6_X5,
CxpLinkConfigurationStatus_CXP1_X6,
CxpLinkConfigurationStatus_CXP2_X6,
CxpLinkConfigurationStatus_CXP3_X6,
CxpLinkConfigurationStatus_CXP5_X6,
CxpLinkConfigurationStatus_CXP6_X6,
NUM_CXPLINKCONFIGURATIONSTATUS }

```

- `enum spinCxpLinkConfigurationPreferredEnums {`

```

CxpLinkConfigurationPreferred_CXP1_X1,
CxpLinkConfigurationPreferred_CXP2_X1,
CxpLinkConfigurationPreferred_CXP3_X1,
CxpLinkConfigurationPreferred_CXP5_X1,
CxpLinkConfigurationPreferred_CXP6_X1,
CxpLinkConfigurationPreferred_CXP1_X2,
CxpLinkConfigurationPreferred_CXP2_X2,
CxpLinkConfigurationPreferred_CXP3_X2,
CxpLinkConfigurationPreferred_CXP5_X2,
CxpLinkConfigurationPreferred_CXP6_X2,
CxpLinkConfigurationPreferred_CXP1_X3,
CxpLinkConfigurationPreferred_CXP2_X3,
CxpLinkConfigurationPreferred_CXP3_X3,
CxpLinkConfigurationPreferred_CXP5_X3,
CxpLinkConfigurationPreferred_CXP6_X3,
CxpLinkConfigurationPreferred_CXP1_X4,
CxpLinkConfigurationPreferred_CXP2_X4,
CxpLinkConfigurationPreferred_CXP3_X4,
CxpLinkConfigurationPreferred_CXP5_X4,
CxpLinkConfigurationPreferred_CXP6_X4,
CxpLinkConfigurationPreferred_CXP1_X5,
CxpLinkConfigurationPreferred_CXP2_X5,
CxpLinkConfigurationPreferred_CXP3_X5,
CxpLinkConfigurationPreferred_CXP5_X5,
CxpLinkConfigurationPreferred_CXP6_X5,
CxpLinkConfigurationPreferred_CXP1_X6,
CxpLinkConfigurationPreferred_CXP2_X6,
CxpLinkConfigurationPreferred_CXP3_X6,

```

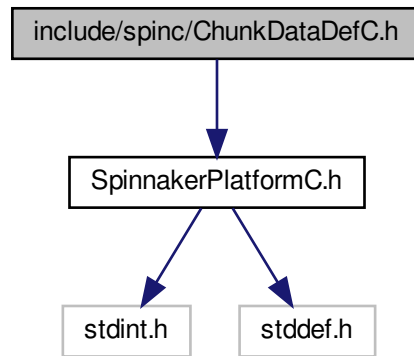


```
CxpLinkConfigurationPreferred_CXP5_X6,  
CxpLinkConfigurationPreferred_CXP6_X6,  
NUM_CXPLINKCONFIGURATIONPREFERRED }
```

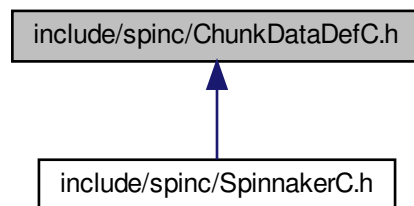
- `enum spinCxpLinkConfigurationEnums {`  
    CxpLinkConfiguration\_Auto,  
    CxpLinkConfiguration\_CXP1\_X1,  
    CxpLinkConfiguration\_CXP2\_X1,  
    CxpLinkConfiguration\_CXP3\_X1,  
    CxpLinkConfiguration\_CXP5\_X1,  
    CxpLinkConfiguration\_CXP6\_X1,  
    CxpLinkConfiguration\_CXP1\_X2,  
    CxpLinkConfiguration\_CXP2\_X2,  
    CxpLinkConfiguration\_CXP3\_X2,  
    CxpLinkConfiguration\_CXP5\_X2,  
    CxpLinkConfiguration\_CXP6\_X2,  
    CxpLinkConfiguration\_CXP1\_X3,  
    CxpLinkConfiguration\_CXP2\_X3,  
    CxpLinkConfiguration\_CXP3\_X3,  
    CxpLinkConfiguration\_CXP5\_X3,  
    CxpLinkConfiguration\_CXP6\_X3,  
    CxpLinkConfiguration\_CXP1\_X4,  
    CxpLinkConfiguration\_CXP2\_X4,  
    CxpLinkConfiguration\_CXP3\_X4,  
    CxpLinkConfiguration\_CXP5\_X4,  
    CxpLinkConfiguration\_CXP6\_X4,  
    CxpLinkConfiguration\_CXP1\_X5,  
    CxpLinkConfiguration\_CXP2\_X5,  
    CxpLinkConfiguration\_CXP3\_X5,  
    CxpLinkConfiguration\_CXP5\_X5,  
    CxpLinkConfiguration\_CXP6\_X5,  
    CxpLinkConfiguration\_CXP1\_X6,  
    CxpLinkConfiguration\_CXP2\_X6,  
    CxpLinkConfiguration\_CXP3\_X6,  
    CxpLinkConfiguration\_CXP5\_X6,  
    CxpLinkConfiguration\_CXP6\_X6,  
    NUM\_CXPLINKCONFIGURATION }  
  
• `enum spinCxpConnectionTestModeEnums {`  
    CxpConnectionTestMode\_Off,  
    CxpConnectionTestMode\_Mode1,  
    NUM\_CXPCONNECTIONTESTMODE }  
  
• `enum spinCxpPoCxpStatusEnums {`  
    CxpPoCxpStatus\_Auto,  
    CxpPoCxpStatus\_Off,  
    CxpPoCxpStatus\_Tripped,  
    NUM\_CXPPOCXPSTATUS }

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

Include dependency graph for ChunkDataDefC.h:



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



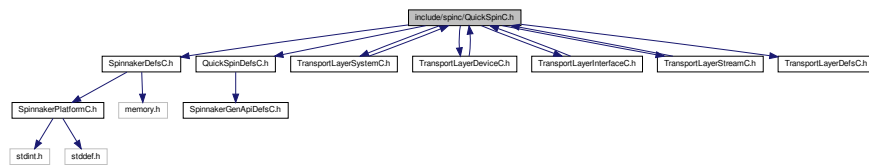
### Data Structures

- struct [spinChunkData](#)

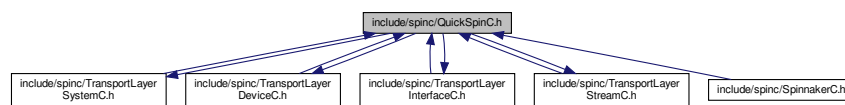
*The type of information that can be obtained from image chunk data.*

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

Include dependency graph for QuickSpinC.h:



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

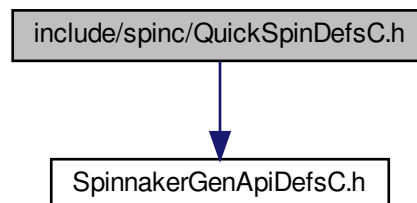


### Functions

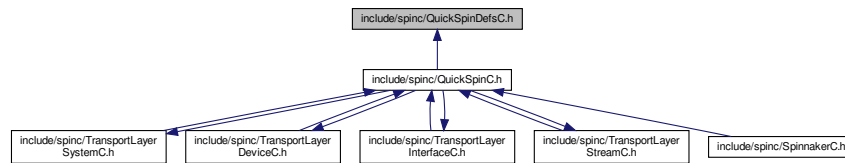
- [SPINNAKERC\\_API quickSpinInit](#) ([spinCamera](#) hCamera, [quickSpin](#) \*pQuickSpin)
- [SPINNAKERC\\_API quickSpinInitEx](#) ([spinCamera](#) hCamera, [quickSpin](#) \*pQuickSpin, [quickSpinTLDevice](#) \*pQuickSpinTLDevice, [quickSpinTLStream](#) \*pQuickSpinTLStream)
- [SPINNAKERC\\_API quickSpinTLDeviceInit](#)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

## 8.6 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](#)

### 8.6.1 Typedef Documentation

#### 8.6.1.1 quickSpinBooleanNode

```
typedef spinNodeHandle quickSpinBooleanNode
```

#### 8.6.1.2 quickSpinCommandNode

```
typedef spinNodeHandle quickSpinCommandNode
```

#### 8.6.1.3 quickSpinEnumerationNode

```
typedef spinNodeHandle quickSpinEnumerationNode
```

#### 8.6.1.4 quickSpinFloatNode

```
typedef spinNodeHandle quickSpinFloatNode
```

#### 8.6.1.5 quickSpinIntegerNode

```
typedef spinNodeHandle quickSpinIntegerNode
```

#### 8.6.1.6 quickSpinRegisterNode

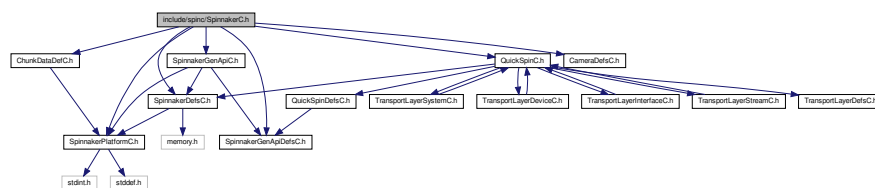
```
typedef spinNodeHandle quickSpinRegisterNode
```

#### 8.6.1.7 quickSpinStringNode

```
typedef spinNodeHandle quickSpinStringNode
```

## 8.7 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 spinSystemRegisterLogEvent](#) ([spinSystem](#) hSystem, [spinLogEvent](#) hLogEvent)  
*Registers a logging event to the system (events registered in this way must be unregistered)*
- [SPINNAKERC\\_API spinSystemUnregisterLogEvent](#) ([spinSystem](#) hSystem, [spinLogEvent](#) hLogEvent)  
*Unregisters a selected logging event from the system.*
- [SPINNAKERC\\_API spinSystemUnregisterAllLogEvents](#) ([spinSystem](#) hSystem)  
*Unregisters all logging events from the system.*
- [SPINNAKERC\\_API spinSystemIsInUse](#) ([spinSystem](#) hSystem, [bool8\\_t](#) \*pbIsInUse)  
*Checks whether a system is currently in use.*
- [SPINNAKERC\\_API spinSystemRegisterArrivalEvent](#) ([spinSystem](#) hSystem, [spinArrivalEvent](#) hArrivalEvent)  
*Registers an arrival event to every interface on the system (events registered this way must be unregistered)*
- [SPINNAKERC\\_API spinSystemRegisterRemovalEvent](#) ([spinSystem](#) hSystem, [spinRemovalEvent](#) hRemovalEvent)  
*Registers a removal event to the system to every interface on the system (events registered this way must be unregistered)*
- [SPINNAKERC\\_API spinSystemUnregisterArrivalEvent](#) ([spinSystem](#) hSystem, [spinArrivalEvent](#) hArrivalEvent)  
*Unregisters an arrival event from the system (events registered in this way must be unregistered)*

- Unregisters an arrival event from the system.*
- [SPINNAKERC\\_API spinSystemUnregisterRemovalEvent](#) ([spinSystem](#) hSystem, [spinRemovalEvent](#) hRemovalEvent)
- Unregisters a removal event from the system.*
- [SPINNAKERC\\_API spinSystemRegisterInterfaceEvent](#) ([spinSystem](#) hSystem, [spinInterfaceEvent](#) hInterfaceEvent)
- Registers an interface event (arrival and removal) to every interface on the system (interface events registered this way must be unregistered)*
- [SPINNAKERC\\_API spinSystemUnregisterInterfaceEvent](#) ([spinSystem](#) hSystem, [spinInterfaceEvent](#) hInterfaceEvent)
- Unregisters an interface event 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 spinInterfaceUpdateCameras](#) ([spinInterface](#) hInterface, [bool8\\_t](#) \*pbChanged)  
*Checks whether any cameras have been connected or disconnected on an interface.*
- [SPINNAKERC\\_API spinInterfaceGetCameras](#) ([spinInterface](#) hInterface, [spinCameraList](#) hCameraList)  
*Retrieves a camera list from an interface; camera lists must be created and destroy.*
- [SPINNAKERC\\_API spinInterfaceGetCamerasEx](#) ([spinInterface](#) hInterface, [bool8\\_t](#) bUpdateCameras, [spinCameraList](#) hCameraList)  
*Retrieves a camera list from an interface; manually set whether to update the cameras; camera lists must be created and destroyed.*
- [SPINNAKERC\\_API spinInterfaceGetTLNodeMap](#) ([spinInterface](#) hInterface, [spinNodeMapHandle](#) \*phNodeMap)  
*Retrieves the transport layer nodemap from an interface.*
- [SPINNAKERC\\_API spinInterfaceRegisterArrivalEvent](#) ([spinInterface](#) hInterface, [spinArrivalEvent](#) hArrivalEvent)  
*Registers an arrival event on an interface (events registered in this way must be unregistered)*
- [SPINNAKERC\\_API spinInterfaceRegisterRemovalEvent](#) ([spinInterface](#) hInterface, [spinRemovalEvent](#) hRemovalEvent)  
*Registers a removal event on an interface (events registered in this way must be unregistered)*
- [SPINNAKERC\\_API spinInterfaceUnregisterArrivalEvent](#) ([spinInterface](#) hInterface, [spinArrivalEvent](#) hArrivalEvent)  
*Unregisters an arrival event from an interface.*
- [SPINNAKERC\\_API spinInterfaceUnregisterRemovalEvent](#) ([spinInterface](#) hInterface, [spinRemovalEvent](#) hRemovalEvent)  
*Unregisters a removal event from an interface.*
- [SPINNAKERC\\_API spinInterfaceRegisterInterfaceEvent](#) ([spinInterface](#) hInterface, [spinInterfaceEvent](#) hInterfaceEvent)  
*Registers an interface event (both arrival and removal) on an interface.*
- [SPINNAKERC\\_API spinInterfaceUnregisterInterfaceEvent](#) ([spinInterface](#) hInterface, [spinInterfaceEvent](#) hInterfaceEvent)  
*Unregisters an interface event 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](#) \*phNodeMap)  
*Retrieves the transport layer device nodemap from a camera.*
- [SPINNAKERC\\_API spinCameraGetTLStreamNodeMap](#) ([spinCamera](#) hCamera, [spinNodeMapHandle](#) \*phNodeMap)



- Retrieves the transport layer stream nodemap from a camera.*
- [SPINNAKERC\\_API spinCameraGetAccessMode](#) ([spinCamera](#) hCamera, [spinAccessMode](#) \*pAccessMode)
- Retrieves the access mode of a camera (as an enum, spinAccessMode)*
- [SPINNAKERC\\_API spinCameraReadPort](#) ([spinCamera](#) hCamera, uint64\_t iAddress, void \*pBuffer, size\_t iSize)
- [SPINNAKERC\\_API spinCameraWritePort](#) ([spinCamera](#) hCamera, uint64\_t iAddress, void \*pBuffer, size\_t iSize)
- [SPINNAKERC\\_API spinCameraBeginAcquisition](#) ([spinCamera](#) hCamera)
- Has a camera start acquiring images.*
- [SPINNAKERC\\_API spinCameraEndAcquisition](#) ([spinCamera](#) hCamera)
- Has a camera stop acquiring images.*
- [SPINNAKERC\\_API spinCameraGetNextImage](#) ([spinCamera](#) hCamera, [spinImage](#) \*phImage)
- Retrieves an image from a camera.*
- [SPINNAKERC\\_API spinCameraGetNextImageEx](#) ([spinCamera](#) hCamera, uint64\_t grabTimeout, [spinImage](#) \*phImage)
- Retrieves an image from a camera; manually set the timeout in milliseconds.*
- [SPINNAKERC\\_API spinCameraGetUniqueID](#) ([spinCamera](#) hCamera, char \*pBuf, size\_t \*pBufLen)
- Retrieves a unique identifier for a camera.*
- [SPINNAKERC\\_API spinCameralStreaming](#) ([spinCamera](#) hCamera, bool8\_t \*pbIsStreaming)
- Checks whether a camera is currently acquiring images.*
- [SPINNAKERC\\_API spinCameraGetGuiXml](#) ([spinCamera](#) hCamera, char \*pBuf, size\_t \*pBufLen)
- Retrieves the GUI XML from a camera.*
- [SPINNAKERC\\_API spinCameraRegisterDeviceEvent](#) ([spinCamera](#) hCamera, [spinDeviceEvent](#) hDevice↔Event)
- Registers a universal device event (every device event type) to a camera.*
- [SPINNAKERC\\_API spinCameraRegisterDeviceEventEx](#) ([spinCamera](#) hCamera, [spinDeviceEvent](#) hDevice↔Event, const char \*pName)
- Registers a specific device event (only one device event type) to a camera.*
- [SPINNAKERC\\_API spinCameraUnregisterDeviceEvent](#) ([spinCamera](#) hCamera, [spinDeviceEvent](#) hDevice↔Event)
- Unregisters a device event from a camera.*
- [SPINNAKERC\\_API spinCameraRegisterImageEvent](#) ([spinCamera](#) hCamera, [spinImageEvent](#) hImageEvent)
- Registers an image event to a camera.*
- [SPINNAKERC\\_API spinCameraUnregisterImageEvent](#) ([spinCamera](#) hCamera, [spinImageEvent](#) hImage↔Event)
- Unregisters an image event from a camera.*
- [SPINNAKERC\\_API spinCameraRelease](#) ([spinCamera](#) hCamera)
- Releases a camera.*
- [SPINNAKERC\\_API spinCameralValid](#) ([spinCamera](#) hCamera, bool8\_t \*pbValid)
- Checks whether a camera is still valid for use.*
- [SPINNAKERC\\_API spinCameralInitialized](#) ([spinCamera](#) hCamera, bool8\_t \*pbInit)
- Checks whether a camera is currently initialized.*
- [SPINNAKERC\\_API spinCameraDiscoverMaxPacketSize](#) ([spinCamera](#) hCamera, unsigned int \*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 spinImageDestroy](#) ([spinImage](#) hImage)  
*Destroys an image.*
- [SPINNAKERC\\_API spinImageSetDefaultColorProcessing](#) ([spinColorProcessingAlgorithm](#) algorithm)  
*Sets the default color processing algorithm of all images (if not otherwise set)*
- [SPINNAKERC\\_API spinImageGetDefaultColorProcessing](#) ([spinColorProcessingAlgorithm](#) \*pAlgorithm)  
*Retrieves the default color processing algorithm.*
- [SPINNAKERC\\_API spinImageGetColorProcessing](#) ([spinImage](#) hImage, [spinColorProcessingAlgorithm](#) \*pAlgorithm)  
*Retrieves the color processing algorithm of a specific image.*
- [SPINNAKERC\\_API spinImageConvert](#) ([spinImage](#) hSrcImage, [spinPixelFormatEnums](#) pixelFormat, [spinImage](#) hDestImage)  
*Converts the pixel format of one image into a new image.*
- [SPINNAKERC\\_API spinImageConvertEx](#) ([spinImage](#) hSrcImage, [spinPixelFormatEnums](#) pixelFormat, [spinColorProcessingAlgorithm](#) algorithm, [spinImage](#) hDestImage)  
*Converts the pixel format and color processing algorithm of one image into a new image.*
- [SPINNAKERC\\_API spinImageReset](#) ([spinImage](#) hImage, [size\\_t](#) width, [size\\_t](#) height, [size\\_t](#) offsetX, [size\\_t](#) offsetY, [spinPixelFormatEnums](#) pixelFormat)  
*Resets an image with some set properties.*
- [SPINNAKERC\\_API spinImageResetEx](#) ([spinImage](#) hImage, [size\\_t](#) width, [size\\_t](#) height, [size\\_t](#) offsetX, [size\\_t](#) offsetY, [spinPixelFormatEnums](#) pixelFormat, void \*pData)  
*Resets an image with some set properties and image data.*
- [SPINNAKERC\\_API spinImageGetID](#) ([spinImage](#) hImage, [uint64\\_t](#) \*pId)  
*Retrieves the ID of an image.*
- [SPINNAKERC\\_API spinImageGetData](#) ([spinImage](#) hImage, void \*\*ppData)  
*Retrieves the image data of an image.*
- [SPINNAKERC\\_API spinImageGetPrivateData](#) ([spinImage](#) hImage, void \*\*ppData)  
*Retrieves the private data of an image.*
- [SPINNAKERC\\_API spinImageGetBufferSize](#) ([spinImage](#) hImage, [size\\_t](#) \*pSize)  
*Retrieves the buffer size of an image.*
- [SPINNAKERC\\_API spinImageDeepCopy](#) ([spinImage](#) hSrcImage, [spinImage](#) hDestImage)  
*Creates a deep copy of an image (the destination image must be created as an empty image prior to the deep copy)*
- [SPINNAKERC\\_API spinImageGetWidth](#) ([spinImage](#) hImage, [size\\_t](#) \*pWidth)  
*Retrieves the width of an image.*
- [SPINNAKERC\\_API spinImageGetHeight](#) ([spinImage](#) hImage, [size\\_t](#) \*pHeight)  
*Retrieves the height of an image.*
- [SPINNAKERC\\_API spinImageGetOffsetX](#) ([spinImage](#) hImage, [size\\_t](#) \*pOffsetX)  
*Retrieves the offset of an image along its X axis.*
- [SPINNAKERC\\_API spinImageGetOffsetY](#) ([spinImage](#) hImage, [size\\_t](#) \*pOffsetY)  
*Retrieves the offset of an image along its Y axis.*
- [SPINNAKERC\\_API spinImageGetPaddingX](#) ([spinImage](#) hImage, [size\\_t](#) \*pPaddingX)  
*Retrieves the padding of an image along its X axis.*
- [SPINNAKERC\\_API spinImageGetPaddingY](#) ([spinImage](#) hImage, [size\\_t](#) \*pPaddingY)  
*Retrieves the padding of an image along its Y axis.*
- [SPINNAKERC\\_API spinImageGetFrameID](#) ([spinImage](#) hImage, [uint64\\_t](#) \*pFrameID)  
*Retrieves the frame ID of an image.*
- [SPINNAKERC\\_API spinImageGetTimeStamp](#) ([spinImage](#) hImage, [uint64\\_t](#) \*pTimeStamp)  
*Retrieves the timestamp of an image.*
- [SPINNAKERC\\_API spinImageGetPayloadType](#) ([spinImage](#) hImage, [size\\_t](#) \*pPayloadType)

- Retrieves the payload type of an image (as an enum, spinPayloadTypeInfos)*

  - [SPINNAKERC\\_API spinImageGetTLPayloadType](#) ([spinImage](#) hImage, [spinPayloadTypeInfoIDs](#) \*pPayloadType)
- Retrieves the transport layer payload type of an image (as an enum, spinPayloadTypeInfos)*

  - [SPINNAKERC\\_API spinImageGetPixelFormat](#) ([spinImage](#) hImage, [spinPixelFormatEnums](#) \*pPixelFormat)
- Retrieves the pixel format of an image (as an enum, spinPixelFormatEnums)*

  - [SPINNAKERC\\_API spinImageGetTLPixelFormat](#) ([spinImage](#) hImage, [uint64\\_t](#) \*pPixelFormat)
- Retrieves the transport layer pixel format of an image (as an unsigned integer)*

  - [SPINNAKERC\\_API spinImageGetTLPixelFormatNamespace](#) ([spinImage](#) hImage, [spinPixelFormatNamespaceID](#) \*pPixelFormatNamespace)
- Retrieves the transport layer pixel format namespace of an image (as an enum, spinPixelFormatNamespaceID)*

  - [SPINNAKERC\\_API spinImageGetPixelFormatName](#) ([spinImage](#) hImage, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)
- Retrieves the pixel format of an image (as a symbolic)*

  - [SPINNAKERC\\_API spinImageIsIncomplete](#) ([spinImage](#) hImage, [bool8\\_t](#) \*pIsIncomplete)
- Checks whether an image is incomplete.*

  - [SPINNAKERC\\_API spinImageGetValidPayloadSize](#) ([spinImage](#) hImage, [size\\_t](#) \*pSize)
- Retrieves the valid payload size of an image.*

  - [SPINNAKERC\\_API spinImageSave](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [spinImageFileFormat](#) format)
- Saves an image using a specified file format (using an enum, spinImageFileFormat)*

  - [SPINNAKERC\\_API spinImageSaveFromExt](#) ([spinImage](#) hImage, [const char](#) \*pFilename)
- Saves an image using a specified file format (using the extension of the filename)*

  - [SPINNAKERC\\_API spinImageSavePng](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinPNGOption](#) \*pOption)
- Saves an image as a PNG image.*

  - [SPINNAKERC\\_API spinImageSavePpm](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinPPMOption](#) \*pOption)
- Saves an image as a PPM image.*

  - [SPINNAKERC\\_API spinImageSavePgm](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinPGMOption](#) \*pOption)
- Saves an image as an PGM image.*

  - [SPINNAKERC\\_API spinImageSaveTiff](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinTIFFOption](#) \*pOption)
- Saves an image as a TIFF image.*

  - [SPINNAKERC\\_API spinImageSaveJpeg](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinJPEGOption](#) \*pOption)
- Saves an image as a JPEG image.*

  - [SPINNAKERC\\_API spinImageSaveJpg2](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinJPG2Option](#) \*pOption)
- Saves an image as a JPEG 2000 image.*

  - [SPINNAKERC\\_API spinImageSaveBmp](#) ([spinImage](#) hImage, [const char](#) \*pFilename, [const spinBMPOption](#) \*pOption)
- Saves an image as a BMP image.*

  - [SPINNAKERC\\_API spinImageGetChunkLayoutID](#) ([spinImage](#) hImage, [uint64\\_t](#) \*pId)
- Retrieves the chunk layout ID of an image.*

  - [SPINNAKERC\\_API spinImageCalculateStatistics](#) ([spinImage](#) hImage, [const spinImageStatistics](#) hStatistics)
- Calculates the image statistics of an image.*

  - [SPINNAKERC\\_API spinImageGetStatus](#) ([spinImage](#) hImage, [spinImageStatus](#) \*pStatus)
- Retrieves the image status of an image.*

  - [SPINNAKERC\\_API spinImageGetStatusDescription](#) ([spinImageStatus](#) status, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)
- Retrieves the description of image status.*

  - [SPINNAKERC\\_API spinImageRelease](#) ([spinImage](#) hImage)

- Releases an image.*

  - [SPINNAKERC\\_API spinImageHasCRC](#) ([spinImage](#) hImage, [bool8\\_t](#) \*pbHasCRC)

*Checks whether an image has CRC.*
- [SPINNAKERC\\_API spinImageCheckCRC](#) ([spinImage](#) hImage, [bool8\\_t](#) \*pbCheckCRC)

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

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

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

*Retrieves the stride of an image.*
- [SPINNAKERC\\_API spinDeviceEventCreate](#) ([spinDeviceEvent](#) \*phDeviceEvent, [spinDeviceEventFunction](#) pFunction, void \*pUserData)

*Creates a device event.*
- [SPINNAKERC\\_API spinDeviceEventDestroy](#) ([spinDeviceEvent](#) hDeviceEvent)

*Destroys a device event.*
- [SPINNAKERC\\_API spinImageEventCreate](#) ([spinImageEvent](#) \*phImageEvent, [spinImageEventFunction](#) pFunction, void \*pUserData)

*Creates an image event.*
- [SPINNAKERC\\_API spinImageEventDestroy](#) ([spinImageEvent](#) hImageEvent)

*Destroys an image event.*
- [SPINNAKERC\\_API spinArrivalEventCreate](#) ([spinArrivalEvent](#) \*phArrivalEvent, [spinArrivalEventFunction](#) pFunction, void \*pUserData)

*Creates an arrival event.*
- [SPINNAKERC\\_API spinArrivalEventDestroy](#) ([spinArrivalEvent](#) hArrivalEvent)

*Destroys an arrival event.*
- [SPINNAKERC\\_API spinRemovalEventCreate](#) ([spinRemovalEvent](#) \*phRemovalEvent, [spinRemovalEventFunction](#) pFunction, void \*pUserData)

*Creates a removal event.*
- [SPINNAKERC\\_API spinRemovalEventDestroy](#) ([spinRemovalEvent](#) hRemovalEvent)

*Destroys a removal event.*
- [SPINNAKERC\\_API spinInterfaceEventCreate](#) ([spinInterfaceEvent](#) \*phInterfaceEvent, [spinArrivalEventFunction](#) pArrivalFunction, [spinRemovalEventFunction](#) pRemovalFunction, void \*pUserData)

*Creates an interface event (both arrival and removal)*
- [SPINNAKERC\\_API spinInterfaceEventDestroy](#) ([spinInterfaceEvent](#) hInterfaceEvent)

*Destroys an interface event (both arrival and removal)*
- [SPINNAKERC\\_API spinLogEventCreate](#) ([spinLogEvent](#) \*phLogEvent, [spinLogEventFunction](#) pFunction, void \*pUserData)

*Creates a log event.*
- [SPINNAKERC\\_API spinLogEventDestroy](#) ([spinLogEvent](#) hLogEvent)

*Destroys a log event.*
- [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\\_DEPRECATED](#) ("spinAVIRecorderOpenUncompressed is deprecated, use [spinVideoOpenUncompressed](#) instead.", [spinAVIRecorderOpenUncompressed](#)([spinAVIRecorder](#) \*phRecorder, [const char](#) \*pName, [spinAVIOption](#) option))
- [SPINNAKERC\\_API\\_DEPRECATED](#) ("spinAVIRecorderOpenMJPEG is deprecated, use [spinVideoOpenMJPEG](#) instead.", [spinAVIRecorderOpenMJPEG](#)([spinAVIRecorder](#) \*phRecorder, [const char](#) \*pName, [spinMJPGOption](#) option))
- [SPINNAKERC\\_API\\_DEPRECATED](#) ("spinAVIRecorderOpenH264 is deprecated, use [spinVideoOpenH264](#) instead.", [spinAVIRecorderOpenH264](#)([spinAVIRecorder](#) \*phRecorder, [const char](#) \*pName, [spinH264Option](#) option))
- [SPINNAKERC\\_API\\_DEPRECATED](#) ("spinAVIRecorderAppend is deprecated, use [spinVideoAppend](#) instead.", [spinAVIRecorderAppend](#)([spinAVIRecorder](#) hRecorder, [spinImage](#) hImage))
- [SPINNAKERC\\_API\\_DEPRECATED](#) ("spinAVISetMaximumSize is deprecated, use [spinVideoSetMaximumFileSize](#) instead.", [spinAVISetMaximumSize](#)([spinAVIRecorder](#) hRecorder, [unsigned int](#) size))  
*Set the maximum file size (in megabytes) of a AVI/MP4 file.*
- [SPINNAKERC\\_API\\_DEPRECATED](#) ("spinAVIRecorderClose is deprecated, use [spinVideoClose](#) instead.", [spinAVIRecorderClose](#)([spinAVIRecorder](#) hRecorder))
- [SPINNAKERC\\_API spinImageChunkDataGetIntValue](#) ([spinImage](#) hImage, [const char](#) \*pName, [int64\\_t](#) \*pValue)
- [SPINNAKERC\\_API spinImageChunkDataGetFloatValue](#) ([spinImage](#) hImage, [const char](#) \*pName, [double](#) \*pValue)

## 8.7.1 Function Documentation

### 8.7.1.1 spinCameraForceIP()

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

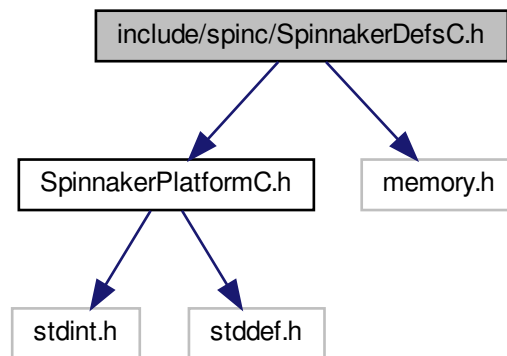
Forces the camera to be on the same subnet as its corresponding interface.

#### Returns

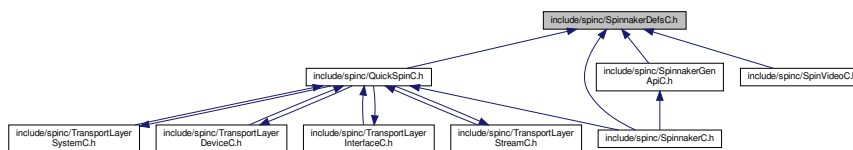
[spinError](#) The error code; returns [SPINNAKER\\_ERR\\_SUCCESS](#) (or 0) for no error

## 8.8 include/spinc/SpinnakerDefsC.h File Reference

Include dependency graph for SpinnakerDefsC.h:



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



### Data Structures

- struct [spinPNGOption](#)  
Options for saving PNG images.
- struct [spinPPMOption](#)  
Options for saving PPM images.
- struct [spinPGMOption](#)  
Options for saving PGM images.
- struct [spinTIFFOption](#)  
Options for saving TIFF images.
- struct [spinJPEGOption](#)  
Options for saving JPEG images.
- struct [spinJPG2Option](#)  
Options for saving JPEG 2000 images.
- struct [spinBMPOption](#)  
Options for saving BMP images.
- struct [spinMJPGOption](#)  
Options for saving MJPG videos.



- struct [spinH264Option](#)  
*Options for saving H264 videos.*
- struct [spinAVIOption](#)  
*Options for saving uncompressed videos.*
- struct [spinLibraryVersion](#)  
*Provides easier access to the current version of Spinnaker.*
- struct [actionCommandResult](#)  
*Action Command Result.*

## Typedefs

- typedef uint8\_t [bool8\\_t](#)
- typedef void \* [spinSystem](#)  
*Handle for system functionality.*
- typedef void \* [spinInterfaceList](#)  
*Handle for interface list functionality.*
- typedef void \* [spinInterface](#)  
*Handle for interface functionality.*
- typedef void \* [spinCameraList](#)  
*Handle for interface functionality.*
- typedef void \* [spinCamera](#)  
*Handle for camera functionality.*
- typedef void \* [spinImage](#)  
*Handle for image functionality.*
- typedef void \* [spinImageStatistics](#)  
*Handle for image statistics functionality.*
- typedef void \* [spinDeviceEvent](#)  
*Handle for device event functionality.*
- typedef void \* [spinImageEvent](#)  
*Handle for image event functionality.*
- typedef void \* [spinArrivalEvent](#)  
*Handle for arrival event functionality.*
- typedef void \* [spinRemovalEvent](#)  
*Handle for removal event functionality.*
- typedef void \* [spinInterfaceEvent](#)  
*Handle for interface event functionality.*
- typedef void \* [spinLogEvent](#)  
*Handle for logging event functionality.*
- typedef void \* [spinLogEventData](#)  
*Handle for logging event data functionality.*
- typedef void \* [spinDeviceEventData](#)  
*Handle for device event data functionality.*
- typedef void \* [spinAVIRecorder](#)  
*Handle for video recording functionality.*
- typedef void \* [spinVideo](#)
- typedef void(\* [spinDeviceEventFunction](#)) (const [spinDeviceEventData](#) hEventData, const char \*pEventName, void \*pUserData)  
*Function signatures are used to create and trigger callbacks and events.*
- typedef void(\* [spinImageEventFunction](#)) (const [spinImage](#) hImage, void \*pUserData)
- typedef void(\* [spinArrivalEventFunction](#)) (uint64\_t deviceSerialNumber, void \*pUserData)
- typedef void(\* [spinRemovalEventFunction](#)) (uint64\_t deviceSerialNumber, void \*pUserData)
- typedef void(\* [spinLogEventFunction](#)) (const [spinLogEventData](#) hEventData, void \*pUserData)



## Enumerations

- enum `spinError` {
  - `SPINNAKER_ERR_SUCCESS` = 0,
  - `SPINNAKER_ERR_ERROR` = -1001,
  - `SPINNAKER_ERR_NOT_INITIALIZED` = -1002,
  - `SPINNAKER_ERR_NOT_IMPLEMENTED` = -1003,
  - `SPINNAKER_ERR_RESOURCE_IN_USE` = -1004,
  - `SPINNAKER_ERR_ACCESS_DENIED` = -1005,
  - `SPINNAKER_ERR_INVALID_HANDLE` = -1006,
  - `SPINNAKER_ERR_INVALID_ID` = -1007,
  - `SPINNAKER_ERR_NO_DATA` = -1008,
  - `SPINNAKER_ERR_INVALID_PARAMETER` = -1009,
  - `SPINNAKER_ERR_IO` = -1010,
  - `SPINNAKER_ERR_TIMEOUT` = -1011,
  - `SPINNAKER_ERR_ABORT` = -1012,
  - `SPINNAKER_ERR_INVALID_BUFFER` = -1013,
  - `SPINNAKER_ERR_NOT_AVAILABLE` = -1014,
  - `SPINNAKER_ERR_INVALID_ADDRESS` = -1015,
  - `SPINNAKER_ERR_BUFFER_TOO_SMALL` = -1016,
  - `SPINNAKER_ERR_INVALID_INDEX` = -1017,
  - `SPINNAKER_ERR_PARSING_CHUNK_DATA` = -1018,
  - `SPINNAKER_ERR_INVALID_VALUE` = -1019,
  - `SPINNAKER_ERR_RESOURCE_EXHAUSTED` = -1020,
  - `SPINNAKER_ERR_OUT_OF_MEMORY` = -1021,
  - `SPINNAKER_ERR_BUSY` = -1022,
  - `GENICAM_ERR_INVALID_ARGUMENT` = -2001,
  - `GENICAM_ERR_OUT_OF_RANGE` = -2002,
  - `GENICAM_ERR_PROPERTY` = -2003,
  - `GENICAM_ERR_RUN_TIME` = -2004,
  - `GENICAM_ERR_LOGICAL` = -2005,
  - `GENICAM_ERR_ACCESS` = -2006,
  - `GENICAM_ERR_TIMEOUT` = -2007,
  - `GENICAM_ERR_DYNAMIC_CAST` = -2008,
  - `GENICAM_ERR_GENERIC` = -2009,
  - `GENICAM_ERR_BAD_ALLOCATION` = -2010,
  - `SPINNAKER_ERR_IM_CONVERT` = -3001,
  - `SPINNAKER_ERR_IM_COPY` = -3002,
  - `SPINNAKER_ERR_IM_MALLOC` = -3003,
  - `SPINNAKER_ERR_IM_NOT_SUPPORTED` = -3004,
  - `SPINNAKER_ERR_IM_HISTOGRAM_RANGE` = -3005,
  - `SPINNAKER_ERR_IM_HISTOGRAM_MEAN` = -3006,
  - `SPINNAKER_ERR_IM_MIN_MAX` = -3007,
  - `SPINNAKER_ERR_IM_COLOR_CONVERSION` = -3008,
  - `SPINNAKER_ERR_CUSTOM_ID` = -10000 }

*The error codes used in Spinnaker C.*

- enum `spinColorProcessingAlgorithm` {
  - `DEFAULT`,
  - `NO_COLOR_PROCESSING`,
  - `NEAREST_NEIGHBOR`,
  - `NEAREST_NEIGHBOR_AVG`,
  - `BILINEAR`,
  - `EDGE_SENSING`,
  - `HQ_LINEAR`,
  - `IPP`,
  - `DIRECTIONAL_FILTER`,
  - `RIGOROUS`,
  - `WEIGHTED_DIRECTIONAL_FILTER` }

*Color processing algorithms.*

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

*Channels that allow statistics to be calculated.*

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

*File formats to be used for saving images to disk.*

- enum `spinPixelFormatNamespaceID` {  
`SPINNAKER_PIXELFORMAT_NAMESPACE_UNKNOWN` = 0,  
`SPINNAKER_PIXELFORMAT_NAMESPACE_GEV` = 1,  
`SPINNAKER_PIXELFORMAT_NAMESPACE_IIDC` = 2,  
`SPINNAKER_PIXELFORMAT_NAMESPACE_PFNC_16BIT` = 3,  
`SPINNAKER_PIXELFORMAT_NAMESPACE_PFNC_32BIT` = 4,  
`SPINNAKER_PIXELFORMAT_NAMESPACE_CUSTOM_ID` = 1000 }

*This enum represents the namespace in which the TL specific pixel format resides.*

- enum `spinImageStatus` {  
`IMAGE_UNKNOWN_ERROR` = -1,  
`IMAGE_NO_ERROR` = 0,  
`IMAGE_CRC_CHECK_FAILED` = 1,  
`IMAGE_DATA_OVERFLOW` = 2,  
`IMAGE_MISSING_PACKETS` = 3,  
`IMAGE_LEADER_BUFFER_SIZE_INCONSISTENT` = 4,  
`IMAGE_TRAILER_BUFFER_SIZE_INCONSISTENT` = 5,  
`IMAGE_PACKETID_INCONSISTENT` = 6,  
`IMAGE_MISSING_LEADER` = 7,  
`IMAGE_MISSING_TRAILER` = 8,  
`IMAGE_DATA_INCOMPLETE` = 9,  
`IMAGE_INFO_INCONSISTENT` = 10,  
`IMAGE_CHUNK_DATA_INVALID` = 11,  
`IMAGE_NO_SYSTEM_RESOURCES` = 12 }

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

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

*log levels*

- enum [spinPayloadTypeInfoIds](#) {  
PAYLOAD\_TYPE\_UNKNOWN = 0,  
PAYLOAD\_TYPE\_IMAGE = 1,  
PAYLOAD\_TYPE\_RAW\_DATA = 2,  
PAYLOAD\_TYPE\_FILE = 3,  
PAYLOAD\_TYPE\_CHUNK\_DATA = 4,  
PAYLOAD\_TYPE\_JPEG = 5,  
PAYLOAD\_TYPE\_JPEG2000 = 6,  
PAYLOAD\_TYPE\_H264 = 7,  
PAYLOAD\_TYPE\_CHUNK\_ONLY = 8,  
PAYLOAD\_TYPE\_DEVICE\_SPECIFIC = 9,  
PAYLOAD\_TYPE\_MULTI\_PART = 10,  
PAYLOAD\_TYPE\_CUSTOM\_ID = 1000,  
PAYLOAD\_TYPE\_EXTENDED\_CHUNK = 1001 }
- enum [spinCompressionMethod](#) {  
NONE = 1,  
PACKBITS,  
DEFLATE,  
ADOBE\_DEFLATE,  
CCITTFAX3,  
CCITTFAX4,  
LZW,  
JPG }

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

- enum [actionCommandStatus](#) {  
ACTION\_COMMAND\_STATUS\_OK = 0,  
ACTION\_COMMAND\_STATUS\_NO\_REF\_TIME = 0x8013,  
ACTION\_COMMAND\_STATUS\_OVERFLOW = 0x8015,  
ACTION\_COMMAND\_STATUS\_ACTION\_LATE = 0x8016,  
ACTION\_COMMAND\_STATUS\_ERROR = 0x8FFF }

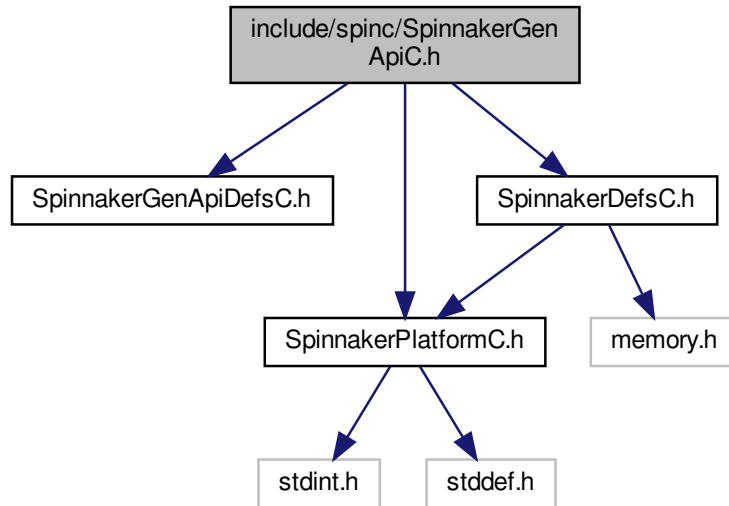
*Possible Status Codes Returned from Action Command.*

## Variables

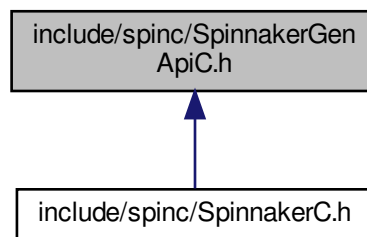
- static const [bool8\\_t False](#) = 0
- static const [bool8\\_t True](#) = 1

## 8.9 include/spinc/SpinnakerGenApiC.h File Reference

Include dependency graph for SpinnakerGenApiC.h:



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



### Functions

- **SPINNAKER\_API spinNodeMapGetNode** (**spinNodeMapHandle** hNodeMap, const char \*pName, **spinNodeHandle** \*phNode)  
*Retrieves a node from the nodemap by name.*
- **SPINNAKER\_API spinNodeMapGetNumNodes** (**spinNodeMapHandle** hNodeMap, size\_t \*pValue)  
*Gets the number of nodes in the map.*
- **SPINNAKER\_API spinNodeMapGetNodeByIndex** (**spinNodeMapHandle** hNodeMap, size\_t index, **spinNodeHandle** \*phNode)

- Retrieves a node from the nodemap by index.*
- [SPINNAKERC\\_API spinNodeMapPoll](#) ([spinNodeMapHandle](#) hNodeMap, [int64\\_t](#) timestamp)  
*Fires nodes which have a polling time.*
- [SPINNAKERC\\_API spinNodesImplemented](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbResult)  
*Checks whether a node is implemented.*
- [SPINNAKERC\\_API spinNodesReadable](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbResult)  
*Checks whether a node is readable.*
- [SPINNAKERC\\_API spinNodesWritable](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbResult)  
*Checks whether a node is writable.*
- [SPINNAKERC\\_API spinNodesAvailable](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) \*pbResult)  
*Checks whether a node is available.*
- [SPINNAKERC\\_API spinNodesEqual](#) ([spinNodeHandle](#) hNodeFirst, [spinNodeHandle](#) hNodeSecond, [bool8\\_t](#) \*pbResult)  
*Checks whether two nodes are equal.*
- [SPINNAKERC\\_API spinNodeGetAccessMode](#) ([spinNodeHandle](#) hNode, [spinAccessMode](#) \*pAccessMode)  
*Retrieves the access mode of a node (as an enum, spinAccessMode)*
- [SPINNAKERC\\_API spinNodeGetName](#) ([spinNodeHandle](#) hNode, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves the name of a node (no whitespace)*
- [SPINNAKERC\\_API spinNodeGetNameSpace](#) ([spinNodeHandle](#) hNode, [spinNameSpace](#) \*pNamespace)  
*Retrieve the namespace of a node (as an enum, spinNameSpace)*
- [SPINNAKERC\\_API spinNodeGetVisibility](#) ([spinNodeHandle](#) hNode, [spinVisibility](#) \*pVisibility)  
*Retrieves the recommended visibility of a node (as an enum, spinVisibility)*
- [SPINNAKERC\\_API spinNodeInvalidateNode](#) ([spinNodeHandle](#) hNode)  
*Invalidates a node in case its values may have changed, rendering it no longer valid.*
- [SPINNAKERC\\_API spinNodeGetCachingMode](#) ([spinNodeHandle](#) hNode, [spinCachingMode](#) \*pCachingMode)  
*Retrieves the caching mode of a node (as an enum, spinCachingMode)*
- [SPINNAKERC\\_API spinNodeGetToolTip](#) ([spinNodeHandle](#) hNode, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves a short description of a node.*
- [SPINNAKERC\\_API spinNodeGetDescription](#) ([spinNodeHandle](#) hNode, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves a longer description of a node.*
- [SPINNAKERC\\_API spinNodeGetDisplayName](#) ([spinNodeHandle](#) hNode, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves the display name of a node (whitespace possible)*
- [SPINNAKERC\\_API spinNodeGetType](#) ([spinNodeHandle](#) hNode, [spinNodeType](#) \*pType)  
*Retrieves the type of a node (as an enum, spinNodeType)*
- [SPINNAKERC\\_API spinNodeGetPollingTime](#) ([spinNodeHandle](#) hNode, [int64\\_t](#) \*pPollingTime)  
*Retrieve the polling time of a node.*
- [SPINNAKERC\\_API spinNodeRegisterCallback](#) ([spinNodeHandle](#) hNode, [spinNodeCallbackFunction](#) pCbFunction, [spinNodeCallbackHandle](#) \*phCb)  
*Registers a callback to a node.*
- [SPINNAKERC\\_API spinNodeDeregisterCallback](#) ([spinNodeHandle](#) hNode, [spinNodeCallbackHandle](#) hCb)  
*Unregisters a callback from a node.*
- [SPINNAKERC\\_API spinNodeGetImposedAccessMode](#) ([spinNodeHandle](#) hNode, [spinAccessMode](#) imposedAccessMode)  
*Retrieves the imposed access mode of a node.*
- [SPINNAKERC\\_API spinNodeGetImposedVisibility](#) ([spinNodeHandle](#) hNode, [spinVisibility](#) imposedVisibility)  
*Retrieves the imposed visibility of a node.*
- [SPINNAKERC\\_API spinNodeToString](#) ([spinNodeHandle](#) hNode, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)  
*Retrieves the value of any node type as a c-string.*
- [SPINNAKERC\\_API spinNodeToStringEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, [char](#) \*pBuf, [size\\_t](#) \*pBufLen)

- Retrieves the value of any node type as a c-string; manually set whether to verify the node.*

  - [SPINNAKERC\\_API spinNodeFromString](#) ([spinNodeHandle](#) hNode, const char \*pBuf)

*Sets the value of any node type from a c-string; it is important to ensure that the value of the c-string is appropriate to the node type.*
- [SPINNAKERC\\_API spinNodeFromStringEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, const char \*pBuf)

*Sets the value of any node type from a c-string; manually set whether to verify the node; ensure the value of the c-string is appropriate to the node type.*
- [SPINNAKERC\\_API spinStringSetValue](#) ([spinNodeHandle](#) hNode, const char \*pBuf)

*Sets the value of a string node.*
- [SPINNAKERC\\_API spinStringSetValueEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, const char \*pBuf)

*Sets the value of a string node; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinStringGetValue](#) ([spinNodeHandle](#) hNode, char \*pBuf, [size\\_t](#) \*pBufLen)

*Retrieves the value of a string node as a c-string.*
- [SPINNAKERC\\_API spinStringGetValueEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, char \*pBuf, [size\\_t](#) \*pBufLen)

*Retrieves the value of a string node as a cstring; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinStringGetMaxLength](#) ([spinNodeHandle](#) hNode, [int64\\_t](#) \*pValue)

*Retrieves the maximum length of the c-string to be returned.*
- [SPINNAKERC\\_API spinIntegerSetValue](#) ([spinNodeHandle](#) hNode, [int64\\_t](#) value)

*Sets the value of an integer node.*
- [SPINNAKERC\\_API spinIntegerSetValueEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, [int64\\_t](#) value)

*Sets the value of an integer node; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinIntegerGetValue](#) ([spinNodeHandle](#) hNode, [int64\\_t](#) \*pValue)

*Retrieves the value of an integer node.*
- [SPINNAKERC\\_API spinIntegerGetValueEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, [int64\\_t](#) \*pValue)

*Retrieves the value of an integer node; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinIntegerGetMin](#) ([spinNodeHandle](#) hNode, [int64\\_t](#) \*pValue)

*Retrieves the minimum value of an integer node; all potential values must be greater than or equal to the minimum.*
- [SPINNAKERC\\_API spinIntegerGetMax](#) ([spinNodeHandle](#) hNode, [int64\\_t](#) \*pValue)

*Retrieves the maximum value of an integer node; all potential values must be lesser than or equal to the maximum.*
- [SPINNAKERC\\_API spinIntegerGetInc](#) ([spinNodeHandle](#) hNode, [int64\\_t](#) \*pValue)

*Retrieves the increment of an integer node; all possible values must be divisible by the increment.*
- [SPINNAKERC\\_API spinIntegerGetRepresentation](#) ([spinNodeHandle](#) hNode, [spinRepresentation](#) \*pValue)

*Retrieves the numerical representation of the value of a node; i.e.*
- [SPINNAKERC\\_API spinFloatSetValue](#) ([spinNodeHandle](#) hNode, double value)

*Sets the value of a float node.*
- [SPINNAKERC\\_API spinFloatSetValueEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, double value)

*Sets the value of a float node; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinFloatGetValue](#) ([spinNodeHandle](#) hNode, double \*pValue)

*Retrieves the value of a float node.*
- [SPINNAKERC\\_API spinFloatGetValueEx](#) ([spinNodeHandle](#) hNode, [bool8\\_t](#) bVerify, double \*pValue)

*Retrieves the value of a float node; manually set whether to verify the node.*
- [SPINNAKERC\\_API spinFloatGetMin](#) ([spinNodeHandle](#) hNode, double \*pValue)

*Retrieves the minimum value of a float node; all potential values must be greater than or equal to the minimum.*
- [SPINNAKERC\\_API spinFloatGetMax](#) ([spinNodeHandle](#) hNode, double \*pValue)

*Retrieves the maximum value of a float node; all potential values must be lesser than or equal to the maximum.*
- [SPINNAKERC\\_API spinFloatGetRepresentation](#) ([spinNodeHandle](#) hNode, [spinRepresentation](#) \*pValue)

*Retrieves the numerical representation of the value of a node; i.e.*
- [SPINNAKERC\\_API spinFloatGetUnit](#) ([spinNodeHandle](#) hNode, char \*pBuf, [size\\_t](#) \*pBufLen)

*Retrieves the units of the float node value.*
- [SPINNAKERC\\_API spinEnumerationGetNumEntries](#) ([spinNodeHandle](#) hNode, [size\\_t](#) \*pValue)

- Retrieves the number of entries of an enum node.*

  - [SPINNAKERC\\_API spinEnumerationGetEntryByIndex](#) ([spinNodeHandle](#) hNode, [size\\_t](#) index, [spinNodeHandle](#) \*phEntry)

*Retrieves an entry node from an enum node using an index.*

  - [SPINNAKERC\\_API spinEnumerationGetEntryByName](#) ([spinNodeHandle](#) hNode, [const char](#) \*pName, [spinNodeHandle](#) \*phEntry)

*Retrieves an entry node from an enum node using the entry's symbolic.*

  - [SPINNAKERC\\_API spinEnumerationGetCurrentEntry](#) ([spinNodeHandle](#) hNode, [spinNodeHandle](#) \*phEntry)

*Retrieves the currently selected entry node from an enum node.*

  - [SPINNAKERC\\_API spinEnumerationSetIntValue](#) ([spinNodeHandle](#) hNode, [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](#) hNode, [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](#) hNode, [size\\_t](#) \*pValue)

*Retrieves the number of a features (or child nodes) or a category node.*

  - [SPINNAKERC\\_API spinCategoryGetFeatureByIndex](#) ([spinNodeHandle](#) hNode, [size\\_t](#) index, [spinNodeHandle](#) \*phFeature)

*Retrieves a node from a category node using an index.*

  - [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)

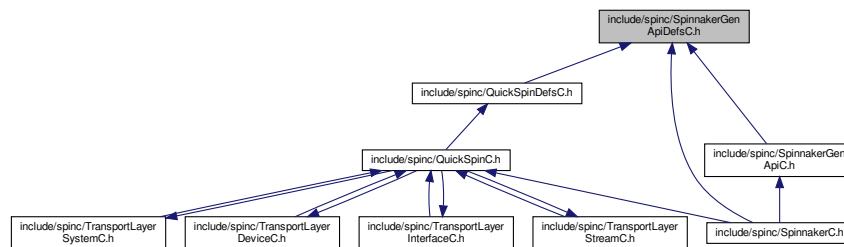
*Sets the value of a register node; manually set whether to verify the node.*

  - [SPINNAKERC\\_API spinRegisterSetReference](#) ([spinNodeHandle](#) hNode, [spinNodeHandle](#) hRef)

*Uses a second node as a reference for a register node.*

## 8.10 include/spinc/SpinnakerGenApiDefsC.h File Reference

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



### Typedefs

- typedef void \* [spinNodeMapHandle](#)  
Handle for nodemap functionality.
- typedef void \* [spinNodeHandle](#)  
Handle for node functionality.
- typedef void \* [spinNodeCallbackHandle](#)  
Handle for callback functionality.
- typedef void(\* [spinNodeCallbackFunction](#)) ([spinNodeHandle](#) hNode)  
Function signatures are used to create and trigger callbacks and events.

### Enumerations

- enum [spinNodeType](#) {  
  [ValueNode](#),  
  [BaseNode](#),  
  [IntegerNode](#),  
  [BooleanNode](#),  
  [FloatNode](#),  
  [CommandNode](#),  
  [StringNode](#),  
  [RegisterNode](#),  
  [EnumerationNode](#),  
  [EnumEntryNode](#),  
  [CategoryNode](#),  
  [PortNode](#),  
  [UnknownNode](#) = -1 }
- enum [spinSign](#) {  
  [Signed](#),  
  [Unsigned](#),  
  [\\_UndefinedSign](#) }
- enum [spinAccessMode](#) {  
  [NI](#),  
  [NA](#),  
  [WO](#),  
  [RO](#),  
  [RW](#),  
  [\\_UndefinedAccesMode](#),  
  [\\_CycleDetectAccesMode](#) }



- enum `spinVisibility` {  
`Beginner` = 0,  
`Expert` = 1,  
`Guru` = 2,  
`Invisible` = 3,  
`_UndefinedVisibility` = 99 }
- enum `spinCachingMode` {  
`NoCache`,  
`WriteThrough`,  
`WriteAround`,  
`_UndefinedCachingMode` }
- enum `spinRepresentation` {  
`Linear`,  
`Logarithmic`,  
`Boolean`,  
`PureNumber`,  
`HexNumber`,  
`IPV4Address`,  
`MACAddress`,  
`_UndefinedRepresentation` }  
*recommended representation of a node value*
- enum `spinEndianness` {  
`BigEndian`,  
`LittleEndian`,  
`_UndefinedEndian` }  
*Endianness of a value in a register.*
- enum `spinNameSpace` {  
`Custom`,  
`Standard`,  
`_UndefinedNameSpace` }  
*Defines if a node name is standard or custom.*
- enum `spinStandardNameSpace` {  
`None`,  
`GEV`,  
`IIDC`,  
`CL`,  
`USB`,  
`_UndefinedStandardNameSpace` }  
*Defines from which standard namespace a node name comes from.*
- enum `spinYesNo` {  
`Yes` = 1,  
`No` = 0,  
`_UndefinedYesNo` = 2 }  
*Defines the chices of a Yes/No alternaitve.*
- enum `spinSlope` {  
`Increasing`,  
`Decreasing`,  
`Varying`,  
`Automatic`,  
`_UndefinedESlope` }  
*typedef for fomula type*
- enum `spinXMLValidation` {  
`xvLoad` = 0x00000001L,  
`xvCycles` = 0x00000002L,  
`xvSFNC` = 0x00000004L,  
`xvDefault` = 0x00000000L,

```
xvAll = 0xffffffffL,
_UndefinedEXMLValidation = 0x8000000L }
```

*typedef describing the different validity checks which can be performed on an XML file*

- enum `spinDisplayNotation` {  
`fnAutomatic`,  
`fnFixed`,  
`fnScientific`,  
`_UndefinedEDisplayNotation` }

*typedef for float notation*

- enum `spinInterfaceType` {  
`intflValue`,  
`intflBase`,  
`intflInteger`,  
`intflBoolean`,  
`intflCommand`,  
`intflFloat`,  
`intflString`,  
`intflRegister`,  
`intflCategory`,  
`intflEnumeration`,  
`intflEnumEntry`,  
`intflPort` }

*typedef for interface type*

- enum `spinLinkType` {  
`ctAllDependingNodes`,  
`ctAllTerminalNodes`,  
`ctInvalidators`,  
`ctReadingChildren`,  
`ctWritingChildren`,  
`ctDependingChildren` }

*typedef for link type*

- enum `spinIncMode` {  
`noIncrement`,  
`fixedIncrement`,  
`listIncrement` }

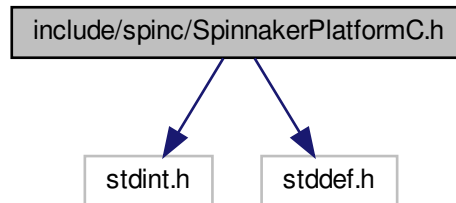
*typedef for increment mode*

- enum `spinInputDirection` {  
`idFrom`,  
`idTo`,  
`idNone` }

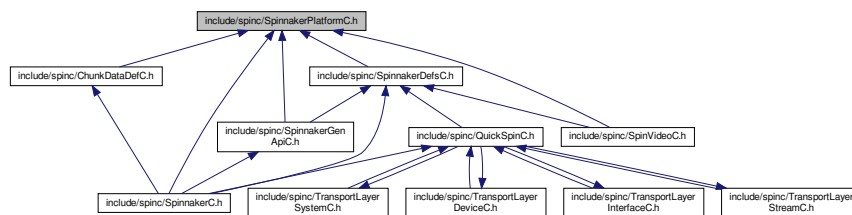
*typedef for link type*

## 8.11 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`

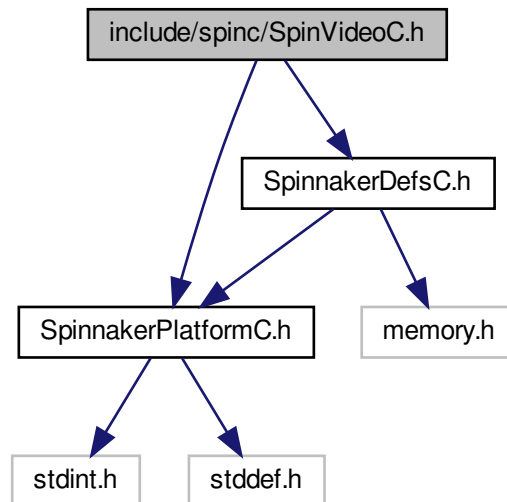
### 8.11.1 Macro Definition Documentation

#### 8.11.1.1 SPINNAKERC\_API

```
#define SPINNAKERC_API SPINC_IMPORT_EXPORT spinError SPINC_CALLTYPE
```

## 8.12 include/spinc/SpinVideoC.h File Reference

Include dependency graph for SpinVideoC.h:

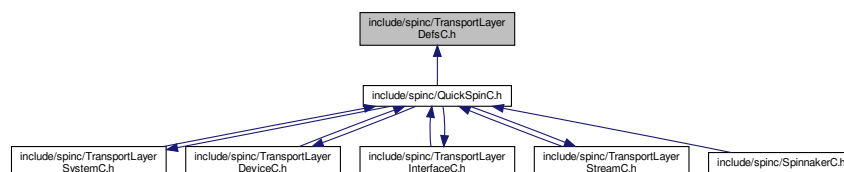


### Functions

- [SPINNAKERC\\_API spinVideoOpenUncompressed](#) ([spinVideo](#) \*phSpinVideo, const char \*pName, [spinAVIOOption](#) option)
- [SPINNAKERC\\_API spinVideoOpenMJPEG](#) ([spinVideo](#) \*phSpinVideo, const char \*pName, [spinMJPEGOption](#) option)
- [SPINNAKERC\\_API spinVideoOpenH264](#) ([spinVideo](#) \*phSpinVideo, const char \*pName, [spinH264Option](#) option)
- [SPINNAKERC\\_API spinVideoAppend](#) ([spinVideo](#) hSpinVideo, [spinImage](#) hImage)
- [SPINNAKERC\\_API spinVideoSetMaximumFileSize](#) ([spinVideo](#) hSpinVideo, unsigned int size)  
Set the maximum file size (in megabytes) of a AVI/MP4 file.
- [SPINNAKERC\\_API spinVideoClose](#) ([spinVideo](#) hSpinVideo)

## 8.13 include/spinc/TransportLayerDefsC.h File Reference

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



## Enumerations

- enum [spinTLStreamTypeEnums](#) {  
[StreamType\\_Mixed](#),  
[StreamType\\_Custom](#),  
[StreamType\\_GEV](#),  
[StreamType\\_CL](#),  
[StreamType\\_IIDC](#),  
[StreamType\\_UVC](#),  
[StreamType\\_CXP](#),  
[StreamType\\_CLHS](#),  
[StreamType\\_U3V](#),  
[StreamType\\_ETHERNET](#),  
[StreamType\\_PCI](#),  
[NUMSTREAMTYPE](#) }
- The enumeration definitions for transport layer nodes.*
- enum [spinTLStreamDefaultBufferCountModeEnums](#) {  
[StreamDefaultBufferCountMode\\_Manual](#),  
[StreamDefaultBufferCountMode\\_Auto](#),  
[NUMSTREAMDEFAULTBUFFERCOUNTMODE](#) }
- enum [spinTLStreamBufferCountModeEnums](#) {  
[StreamBufferCountMode\\_Manual](#),  
[StreamBufferCountMode\\_Auto](#),  
[NUMSTREAMBUFFERCOUNTMODE](#) }
- enum [spinTLStreamBufferHandlingModeEnums](#) {  
[StreamBufferHandlingMode\\_OldestFirst](#),  
[StreamBufferHandlingMode\\_OldestFirstOverwrite](#),  
[StreamBufferHandlingMode\\_NewestFirst](#),  
[StreamBufferHandlingMode\\_NewestFirstOverwrite](#),  
[StreamBufferHandlingMode\\_NewestOnly](#),  
[NUMSTREAMBUFFERHANDLINGMODE](#) }
- enum [spinTLDeviceTypeEnums](#) {  
[DeviceType\\_Mixed](#),  
[DeviceType\\_Custom](#),  
[DeviceType\\_GEV](#),  
[DeviceType\\_CL](#),  
[DeviceType\\_IIDC](#),  
[DeviceType\\_UVC](#),  
[DeviceType\\_CXP](#),  
[DeviceType\\_CLHS](#),  
[DeviceType\\_U3V](#),  
[DeviceType\\_ETHERNET](#),  
[DeviceType\\_PCI](#),  
[NUMDEVICETYPE](#) }
- enum [spinTLDeviceAccessStatusEnums](#) {  
[DeviceAccessStatus\\_Unknown](#),  
[DeviceAccessStatus\\_ReadWrite](#),  
[DeviceAccessStatus\\_ReadOnly](#),  
[DeviceAccessStatus\\_NoAccess](#),  
[NUMDEVICEACCESSSTATUS](#) }
- enum [spinTLGevCCPEnums](#) {  
[GevCCP\\_EnumEntry\\_GevCCP\\_OpenAccess](#),  
[GevCCP\\_EnumEntry\\_GevCCP\\_ExclusiveAccess](#),  
[GevCCP\\_EnumEntry\\_GevCCP\\_ControlAccess](#),  
[NUMGEVCCP](#) }
- enum [spinTLGUIXMLLocationEnums](#) {  
[GUIXMLLocation\\_Device](#),

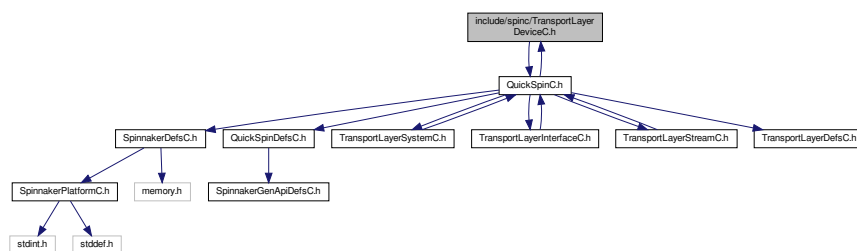
```

GUIXMLLocation_Host,
NUMGUIXMLLOCATION }
• enum spinTLGenICamXMLLocationEnums {
  GenICamXMLLocation_Device,
  GenICamXMLLocation_Host,
  NUMGENICAMXMLLOCATION }
• enum spinTLDeviceEndiannessMechanismEnums {
  DeviceEndiannessMechanism_Legacy,
  DeviceEndiannessMechanism_Standard,
  NUMDEVICEENDIANESSMECHANISM }
• enum spinTLDeviceCurrentSpeedEnums {
  DeviceCurrentSpeed_UnknownSpeed,
  DeviceCurrentSpeed_LowSpeed,
  DeviceCurrentSpeed_FullSpeed,
  DeviceCurrentSpeed_HighSpeed,
  DeviceCurrentSpeed_SuperSpeed,
  NUMDEVICECURRENTSPEED }
• enum spinTLPOEStatusEnums {
  POEStatus_NotSupported,
  POEStatus_PowerOff,
  POEStatus_PowerOn,
  NUMPOESTATUS }
• enum spinTLFilterDriverStatusEnums {
  FilterDriverStatus_NotSupported,
  FilterDriverStatus_Disabled,
  FilterDriverStatus_Enabled,
  NUMFILTERDRIVERSTATUS }

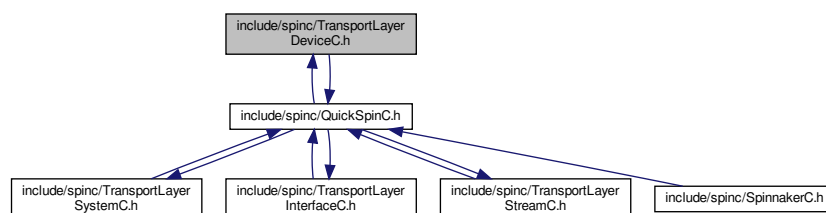
```

## 8.14 include/spinc/TransportLayerDeviceC.h File Reference

Include dependency graph for TransportLayerDeviceC.h:



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

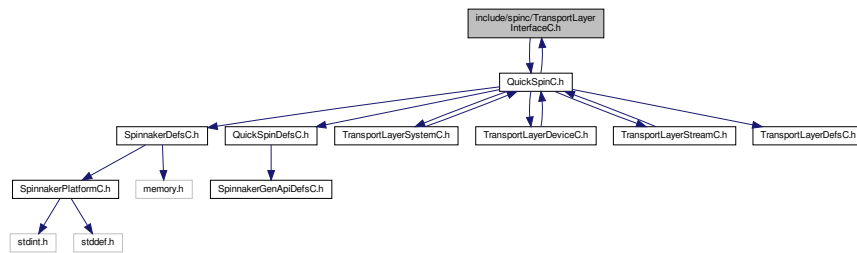


## Data Structures

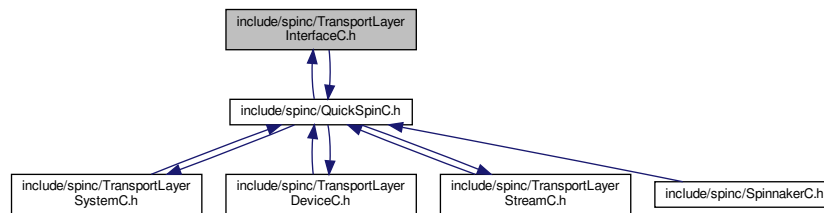
- struct [quickSpinTLDevice](#)

## 8.15 include/spinc/TransportLayerInterfaceC.h File Reference

Include dependency graph for TransportLayerInterfaceC.h:



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

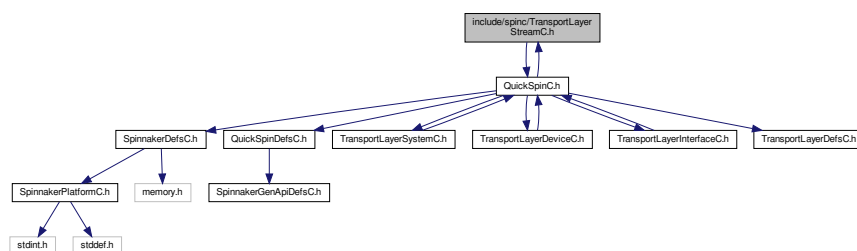


## Data Structures

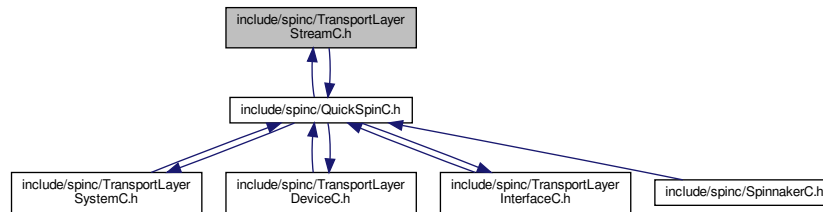
- struct [quickSpinTLInterface](#)

## 8.16 include/spinc/TransportLayerStreamC.h File Reference

Include dependency graph for TransportLayerStreamC.h:



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

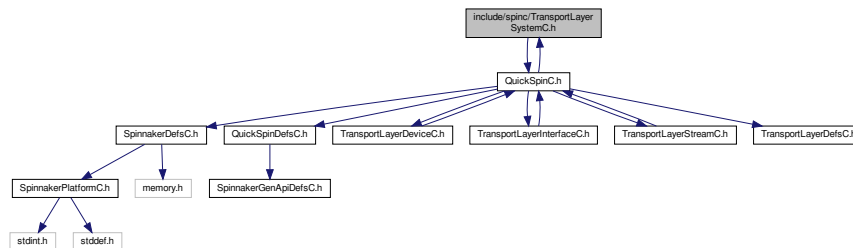


## Data Structures

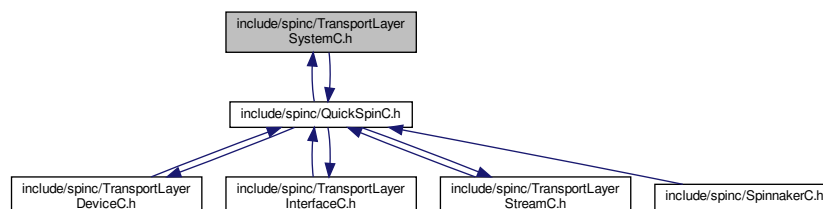
- struct [quickSpinTLStream](#)

## 8.17 include/spinc/TransportLayerSystemC.h File Reference

Include dependency graph for TransportLayerSystemC.h:



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



## Data Structures

- struct [quickSpinTLSystem](#)



# Index

- aPAUSEMACCtrlFramesReceived
  - quickSpin, [351](#)
- aPAUSEMACCtrlFramesTransmitted
  - quickSpin, [351](#)
- AVIRecorder Access, [236](#)
  - SPINNAKER\_API\_DEPRECATED, [236](#), [237](#)
- AasRoiEnable
  - quickSpin, [348](#)
- AasRoiHeight
  - quickSpin, [348](#)
- AasRoiOffsetX
  - quickSpin, [348](#)
- AasRoiOffsetY
  - quickSpin, [348](#)
- AasRoiWidth
  - quickSpin, [348](#)
- AcquisitionAbort
  - quickSpin, [348](#)
- AcquisitionArm
  - quickSpin, [349](#)
- AcquisitionBurstFrameCount
  - quickSpin, [349](#)
- AcquisitionFrameCount
  - quickSpin, [349](#)
- AcquisitionFrameRate
  - quickSpin, [349](#)
- AcquisitionFrameRateEnable
  - quickSpin, [349](#)
- AcquisitionLineRate
  - quickSpin, [349](#)
- AcquisitionMode
  - quickSpin, [349](#)
- AcquisitionResultingFrameRate
  - quickSpin, [349](#)
- AcquisitionStart
  - quickSpin, [350](#)
- AcquisitionStatus
  - quickSpin, [350](#)
- AcquisitionStatusSelector
  - quickSpin, [350](#)
- AcquisitionStop
  - quickSpin, [350](#)
- ActionCommand
  - quickSpinTLInterface, [438](#)
- actionCommandResult, [335](#)
  - DeviceAddress, [335](#)
  - Status, [335](#)
- actionCommandStatus
  - Spinnaker C Structures, [255](#)
- ActionDeviceKey
  - quickSpin, [350](#)
- ActionGroupKey
  - quickSpin, [350](#)
- ActionGroupMask
  - quickSpin, [350](#)
- ActionQueueSize
  - quickSpin, [350](#)
- ActionSelector
  - quickSpin, [351](#)
- ActionUnconditionalMode
  - quickSpin, [351](#)
- AdaptiveCompressionEnable
  - quickSpin, [351](#)
- AdcBitDepth
  - quickSpin, [351](#)
- AutoAlgorithmSelector
  - quickSpin, [351](#)
- AutoExposureControlLoopDamping
  - quickSpin, [351](#)
- AutoExposureControlPriority
  - quickSpin, [352](#)
- AutoExposureEVCompensation
  - quickSpin, [352](#)
- AutoExposureExposureTimeLowerLimit
  - quickSpin, [352](#)
- AutoExposureExposureTimeUpperLimit
  - quickSpin, [352](#)
- AutoExposureGainLowerLimit
  - quickSpin, [352](#)
- AutoExposureGainUpperLimit
  - quickSpin, [352](#)
- AutoExposureGreyValueLowerLimit
  - quickSpin, [352](#)
- AutoExposureGreyValueUpperLimit
  - quickSpin, [352](#)
- AutoExposureLightingMode
  - quickSpin, [353](#)
- AutoExposureMeteringMode
  - quickSpin, [353](#)
- AutoExposureTargetGreyValue
  - quickSpin, [353](#)
- AutoExposureTargetGreyValueAuto
  - quickSpin, [353](#)
- AutoForceIP
  - quickSpinTLInterface, [439](#)
- BalanceRatio
  - quickSpin, [353](#)
- BalanceRatioSelector

- quickSpin, 353
- BalanceWhiteAuto
  - quickSpin, 353
- BalanceWhiteAutoDamping
  - quickSpin, 353
- BalanceWhiteAutoLowerLimit
  - quickSpin, 354
- BalanceWhiteAutoProfile
  - quickSpin, 354
- BalanceWhiteAutoUpperLimit
  - quickSpin, 354
- binaryFile
  - spinPGMOption, 461
  - spinPPMOption, 463
- BinningHorizontal
  - quickSpin, 354
- BinningHorizontalMode
  - quickSpin, 354
- BinningSelector
  - quickSpin, 354
- BinningVertical
  - quickSpin, 354
- BinningVerticalMode
  - quickSpin, 354
- bitrate
  - spinH264Option, 456
- BlackLevel
  - quickSpin, 355
- BlackLevelAuto
  - quickSpin, 355
- BlackLevelAutoBalance
  - quickSpin, 355
- BlackLevelClampingEnable
  - quickSpin, 355
- BlackLevelRaw
  - quickSpin, 355
- BlackLevelSelector
  - quickSpin, 355
- bool8\_t
  - Spinnaker C Definitions, 12
- build
  - spinLibraryVersion, 459
- Camera Access, 171
  - spinCameraBeginAcquisition, 172
  - spinCameraDelInit, 173
  - spinCameraEndAcquisition, 173
  - spinCameraGetAccessMode, 173
  - spinCameraGetGuiXml, 174
  - spinCameraGetNextImage, 174
  - spinCameraGetNextImageEx, 175
  - spinCameraGetNodeMap, 175
  - spinCameraGetTLDeviceNodeMap, 176
  - spinCameraGetTLStreamNodeMap, 176
  - spinCameraGetUniqueID, 177
  - spinCameraInit, 177
  - spinCameralInitialized, 178
  - spinCameralStreaming, 178
  - spinCameralValid, 179
  - spinCameraReadPort, 179
  - spinCameraRegisterDeviceEvent, 179
  - spinCameraRegisterDeviceEventEx, 180
  - spinCameraRegisterImageEvent, 180
  - spinCameraRelease, 181
  - spinCameraUnregisterDeviceEvent, 181
  - spinCameraUnregisterImageEvent, 182
  - spinCameraWritePort, 182
- Camera Enumerations, 13
  - spinAcquisitionModeEnums, 45
  - spinAcquisitionStatusSelectorEnums, 45
  - spinActionUnconditionalModeEnums, 46
  - spinAdcBitDepthEnums, 46
  - spinAutoAlgorithmSelectorEnums, 46
  - spinAutoExposureControlPriorityEnums, 47
  - spinAutoExposureLightingModeEnums, 47
  - spinAutoExposureMeteringModeEnums, 47
  - spinAutoExposureTargetGreyValueAutoEnums, 48
  - spinBalanceRatioSelectorEnums, 48
  - spinBalanceWhiteAutoEnums, 49
  - spinBalanceWhiteAutoProfileEnums, 49
  - spinBinningHorizontalModeEnums, 49
  - spinBinningSelectorEnums, 50
  - spinBinningVerticalModeEnums, 50
  - spinBlackLevelAutoBalanceEnums, 50
  - spinBlackLevelAutoEnums, 51
  - spinBlackLevelSelectorEnums, 51
  - spinChunkBlackLevelSelectorEnums, 51
  - spinChunkCounterSelectorEnums, 51
  - spinChunkEncoderSelectorEnums, 52
  - spinChunkEncoderStatusEnums, 52
  - spinChunkExposureTimeSelectorEnums, 52
  - spinChunkGainSelectorEnums, 53
  - spinChunkImageComponentEnums, 53
  - spinChunkPixelFormatEnums, 54
  - spinChunkRegionIDEnums, 54
  - spinChunkScan3dCoordinateReferenceSelector↔  
Enums, 54
  - spinChunkScan3dCoordinateSelectorEnums, 55
  - spinChunkScan3dCoordinateSystemEnums, 55
  - spinChunkScan3dCoordinateSystemReference↔  
Enums, 55
  - spinChunkScan3dCoordinateTransformSelector↔  
Enums, 56
  - spinChunkScan3dDistanceUnitEnums, 56
  - spinChunkScan3dOutputModeEnums, 57
  - spinChunkSelectorEnums, 57
  - spinChunkSourceIDEnums, 58
  - spinChunkTimerSelectorEnums, 58
  - spinChunkTransferStreamIDEnums, 59
  - spinCIConfigurationEnums, 59
  - spinCITimeSlotsCountEnums, 59
  - spinColorTransformationSelectorEnums, 60
  - spinColorTransformationValueSelectorEnums, 60
  - spinCounterEventActivationEnums, 61
  - spinCounterEventSourceEnums, 61
  - spinCounterResetActivationEnums, 62
  - spinCounterResetSourceEnums, 62

- spinCounterSelectorEnums, 62
- spinCounterStatusEnums, 63
- spinCounterTriggerActivationEnums, 63
- spinCounterTriggerSourceEnums, 63
- spinCxpConnectionTestModeEnums, 64
- spinCxpLinkConfigurationEnums, 64
- spinCxpLinkConfigurationPreferredEnums, 65
- spinCxpLinkConfigurationStatusEnums, 66
- spinCxpPoCxpStatusEnums, 67
- spinDecimationHorizontalModeEnums, 68
- spinDecimationSelectorEnums, 68
- spinDecimationVerticalModeEnums, 68
- spinDefectCorrectionModeEnums, 68
- spinDeinterlacingEnums, 69
- spinDeviceCharacterSetEnums, 69
- spinDeviceClockSelectorEnums, 69
- spinDeviceConnectionStatusEnums, 70
- spinDeviceIndicatorModeEnums, 70
- spinDeviceLinkHeartbeatModeEnums, 70
- spinDeviceLinkThroughputLimitModeEnums, 72
- spinDevicePowerSupplySelectorEnums, 72
- spinDeviceRegistersEndiannessEnums, 72
- spinDeviceScanTypeEnums, 73
- spinDeviceSerialPortBaudRateEnums, 73
- spinDeviceSerialPortSelectorEnums, 73
- spinDeviceStreamChannelEndiannessEnums, 73
- spinDeviceStreamChannelTypeEnums, 74
- spinDeviceTLTypeEnums, 76
- spinDeviceTapGeometryEnums, 74
- spinDeviceTemperatureSelectorEnums, 75
- spinDeviceTypeEnums, 76
- spinEncoderModeEnums, 76
- spinEncoderOutputModeEnums, 77
- spinEncoderResetActivationEnums, 77
- spinEncoderResetSourceEnums, 78
- spinEncoderSelectorEnums, 79
- spinEncoderSourceAEnums, 79
- spinEncoderSourceBEnums, 79
- spinEncoderStatusEnums, 80
- spinEventNotificationEnums, 80
- spinEventSelectorEnums, 80
- spinExposureActiveModeEnums, 81
- spinExposureAutoEnums, 81
- spinExposureModeEnums, 81
- spinExposureTimeModeEnums, 82
- spinExposureTimeSelectorEnums, 82
- spinFileOpenModeEnums, 83
- spinFileOperationSelectorEnums, 83
- spinFileOperationStatusEnums, 83
- spinFileSelectorEnums, 84
- spinGainAutoBalanceEnums, 84
- spinGainAutoEnums, 84
- spinGainSelectorEnums, 85
- spinGevCCPEnums, 85
- spinGevCurrentPhysicalLinkConfigurationEnums, 85
- spinGevGVCPExtendedStatusCodesSelector↔  
Enums, 85
- spinGevGVSPExtendedIDModeEnums, 86
- spinGevIEEE1588ClockAccuracyEnums, 86
- spinGevIEEE1588ModeEnums, 86
- spinGevIEEE1588StatusEnums, 87
- spinGevIPConfigurationStatusEnums, 87
- spinGevPhysicalLinkConfigurationEnums, 87
- spinGevSupportedOptionSelectorEnums, 88
- spinImageComponentSelectorEnums, 89
- spinImageCompressionJPEGFormatOption↔  
Enums, 89
- spinImageCompressionModeEnums, 90
- spinImageCompressionRateOptionEnums, 90
- spinLUTSelectorEnums, 94
- spinLineFormatEnums, 90
- spinLineInputFilterSelectorEnums, 91
- spinLineModeEnums, 91
- spinLineSelectorEnums, 91
- spinLineSourceEnums, 92
- spinLogicBlockLUTInputActivationEnums, 92
- spinLogicBlockLUTInputSelectorEnums, 93
- spinLogicBlockLUTInputSourceEnums, 93
- spinLogicBlockLUTSelectorEnums, 94
- spinLogicBlockSelectorEnums, 94
- spinPixelColorFilterEnums, 95
- spinPixelFormatEnums, 95
- spinPixelFormatInfoSelectorEnums, 101
- spinPixelSizeEnums, 106
- spinRegionDestinationEnums, 107
- spinRegionModeEnums, 107
- spinRegionSelectorEnums, 108
- spinRgbTransformLightSourceEnums, 108
- spinScan3dCoordinateReferenceSelectorEnums, 108
- spinScan3dCoordinateSelectorEnums, 109
- spinScan3dCoordinateSystemEnums, 109
- spinScan3dCoordinateSystemReferenceEnums, 109
- spinScan3dCoordinateTransformSelectorEnums, 110
- spinScan3dDistanceUnitEnums, 110
- spinScan3dOutputModeEnums, 111
- spinSensorDigitizationTapsEnums, 111
- spinSensorShutterModeEnums, 112
- spinSensorTapsEnums, 112
- spinSequencerConfigurationModeEnums, 113
- spinSequencerConfigurationValidEnums, 113
- spinSequencerModeEnums, 113
- spinSequencerSetValidEnums, 113
- spinSequencerTriggerActivationEnums, 114
- spinSequencerTriggerSourceEnums, 114
- spinSerialPortBaudRateEnums, 114
- spinSerialPortParityEnums, 115
- spinSerialPortSelectorEnums, 115
- spinSerialPortSourceEnums, 116
- spinSerialPortStopBitsEnums, 116
- spinSoftwareSignalSelectorEnums, 116
- spinSourceSelectorEnums, 117
- spinTestPatternEnums, 117

- spinTestPatternGeneratorSelectorEnums, 117
- spinTimerSelectorEnums, 118
- spinTimerStatusEnums, 118
- spinTimerTriggerActivationEnums, 118
- spinTimerTriggerSourceEnums, 119
- spinTransferComponentSelectorEnums, 120
- spinTransferControlModeEnums, 120
- spinTransferOperationModeEnums, 121
- spinTransferQueueModeEnums, 121
- spinTransferSelectorEnums, 121
- spinTransferStatusSelectorEnums, 122
- spinTransferTriggerActivationEnums, 122
- spinTransferTriggerModeEnums, 122
- spinTransferTriggerSelectorEnums, 123
- spinTransferTriggerSourceEnums, 123
- spinTriggerActivationEnums, 124
- spinTriggerModeEnums, 125
- spinTriggerOverlapEnums, 125
- spinTriggerSelectorEnums, 125
- spinTriggerSourceEnums, 125
- spinUserOutputSelectorEnums, 126
- spinUserSetDefaultEnums, 126
- spinUserSetSelectorEnums, 127
- spinWhiteClipSelectorEnums, 127
- CameraList Access, 157
  - spinCameraListAppend, 157
  - spinCameraListClear, 158
  - spinCameraListCreateEmpty, 158
  - spinCameraListDestroy, 159
  - spinCameraListGet, 159
  - spinCameraListGetBySerial, 160
  - spinCameraListGetSize, 160
  - spinCameraListRemove, 161
  - spinCameraListRemoveBySerial, 161
- Chunk data access, 239
  - spinImageChunkDataGetFloatValue, 239
  - spinImageChunkDataGetIntValue, 239
- Chunk Data Structures, 128
- ChunkBlackLevel
  - quickSpin, 355
- ChunkBlackLevelSelector
  - quickSpin, 355
- ChunkCRC
  - quickSpin, 356
- ChunkCounterSelector
  - quickSpin, 356
- ChunkCounterValue
  - quickSpin, 356
- ChunkEnable
  - quickSpin, 356
- ChunkEncoderSelector
  - quickSpin, 356
- ChunkEncoderStatus
  - quickSpin, 356
- ChunkEncoderValue
  - quickSpin, 356
- ChunkExposureEndLineStatusAll
  - quickSpin, 356
- ChunkExposureTime
  - quickSpin, 357
- ChunkExposureTimeSelector
  - quickSpin, 357
- ChunkFrameID
  - quickSpin, 357
- ChunkGain
  - quickSpin, 357
- ChunkGainSelector
  - quickSpin, 357
- ChunkHeight
  - quickSpin, 357
- ChunkImage
  - quickSpin, 357
- ChunkImageComponent
  - quickSpin, 357
- ChunkInferenceConfidence
  - quickSpin, 358
- ChunkInferenceResult
  - quickSpin, 358
- ChunkLinePitch
  - quickSpin, 358
- ChunkLineStatusAll
  - quickSpin, 358
- ChunkModeActive
  - quickSpin, 358
- ChunkOffsetX
  - quickSpin, 358
- ChunkOffsetY
  - quickSpin, 358
- ChunkPartSelector
  - quickSpin, 358
- ChunkPixelDynamicRangeMax
  - quickSpin, 359
- ChunkPixelDynamicRangeMin
  - quickSpin, 359
- ChunkPixelFormat
  - quickSpin, 359
- ChunkRegionID
  - quickSpin, 359
- ChunkScan3dAxisMax
  - quickSpin, 359
- ChunkScan3dAxisMin
  - quickSpin, 359
- ChunkScan3dCoordinateOffset
  - quickSpin, 359
- ChunkScan3dCoordinateReferenceSelector
  - quickSpin, 359
- ChunkScan3dCoordinateReferenceValue
  - quickSpin, 360
- ChunkScan3dCoordinateScale
  - quickSpin, 360
- ChunkScan3dCoordinateSelector
  - quickSpin, 360
- ChunkScan3dCoordinateSystem
  - quickSpin, 360
- ChunkScan3dCoordinateSystemReference
  - quickSpin, 360

- ChunkScan3dCoordinateTransformSelector
  - [quickSpin](#), 360
- ChunkScan3dDistanceUnit
  - [quickSpin](#), 360
- ChunkScan3dInvalidDataFlag
  - [quickSpin](#), 360
- ChunkScan3dInvalidDataValue
  - [quickSpin](#), 361
- ChunkScan3dOutputMode
  - [quickSpin](#), 361
- ChunkScan3dTransformValue
  - [quickSpin](#), 361
- ChunkScanLineSelector
  - [quickSpin](#), 361
- ChunkSelector
  - [quickSpin](#), 361
- ChunkSequencerSetActive
  - [quickSpin](#), 361
- ChunkSerialData
  - [quickSpin](#), 361
- ChunkSerialDataLength
  - [quickSpin](#), 361
- ChunkSerialReceiveOverflow
  - [quickSpin](#), 362
- ChunkSourceID
  - [quickSpin](#), 362
- ChunkStreamChannelID
  - [quickSpin](#), 362
- ChunkTimerSelector
  - [quickSpin](#), 362
- ChunkTimerValue
  - [quickSpin](#), 362
- ChunkTimestamp
  - [quickSpin](#), 362
- ChunkTimestampLatchValue
  - [quickSpin](#), 362
- ChunkTransferBlockID
  - [quickSpin](#), 362
- ChunkTransferQueueCurrentBlockCount
  - [quickSpin](#), 363
- ChunkTransferStreamID
  - [quickSpin](#), 363
- ChunkWidth
  - [quickSpin](#), 363
- CIConfiguration
  - [quickSpin](#), 363
- CITimeSlotsCount
  - [quickSpin](#), 363
- ColorTransformationEnable
  - [quickSpin](#), 363
- ColorTransformationSelector
  - [quickSpin](#), 363
- ColorTransformationValue
  - [quickSpin](#), 363
- ColorTransformationValueSelector
  - [quickSpin](#), 364
- compression
  - [spinTIFFOption](#), 464
- compressionLevel
  - [spinPNGOption](#), 462
- CompressionRatio
  - [quickSpin](#), 364
- CounterDelay
  - [quickSpin](#), 364
- CounterDuration
  - [quickSpin](#), 364
- CounterEventActivation
  - [quickSpin](#), 364
- CounterEventSource
  - [quickSpin](#), 364
- CounterReset
  - [quickSpin](#), 364
- CounterResetActivation
  - [quickSpin](#), 364
- CounterResetSource
  - [quickSpin](#), 365
- CounterSelector
  - [quickSpin](#), 365
- CounterStatus
  - [quickSpin](#), 365
- CounterTriggerActivation
  - [quickSpin](#), 365
- CounterTriggerSource
  - [quickSpin](#), 365
- CounterValue
  - [quickSpin](#), 365
- CounterValueAtReset
  - [quickSpin](#), 365
- CxpConnectionSelector
  - [quickSpin](#), 365
- CxpConnectionTestErrorCount
  - [quickSpin](#), 366
- CxpConnectionTestMode
  - [quickSpin](#), 366
- CxpConnectionTestPacketCount
  - [quickSpin](#), 366
- CxpLinkConfiguration
  - [quickSpin](#), 366
- CxpLinkConfigurationPreferred
  - [quickSpin](#), 366
- CxpLinkConfigurationStatus
  - [quickSpin](#), 366
- CxpPoCxpAuto
  - [quickSpin](#), 366
- CxpPoCxpStatus
  - [quickSpin](#), 366
- CxpPoCxpTripReset
  - [quickSpin](#), 367
- CxpPoCxpTurnOff
  - [quickSpin](#), 367
- DecimationHorizontal
  - [quickSpin](#), 367
- DecimationHorizontalMode
  - [quickSpin](#), 367
- DecimationSelector
  - [quickSpin](#), 367

- DecimationVertical
  - [quickSpin](#), 367
- DecimationVerticalMode
  - [quickSpin](#), 367
- DefectCorrectStaticEnable
  - [quickSpin](#), 368
- DefectCorrectionMode
  - [quickSpin](#), 367
- DefectTableApply
  - [quickSpin](#), 368
- DefectTableCoordinateX
  - [quickSpin](#), 368
- DefectTableCoordinateY
  - [quickSpin](#), 368
- DefectTableFactoryRestore
  - [quickSpin](#), 368
- DefectTableIndex
  - [quickSpin](#), 368
- DefectTablePixelCount
  - [quickSpin](#), 368
- DefectTableSave
  - [quickSpin](#), 368
- Deinterlacing
  - [quickSpin](#), 369
- Device Event Data Access, 233
  - [spinDeviceEventGetId](#), 233
  - [spinDeviceEventGetName](#), 234
  - [spinDeviceEventGetPayloadData](#), 234
  - [spinDeviceEventGetPayloadDataSize](#), 235
- DeviceAccessStatus
  - [quickSpinTLDevice](#), 433
  - [quickSpinTLInterface](#), 439
- DeviceAddress
  - [actionCommandResult](#), 335
- DeviceCharacterSet
  - [quickSpin](#), 369
- DeviceClockFrequency
  - [quickSpin](#), 369
- DeviceClockSelector
  - [quickSpin](#), 369
- DeviceConnectionSelector
  - [quickSpin](#), 369
- DeviceConnectionSpeed
  - [quickSpin](#), 369
- DeviceConnectionStatus
  - [quickSpin](#), 369
- DeviceCount
  - [quickSpinTLInterface](#), 439
- DeviceCurrentSpeed
  - [quickSpinTLDevice](#), 433
- DeviceDisplayName
  - [quickSpinTLDevice](#), 433
- DeviceDriverVersion
  - [quickSpinTLDevice](#), 433
- DeviceEndiannessMechanism
  - [quickSpinTLDevice](#), 433
- DeviceEventChannelCount
  - [quickSpin](#), 369
- DeviceFamilyName
  - [quickSpin](#), 370
- DeviceFeaturePersistenceEnd
  - [quickSpin](#), 370
- DeviceFeaturePersistenceStart
  - [quickSpin](#), 370
- DeviceFirmwareVersion
  - [quickSpin](#), 370
- DeviceGenCPVersionMajor
  - [quickSpin](#), 370
- DeviceGenCPVersionMinor
  - [quickSpin](#), 370
- DeviceID
  - [quickSpin](#), 370
  - [quickSpinTLDevice](#), 433
  - [quickSpinTLInterface](#), 439
- DeviceIndicatorMode
  - [quickSpin](#), 370
- DeviceInstanceId
  - [quickSpinTLDevice](#), 433
- DevicesUpdater
  - [quickSpinTLDevice](#), 433
- DeviceLinkBandwidthReserve
  - [quickSpin](#), 371
- DeviceLinkCommandTimeout
  - [quickSpin](#), 371
- DeviceLinkConnectionCount
  - [quickSpin](#), 371
- DeviceLinkCurrentThroughput
  - [quickSpin](#), 371
- DeviceLinkHeartbeatMode
  - [quickSpin](#), 371
- DeviceLinkHeartbeatTimeout
  - [quickSpin](#), 371
- DeviceLinkSelector
  - [quickSpin](#), 371
- DeviceLinkSpeed
  - [quickSpin](#), 371
  - [quickSpinTLDevice](#), 434
- DeviceLinkThroughputLimit
  - [quickSpin](#), 372
- DeviceLinkThroughputLimitMode
  - [quickSpin](#), 372
- DeviceLocation
  - [quickSpinTLDevice](#), 434
- DeviceManifestEntrySelector
  - [quickSpin](#), 372
- DeviceManifestPrimaryURL
  - [quickSpin](#), 372
- DeviceManifestSchemaMajorVersion
  - [quickSpin](#), 372
- DeviceManifestSchemaMinorVersion
  - [quickSpin](#), 372
- DeviceManifestSecondaryURL
  - [quickSpin](#), 372
- DeviceManifestXMLMajorVersion
  - [quickSpin](#), 372
- DeviceManifestXMLMinorVersion



- quickSpin, [373](#)
- DeviceManifestXMLSubMinorVersion
  - quickSpin, [373](#)
- DeviceManufacturerInfo
  - quickSpin, [373](#)
- DeviceMaxThroughput
  - quickSpin, [373](#)
- DeviceModelName
  - quickSpin, [373](#)
  - quickSpinTLDevice, [434](#)
  - quickSpinTLInterface, [439](#)
- DeviceMulticastMonitorMode
  - quickSpinTLDevice, [434](#)
- DevicePowerSupplySelector
  - quickSpin, [373](#)
- DeviceRegistersCheck
  - quickSpin, [373](#)
- DeviceRegistersEndianness
  - quickSpin, [373](#)
- DeviceRegistersStreamingEnd
  - quickSpin, [374](#)
- DeviceRegistersStreamingStart
  - quickSpin, [374](#)
- DeviceRegistersValid
  - quickSpin, [374](#)
- DeviceReset
  - quickSpin, [374](#)
- DeviceSFNCVersionMajor
  - quickSpin, [375](#)
- DeviceSFNCVersionMinor
  - quickSpin, [375](#)
- DeviceSFNCVersionSubMinor
  - quickSpin, [375](#)
- DeviceScanType
  - quickSpin, [374](#)
- DeviceSelector
  - quickSpinTLInterface, [439](#)
- DeviceSerialNumber
  - quickSpin, [374](#)
  - quickSpinTLDevice, [434](#)
- DeviceSerialPortBaudRate
  - quickSpin, [374](#)
- DeviceSerialPortSelector
  - quickSpin, [374](#)
- DeviceStreamChannelCount
  - quickSpin, [375](#)
- DeviceStreamChannelEndianness
  - quickSpin, [375](#)
- DeviceStreamChannelLink
  - quickSpin, [375](#)
- DeviceStreamChannelPacketSize
  - quickSpin, [375](#)
- DeviceStreamChannelSelector
  - quickSpin, [375](#)
- DeviceStreamChannelType
  - quickSpin, [376](#)
- DeviceTLType
  - quickSpin, [376](#)
- DeviceTLVersionMajor
  - quickSpin, [376](#)
- DeviceTLVersionMinor
  - quickSpin, [376](#)
- DeviceTLVersionSubMinor
  - quickSpin, [376](#)
- DeviceTapGeometry
  - quickSpin, [376](#)
- DeviceTemperature
  - quickSpin, [376](#)
- DeviceTemperatureSelector
  - quickSpin, [376](#)
- DeviceType
  - quickSpin, [377](#)
  - quickSpinTLDevice, [434](#)
- DeviceU3VProtocol
  - quickSpinTLDevice, [434](#)
- DeviceUnlock
  - quickSpinTLInterface, [439](#)
- DeviceUpdateList
  - quickSpinTLInterface, [439](#)
- DeviceUptime
  - quickSpin, [377](#)
- DeviceUserID
  - quickSpin, [377](#)
  - quickSpinTLDevice, [434](#)
- DeviceVendorName
  - quickSpin, [377](#)
  - quickSpinTLDevice, [435](#)
  - quickSpinTLInterface, [440](#)
- DeviceVersion
  - quickSpin, [377](#)
  - quickSpinTLDevice, [435](#)
- [doc/Doxygen/spindocs/C/Licensing.dox](#), [465](#)
- [doc/Doxygen/spindocs/C/MainPage.dox](#), [465](#)
- EncoderDivider
  - quickSpin, [377](#)
- EncoderMode
  - quickSpin, [377](#)
- EncoderOutputMode
  - quickSpin, [377](#)
- EncoderReset
  - quickSpin, [378](#)
- EncoderResetActivation
  - quickSpin, [378](#)
- EncoderResetSource
  - quickSpin, [378](#)
- EncoderSelector
  - quickSpin, [378](#)
- EncoderSourceA
  - quickSpin, [378](#)
- EncoderSourceB
  - quickSpin, [378](#)
- EncoderStatus
  - quickSpin, [378](#)
- EncoderTimeout
  - quickSpin, [378](#)
- EncoderValue

- quickSpin, [379](#)
- EncoderValueAtReset
  - quickSpin, [379](#)
- EnumerateGEVInterfaces
  - quickSpinTLSystem, [448](#)
- EnumerationCount
  - quickSpin, [379](#)
- Error Handling, [134](#)
  - spinErrorGetLast, [134](#)
  - spinErrorGetLastBuildDate, [135](#)
  - spinErrorGetLastBuildTime, [135](#)
  - spinErrorGetLastFileName, [136](#)
  - spinErrorGetLastFullMessage, [136](#)
  - spinErrorGetLastFunctionName, [137](#)
  - spinErrorGetLastLineNumber, [137](#)
  - spinErrorGetLastMessage, [138](#)
- Event Access, [212](#)
  - spinArrivalEventCreate, [212](#)
  - spinArrivalEventDestroy, [213](#)
  - spinDeviceEventCreate, [213](#)
  - spinDeviceEventDestroy, [214](#)
  - spinImageEventCreate, [214](#)
  - spinImageEventDestroy, [215](#)
  - spinInterfaceEventCreate, [215](#)
  - spinInterfaceEventDestroy, [216](#)
  - spinLogEventCreate, [216](#)
  - spinLogEventDestroy, [217](#)
  - spinRemovalEventCreate, [217](#)
  - spinRemovalEventDestroy, [218](#)
- EventAcquisitionEnd
  - quickSpin, [379](#)
- EventAcquisitionEndFrameID
  - quickSpin, [379](#)
- EventAcquisitionEndTimestamp
  - quickSpin, [379](#)
- EventAcquisitionError
  - quickSpin, [379](#)
- EventAcquisitionErrorFrameID
  - quickSpin, [379](#)
- EventAcquisitionErrorTimestamp
  - quickSpin, [380](#)
- EventAcquisitionStart
  - quickSpin, [380](#)
- EventAcquisitionStartFrameID
  - quickSpin, [380](#)
- EventAcquisitionStartTimestamp
  - quickSpin, [380](#)
- EventAcquisitionTransferEnd
  - quickSpin, [380](#)
- EventAcquisitionTransferEndFrameID
  - quickSpin, [380](#)
- EventAcquisitionTransferEndTimestamp
  - quickSpin, [380](#)
- EventAcquisitionTransferStart
  - quickSpin, [380](#)
- EventAcquisitionTransferStartFrameID
  - quickSpin, [381](#)
- EventAcquisitionTransferStartTimestamp
  - quickSpin, [381](#)
- EventAcquisitionTrigger
  - quickSpin, [381](#)
- EventAcquisitionTriggerFrameID
  - quickSpin, [381](#)
- EventAcquisitionTriggerTimestamp
  - quickSpin, [381](#)
- EventActionLate
  - quickSpin, [381](#)
- EventActionLateFrameID
  - quickSpin, [381](#)
- EventActionLateTimestamp
  - quickSpin, [381](#)
- EventCounter0End
  - quickSpin, [382](#)
- EventCounter0EndFrameID
  - quickSpin, [382](#)
- EventCounter0EndTimestamp
  - quickSpin, [382](#)
- EventCounter0Start
  - quickSpin, [382](#)
- EventCounter0StartFrameID
  - quickSpin, [382](#)
- EventCounter0StartTimestamp
  - quickSpin, [382](#)
- EventCounter1End
  - quickSpin, [382](#)
- EventCounter1EndFrameID
  - quickSpin, [382](#)
- EventCounter1EndTimestamp
  - quickSpin, [383](#)
- EventCounter1Start
  - quickSpin, [383](#)
- EventCounter1StartFrameID
  - quickSpin, [383](#)
- EventCounter1StartTimestamp
  - quickSpin, [383](#)
- EventEncoder0Restarted
  - quickSpin, [383](#)
- EventEncoder0RestartedFrameID
  - quickSpin, [383](#)
- EventEncoder0RestartedTimestamp
  - quickSpin, [383](#)
- EventEncoder0Stopped
  - quickSpin, [383](#)
- EventEncoder0StoppedFrameID
  - quickSpin, [384](#)
- EventEncoder0StoppedTimestamp
  - quickSpin, [384](#)
- EventEncoder1Restarted
  - quickSpin, [384](#)
- EventEncoder1RestartedFrameID
  - quickSpin, [384](#)
- EventEncoder1RestartedTimestamp
  - quickSpin, [384](#)
- EventEncoder1Stopped
  - quickSpin, [384](#)
- EventEncoder1StoppedFrameID
  - quickSpin, [384](#)



- quickSpin, [384](#)
- EventEncoder1 StoppedTimestamp
  - quickSpin, [384](#)
- EventError
  - quickSpin, [385](#)
- EventErrorCode
  - quickSpin, [385](#)
- EventErrorFrameID
  - quickSpin, [385](#)
- EventErrorTimestamp
  - quickSpin, [385](#)
- EventExposureEnd
  - quickSpin, [385](#)
- EventExposureEndFrameID
  - quickSpin, [385](#)
- EventExposureEndTimestamp
  - quickSpin, [385](#)
- EventExposureStart
  - quickSpin, [385](#)
- EventExposureStartFrameID
  - quickSpin, [386](#)
- EventExposureStartTimestamp
  - quickSpin, [386](#)
- EventFrameBurstEnd
  - quickSpin, [386](#)
- EventFrameBurstEndFrameID
  - quickSpin, [386](#)
- EventFrameBurstEndTimestamp
  - quickSpin, [386](#)
- EventFrameBurstStart
  - quickSpin, [386](#)
- EventFrameBurstStartFrameID
  - quickSpin, [386](#)
- EventFrameBurstStartTimestamp
  - quickSpin, [386](#)
- EventFrameEnd
  - quickSpin, [387](#)
- EventFrameEndFrameID
  - quickSpin, [387](#)
- EventFrameEndTimestamp
  - quickSpin, [387](#)
- EventFrameStart
  - quickSpin, [387](#)
- EventFrameStartFrameID
  - quickSpin, [387](#)
- EventFrameStartTimestamp
  - quickSpin, [387](#)
- EventFrameTransferEnd
  - quickSpin, [387](#)
- EventFrameTransferEndFrameID
  - quickSpin, [387](#)
- EventFrameTransferEndTimestamp
  - quickSpin, [388](#)
- EventFrameTransferStart
  - quickSpin, [388](#)
- EventFrameTransferStartFrameID
  - quickSpin, [388](#)
- EventFrameTransferStartTimestamp
  - quickSpin, [388](#)
- EventFrameTrigger
  - quickSpin, [388](#)
- EventFrameTriggerFrameID
  - quickSpin, [388](#)
- EventFrameTriggerTimestamp
  - quickSpin, [388](#)
- EventLine0AnyEdge
  - quickSpin, [388](#)
- EventLine0AnyEdgeFrameID
  - quickSpin, [389](#)
- EventLine0AnyEdgeTimestamp
  - quickSpin, [389](#)
- EventLine0FallingEdge
  - quickSpin, [389](#)
- EventLine0FallingEdgeFrameID
  - quickSpin, [389](#)
- EventLine0FallingEdgeTimestamp
  - quickSpin, [389](#)
- EventLine0RisingEdge
  - quickSpin, [389](#)
- EventLine0RisingEdgeFrameID
  - quickSpin, [389](#)
- EventLine0RisingEdgeTimestamp
  - quickSpin, [389](#)
- EventLine1AnyEdge
  - quickSpin, [390](#)
- EventLine1AnyEdgeFrameID
  - quickSpin, [390](#)
- EventLine1AnyEdgeTimestamp
  - quickSpin, [390](#)
- EventLine1FallingEdge
  - quickSpin, [390](#)
- EventLine1FallingEdgeFrameID
  - quickSpin, [390](#)
- EventLine1FallingEdgeTimestamp
  - quickSpin, [390](#)
- EventLine1RisingEdge
  - quickSpin, [390](#)
- EventLine1RisingEdgeFrameID
  - quickSpin, [390](#)
- EventLine1RisingEdgeTimestamp
  - quickSpin, [391](#)
- EventLinkSpeedChange
  - quickSpin, [391](#)
- EventLinkSpeedChangeFrameID
  - quickSpin, [391](#)
- EventLinkSpeedChangeTimestamp
  - quickSpin, [391](#)
- EventLinkTrigger0
  - quickSpin, [391](#)
- EventLinkTrigger0FrameID
  - quickSpin, [391](#)
- EventLinkTrigger0Timestamp
  - quickSpin, [391](#)
- EventLinkTrigger1
  - quickSpin, [391](#)
- EventLinkTrigger1FrameID
  - quickSpin, [391](#)

- quickSpin, [392](#)
- EventLinkTrigger1Timestamp
  - quickSpin, [392](#)
- EventNotification
  - quickSpin, [392](#)
- EventSelector
  - quickSpin, [392](#)
- EventSequencerSetChange
  - quickSpin, [392](#)
- EventSequencerSetChangeFrameID
  - quickSpin, [392](#)
- EventSequencerSetChangeTimestamp
  - quickSpin, [392](#)
- EventSerialData
  - quickSpin, [392](#)
- EventSerialDataLength
  - quickSpin, [393](#)
- EventSerialPortReceive
  - quickSpin, [393](#)
- EventSerialPortReceiveTimestamp
  - quickSpin, [393](#)
- EventSerialReceiveOverflow
  - quickSpin, [393](#)
- EventStream0TransferBlockEnd
  - quickSpin, [393](#)
- EventStream0TransferBlockEndFrameID
  - quickSpin, [393](#)
- EventStream0TransferBlockEndTimestamp
  - quickSpin, [393](#)
- EventStream0TransferBlockStart
  - quickSpin, [393](#)
- EventStream0TransferBlockStartFrameID
  - quickSpin, [394](#)
- EventStream0TransferBlockStartTimestamp
  - quickSpin, [394](#)
- EventStream0TransferBlockTrigger
  - quickSpin, [394](#)
- EventStream0TransferBlockTriggerFrameID
  - quickSpin, [394](#)
- EventStream0TransferBlockTriggerTimestamp
  - quickSpin, [394](#)
- EventStream0TransferBurstEnd
  - quickSpin, [394](#)
- EventStream0TransferBurstEndFrameID
  - quickSpin, [394](#)
- EventStream0TransferBurstEndTimestamp
  - quickSpin, [394](#)
- EventStream0TransferBurstStart
  - quickSpin, [395](#)
- EventStream0TransferBurstStartFrameID
  - quickSpin, [395](#)
- EventStream0TransferBurstStartTimestamp
  - quickSpin, [395](#)
- EventStream0TransferEnd
  - quickSpin, [395](#)
- EventStream0TransferEndFrameID
  - quickSpin, [395](#)
- EventStream0TransferEndTimestamp
  - quickSpin, [395](#)
- EventStream0TransferOverflow
  - quickSpin, [395](#)
- EventStream0TransferOverflowFrameID
  - quickSpin, [395](#)
- EventStream0TransferOverflowTimestamp
  - quickSpin, [396](#)
- EventStream0TransferPause
  - quickSpin, [396](#)
- EventStream0TransferPauseFrameID
  - quickSpin, [396](#)
- EventStream0TransferPauseTimestamp
  - quickSpin, [396](#)
- EventStream0TransferResume
  - quickSpin, [396](#)
- EventStream0TransferResumeFrameID
  - quickSpin, [396](#)
- EventStream0TransferResumeTimestamp
  - quickSpin, [396](#)
- EventStream0TransferStart
  - quickSpin, [396](#)
- EventStream0TransferStartFrameID
  - quickSpin, [397](#)
- EventStream0TransferStartTimestamp
  - quickSpin, [397](#)
- EventTest
  - quickSpin, [397](#)
- EventTestTimestamp
  - quickSpin, [397](#)
- EventTimer0End
  - quickSpin, [397](#)
- EventTimer0EndFrameID
  - quickSpin, [397](#)
- EventTimer0EndTimestamp
  - quickSpin, [397](#)
- EventTimer0Start
  - quickSpin, [397](#)
- EventTimer0StartFrameID
  - quickSpin, [398](#)
- EventTimer0StartTimestamp
  - quickSpin, [398](#)
- EventTimer1End
  - quickSpin, [398](#)
- EventTimer1EndFrameID
  - quickSpin, [398](#)
- EventTimer1EndTimestamp
  - quickSpin, [398](#)
- EventTimer1Start
  - quickSpin, [398](#)
- EventTimer1StartFrameID
  - quickSpin, [398](#)
- EventTimer1StartTimestamp
  - quickSpin, [398](#)
- ExposureActiveMode
  - quickSpin, [399](#)
- ExposureAuto
  - quickSpin, [399](#)
- ExposureMode

- quickSpin, [399](#)
- ExposureTime
  - quickSpin, [399](#)
- ExposureTimeMode
  - quickSpin, [399](#)
- ExposureTimeSelector
  - quickSpin, [399](#)
- FactoryReset
  - quickSpin, [399](#)
- False
  - Spinnaker C Definitions, [12](#)
- FileAccessBuffer
  - quickSpin, [399](#)
- FileAccessLength
  - quickSpin, [400](#)
- FileAccessOffset
  - quickSpin, [400](#)
- FileOpenMode
  - quickSpin, [400](#)
- FileOperationExecute
  - quickSpin, [400](#)
- FileOperationResult
  - quickSpin, [400](#)
- FileOperationSelector
  - quickSpin, [400](#)
- FileOperationStatus
  - quickSpin, [400](#)
- FileSelector
  - quickSpin, [400](#)
- FileSize
  - quickSpin, [401](#)
- FilterDriverStatus
  - quickSpinTLInterface, [440](#)
- frameRate
  - spinAVIOption, [448](#)
  - spinH264Option, [456](#)
  - spinMJPGOption, [460](#)
- GUIXMLLocation
  - quickSpinTLDevice, [437](#)
- GUIXMLPath
  - quickSpinTLDevice, [437](#)
- Gain
  - quickSpin, [401](#)
- GainAuto
  - quickSpin, [401](#)
- GainAutoBalance
  - quickSpin, [401](#)
- GainSelector
  - quickSpin, [401](#)
- Gamma
  - quickSpin, [401](#)
- GammaEnable
  - quickSpin, [401](#)
- GenICamXMLLocation
  - quickSpinTLDevice, [435](#)
- GenICamXMLPath
  - quickSpinTLDevice, [435](#)
- GevActionDeviceKey
  - quickSpinTLInterface, [440](#)
- GevActionGroupKey
  - quickSpinTLInterface, [440](#)
- GevActionGroupMask
  - quickSpinTLInterface, [440](#)
- GevActionTime
  - quickSpinTLInterface, [440](#)
- GevActiveLinkCount
  - quickSpin, [401](#)
- GevCCP
  - quickSpin, [402](#)
  - quickSpinTLDevice, [435](#)
- GevCurrentDefaultGateway
  - quickSpin, [402](#)
- GevCurrentIPAddress
  - quickSpin, [402](#)
- GevCurrentIPConfigurationDHCP
  - quickSpin, [402](#)
- GevCurrentIPConfigurationLLA
  - quickSpin, [402](#)
- GevCurrentIPConfigurationPersistentIP
  - quickSpin, [402](#)
- GevCurrentPhysicalLinkConfiguration
  - quickSpin, [402](#)
- GevCurrentSubnetMask
  - quickSpin, [402](#)
- GevDeviceDiscoverMaximumPacketSize
  - quickSpinTLDevice, [435](#)
- GevDeviceForceIP
  - quickSpinTLDevice, [435](#)
- GevDeviceGateway
  - quickSpinTLDevice, [435](#)
- GevDeviceIPAddress
  - quickSpinTLDevice, [436](#)
  - quickSpinTLInterface, [440](#)
- GevDevicesWrongSubnet
  - quickSpinTLDevice, [436](#)
- GevDeviceMACAddress
  - quickSpinTLDevice, [436](#)
  - quickSpinTLInterface, [440](#)
- GevDeviceMaximumPacketSize
  - quickSpinTLDevice, [436](#)
- GevDeviceMaximumRetryCount
  - quickSpinTLDevice, [436](#)
- GevDeviceModelsBigEndian
  - quickSpinTLDevice, [436](#)
- GevDevicePort
  - quickSpinTLDevice, [436](#)
- GevDeviceReadAndWriteTimeout
  - quickSpinTLDevice, [436](#)
- GevDeviceSubnetMask
  - quickSpinTLDevice, [437](#)
  - quickSpinTLInterface, [441](#)
- GevDiscoveryAckDelay
  - quickSpin, [403](#)
- GevFailedPacketCount
  - quickSpinTLStream, [444](#)

- GevFirstURL
  - [quickSpin, 403](#)
- GevGVCPExtendedStatusCodes
  - [quickSpin, 403](#)
- GevGVCPExtendedStatusCodesSelector
  - [quickSpin, 403](#)
- GevGVCPHeartbeatDisable
  - [quickSpin, 403](#)
- GevGVCPPendingAck
  - [quickSpin, 403](#)
- GevGVCPPendingTimeout
  - [quickSpin, 403](#)
- GevGVSPExtendedIDMode
  - [quickSpin, 403](#)
- GevHeartbeatTimeout
  - [quickSpin, 404](#)
- GevIEEE1588
  - [quickSpin, 404](#)
- GevIEEE1588ClockAccuracy
  - [quickSpin, 404](#)
- GevIEEE1588Mode
  - [quickSpin, 404](#)
- GevIEEE1588Status
  - [quickSpin, 404](#)
- GevIPConfigurationStatus
  - [quickSpin, 404](#)
- GevInterfaceGateway
  - [quickSpinTLInterface, 441](#)
- GevInterfaceIPAddress
  - [quickSpinTLInterface, 441](#)
- GevInterfaceMACAddress
  - [quickSpinTLInterface, 441](#)
- GevInterfaceMTU
  - [quickSpinTLInterface, 441](#)
- GevInterfaceReceiveLinkSpeed
  - [quickSpinTLInterface, 441](#)
- GevInterfaceSelector
  - [quickSpin, 404](#)
- GevInterfaceSubnetMask
  - [quickSpinTLInterface, 441](#)
- GevInterfaceTransmitLinkSpeed
  - [quickSpinTLInterface, 441](#)
- GevMACAddress
  - [quickSpin, 404](#)
- GevMCDA
  - [quickSpin, 405](#)
- GevMCPHostPort
  - [quickSpin, 405](#)
- GevMCRC
  - [quickSpin, 405](#)
- GevMCSP
  - [quickSpin, 405](#)
- GevMCTT
  - [quickSpin, 405](#)
- GevMaximumNumberResendBuffers
  - [quickSpinTLStream, 444](#)
- GevMaximumNumberResendRequests
  - [quickSpinTLStream, 445](#)
- GevNumberOfInterfaces
  - [quickSpin, 405](#)
- GevPAUSEFrameReception
  - [quickSpin, 405](#)
- GevPAUSEFrameTransmission
  - [quickSpin, 405](#)
- GevPacketResendMode
  - [quickSpinTLStream, 445](#)
- GevPacketResendTimeout
  - [quickSpinTLStream, 445](#)
- GevPersistentDefaultGateway
  - [quickSpin, 406](#)
- GevPersistentIPAddress
  - [quickSpin, 406](#)
- GevPersistentSubnetMask
  - [quickSpin, 406](#)
- GevPhysicalLinkConfiguration
  - [quickSpin, 406](#)
- GevPrimaryApplicationIPAddress
  - [quickSpin, 406](#)
- GevPrimaryApplicationSocket
  - [quickSpin, 406](#)
- GevPrimaryApplicationSwitchoverKey
  - [quickSpin, 406](#)
- GevResendPacketCount
  - [quickSpinTLStream, 445](#)
- GevResendRequestCount
  - [quickSpinTLStream, 445](#)
- GevSCCFGAllInTransmission
  - [quickSpin, 406](#)
- GevSCCFGExtendedChunkData
  - [quickSpin, 407](#)
- GevSCCFGPacketResendDestination
  - [quickSpin, 407](#)
- GevSCCFGUnconditionalStreaming
  - [quickSpin, 407](#)
- GevSCDA
  - [quickSpin, 407](#)
- GevSCPDDirection
  - [quickSpin, 407](#)
- GevSCPHostPort
  - [quickSpin, 407](#)
- GevSCPIInterfaceIndex
  - [quickSpin, 407](#)
- GevSCPSBigEndian
  - [quickSpin, 408](#)
- GevSCPSDoNotFragment
  - [quickSpin, 408](#)
- GevSCPSFireTestPacket
  - [quickSpin, 408](#)
- GevSCPSPacketSize
  - [quickSpin, 408](#)
- GevSCPD
  - [quickSpin, 407](#)
- GevSCSP
  - [quickSpin, 408](#)
- GevSCZoneConfigurationLock
  - [quickSpin, 408](#)

- GevSCZoneCount
  - quickSpin, [408](#)
- GevSCZoneDirectionAll
  - quickSpin, [408](#)
- GevSecondURL
  - quickSpin, [409](#)
- GevStreamChannelSelector
  - quickSpin, [409](#)
- GevSupportedOption
  - quickSpin, [409](#)
- GevSupportedOptionSelector
  - quickSpin, [409](#)
- GevTimestampTickFrequency
  - quickSpin, [409](#)
- GevTotalPacketCount
  - quickSpinTLStream, [445](#)
- GevVersionMajor
  - quickSpinTLDevice, [437](#)
- GevVersionMinor
  - quickSpinTLDevice, [437](#)
- GuiXmlManifestAddress
  - quickSpin, [409](#)
- Height
  - quickSpin, [409](#)
- height
  - spinH264Option, [456](#)
- HeightMax
  - quickSpin, [409](#)
- HostAdapterDriverVersion
  - quickSpinTLInterface, [442](#)
- HostAdapterName
  - quickSpinTLInterface, [442](#)
- HostAdapterVendor
  - quickSpinTLInterface, [442](#)
- IBoolean Access, [297](#)
  - spinBooleanGetValue, [297](#)
  - spinBooleanSetValue, [298](#)
- ICategory Access, [301](#)
  - spinCategoryGetFeatureByIndex, [301](#)
  - spinCategoryGetNumFeatures, [302](#)
- ICommand Access, [299](#)
  - spinCommandExecute, [299](#)
  - spinCommandIsDone, [300](#)
- IEnumEntry Access, [294](#)
  - spinEnumerationEntryGetEnumValue, [294](#)
  - spinEnumerationEntryGetIntValue, [295](#)
  - spinEnumerationEntryGetSymbolic, [295](#)
- IEnumeration Access, [290](#)
  - spinEnumerationGetCurrentEntry, [290](#)
  - spinEnumerationGetEntryByIndex, [291](#)
  - spinEnumerationGetEntryByName, [291](#)
  - spinEnumerationGetNumEntries, [292](#)
  - spinEnumerationSetEnumValue, [292](#)
  - spinEnumerationSetIntValue, [293](#)
- IFloat Access, [285](#)
  - spinFloatGetMax, [285](#)
  - spinFloatGetMin, [286](#)
  - spinFloatGetRepresentation, [286](#)
  - spinFloatGetUnit, [287](#)
  - spinFloatGetValue, [287](#)
  - spinFloatGetValueEx, [288](#)
  - spinFloatSetValue, [288](#)
  - spinFloatSetValueEx, [289](#)
- IInteger Access, [280](#)
  - spinIntegerGetInc, [280](#)
  - spinIntegerGetMax, [281](#)
  - spinIntegerGetMin, [281](#)
  - spinIntegerGetRepresentation, [282](#)
  - spinIntegerGetValue, [282](#)
  - spinIntegerGetValueEx, [283](#)
  - spinIntegerSetValue, [283](#)
  - spinIntegerSetValueEx, [284](#)
- IRegister Access, [303](#)
  - spinRegisterGet, [303](#)
  - spinRegisterGetAddress, [304](#)
  - spinRegisterGetEx, [304](#)
  - spinRegisterGetLength, [305](#)
  - spinRegisterSet, [306](#)
  - spinRegisterSetEx, [306](#)
  - spinRegisterSetReference, [307](#)
- IValue Access, [273](#)
  - spinNodeFromString, [273](#)
  - spinNodeFromStringEx, [274](#)
  - spinNodeToString, [274](#)
  - spinNodeToStringEx, [275](#)
- Image Access, [183](#)
  - spinImageCalculateStatistics, [185](#)
  - spinImageCheckCRC, [186](#)
  - spinImageConvert, [186](#)
  - spinImageConvertEx, [187](#)
  - spinImageCreate, [187](#)
  - spinImageCreateEmpty, [188](#)
  - spinImageCreateEx, [188](#)
  - spinImageDeepCopy, [189](#)
  - spinImageDestroy, [189](#)
  - spinImageGetBitsPerPixel, [190](#)
  - spinImageGetBufferSize, [190](#)
  - spinImageGetChunkLayoutID, [191](#)
  - spinImageGetColorProcessing, [191](#)
  - spinImageGetData, [192](#)
  - spinImageGetDefaultColorProcessing, [192](#)
  - spinImageGetFrameID, [192](#)
  - spinImageGetHeight, [193](#)
  - spinImageGetID, [193](#)
  - spinImageGetOffsetX, [194](#)
  - spinImageGetOffsetY, [194](#)
  - spinImageGetPaddingX, [195](#)
  - spinImageGetPaddingY, [195](#)
  - spinImageGetPayloadType, [196](#)
  - spinImageGetPixelFormat, [196](#)
  - spinImageGetPixelFormatName, [197](#)
  - spinImageGetPrivateData, [197](#)
  - spinImageGetSize, [198](#)
  - spinImageGetStatus, [198](#)
  - spinImageGetStatusDescription, [199](#)

- spinImageGetStride, [199](#)
- spinImageGetTLPayloadType, [200](#)
- spinImageGetTLPixelFormat, [201](#)
- spinImageGetTLPixelFormatNamespace, [201](#)
- spinImageGetTimeStamp, [200](#)
- spinImageGetValidPayloadSize, [202](#)
- spinImageGetWidth, [202](#)
- spinImageHasCRC, [203](#)
- spinImageIsIncomplete, [203](#)
- spinImageRelease, [204](#)
- spinImageReset, [204](#)
- spinImageResetEx, [205](#)
- spinImageSave, [206](#)
- spinImageSaveBmp, [206](#)
- spinImageSaveFromExt, [207](#)
- spinImageSaveJpeg, [207](#)
- spinImageSaveJpg2, [208](#)
- spinImageSavePgm, [208](#)
- spinImageSavePng, [209](#)
- spinImageSavePpm, [209](#)
- spinImageSaveTiff, [210](#)
- spinImageSetDefaultColorProcessing, [210](#)
- ImageComponentEnable
  - quickSpin, [410](#)
- ImageComponentSelector
  - quickSpin, [410](#)
- ImageCompressionBitrate
  - quickSpin, [410](#)
- ImageCompressionJPEGFormatOption
  - quickSpin, [410](#)
- ImageCompressionMode
  - quickSpin, [410](#)
- ImageCompressionQuality
  - quickSpin, [410](#)
- ImageCompressionRateOption
  - quickSpin, [410](#)
- ImageStatistics Access, [219](#)
  - spinImageStatisticsCreate, [220](#)
  - spinImageStatisticsDestroy, [220](#)
  - spinImageStatisticsDisableAll, [220](#)
  - spinImageStatisticsEnableAll, [221](#)
  - spinImageStatisticsEnableGreyOnly, [221](#)
  - spinImageStatisticsEnableHslOnly, [222](#)
  - spinImageStatisticsEnableRgbOnly, [222](#)
  - spinImageStatisticsGetAll, [223](#)
  - spinImageStatisticsGetChannelStatus, [223](#)
  - spinImageStatisticsGetHistogram, [224](#)
  - spinImageStatisticsGetMean, [224](#)
  - spinImageStatisticsGetNumPixelValues, [225](#)
  - spinImageStatisticsGetPixelValueRange, [225](#)
  - spinImageStatisticsGetRange, [226](#)
  - spinImageStatisticsSetChannelStatus, [226](#)
- include/spinc/CameraDefsC.h, [465](#)
- include/spinc/ChunkDataDefC.h, [498](#)
- include/spinc/QuickSpinC.h, [499](#)
- include/spinc/QuickSpinDefsC.h, [499](#)
- include/spinc/SpinVideoC.h, [524](#)
- include/spinc/SpinnakerC.h, [501](#)
- include/spinc/SpinnakerDefsC.h, [511](#)
- include/spinc/SpinnakerGenApiC.h, [516](#)
- include/spinc/SpinnakerGenApiDefsC.h, [520](#)
- include/spinc/SpinnakerPlatformC.h, [523](#)
- include/spinc/TransportLayerDefsC.h, [524](#)
- include/spinc/TransportLayerDeviceC.h, [526](#)
- include/spinc/TransportLayerInterfaceC.h, [527](#)
- include/spinc/TransportLayerStreamC.h, [527](#)
- include/spinc/TransportLayerSystemC.h, [528](#)
- IncompatibleDeviceCount
  - quickSpinTLInterface, [442](#)
- IncompatibleDeviceID
  - quickSpinTLInterface, [442](#)
- IncompatibleDeviceModelName
  - quickSpinTLInterface, [442](#)
- IncompatibleDeviceSelector
  - quickSpinTLInterface, [442](#)
- IncompatibleDeviceVendorName
  - quickSpinTLInterface, [442](#)
- IncompatibleGevDeviceIPAddress
  - quickSpinTLInterface, [443](#)
- IncompatibleGevDeviceMACAddress
  - quickSpinTLInterface, [443](#)
- IncompatibleGevDeviceSubnetMask
  - quickSpinTLInterface, [443](#)
- indexedColor\_8bit
  - spinBMPOption, [449](#)
- Interface Access, [163](#)
  - spinInterfaceGetCameras, [164](#)
  - spinInterfaceGetCamerasEx, [164](#)
  - spinInterfaceGetTLNodeMap, [165](#)
  - spinInterfaceIsInUse, [165](#)
  - spinInterfaceRegisterArrivalEvent, [166](#)
  - spinInterfaceRegisterInterfaceEvent, [166](#)
  - spinInterfaceRegisterRemovalEvent, [167](#)
  - spinInterfaceRelease, [167](#)
  - spinInterfaceSendActionCommand, [168](#)
  - spinInterfaceUnregisterArrivalEvent, [168](#)
  - spinInterfaceUnregisterInterfaceEvent, [169](#)
  - spinInterfaceUnregisterRemovalEvent, [169](#)
  - spinInterfaceUpdateCameras, [170](#)
- InterfaceDisplayName
  - quickSpinTLInterface, [443](#)
- InterfaceID
  - quickSpinTLInterface, [443](#)
- InterfaceList Access, [153](#)
  - spinInterfaceListClear, [153](#)
  - spinInterfaceListCreateEmpty, [154](#)
  - spinInterfaceListDestroy, [154](#)
  - spinInterfaceListGet, [155](#)
  - spinInterfaceListGetSize, [155](#)
- InterfaceType
  - quickSpinTLInterface, [443](#)
- interlaced
  - spinPNGOption, [462](#)
- IspEnable
  - quickSpin, [410](#)
- LUTEnable



- quickSpin, [413](#)
- LUTIndex
  - quickSpin, [413](#)
- LUTSelector
  - quickSpin, [413](#)
- LUTValue
  - quickSpin, [413](#)
- LUTValueAll
  - quickSpin, [414](#)
- LineFilterWidth
  - quickSpin, [411](#)
- LineFormat
  - quickSpin, [411](#)
- LineInputFilterSelector
  - quickSpin, [411](#)
- LineInverter
  - quickSpin, [411](#)
- LineMode
  - quickSpin, [411](#)
- LinePitch
  - quickSpin, [411](#)
- LineSelector
  - quickSpin, [411](#)
- LineSource
  - quickSpin, [411](#)
- LineStatus
  - quickSpin, [412](#)
- LineStatusAll
  - quickSpin, [412](#)
- LinkErrorCount
  - quickSpin, [412](#)
- LinkUptime
  - quickSpin, [412](#)
- Logging Event Data Access, [228](#)
  - spinLogDataGetCategoryName, [228](#)
  - spinLogDataGetLogMessage, [229](#)
  - spinLogDataGetNDC, [229](#)
  - spinLogDataGetPriority, [230](#)
  - spinLogDataGetPriorityName, [230](#)
  - spinLogDataGetThreadName, [231](#)
  - spinLogDataGetTimestamp, [231](#)
- LogicBlockLUTInputActivation
  - quickSpin, [412](#)
- LogicBlockLUTInputSelector
  - quickSpin, [412](#)
- LogicBlockLUTInputSource
  - quickSpin, [412](#)
- LogicBlockLUTOutputValue
  - quickSpin, [412](#)
- LogicBlockLUTOutputValueAll
  - quickSpin, [413](#)
- LogicBlockLUTRowIndex
  - quickSpin, [413](#)
- LogicBlockLUTSelector
  - quickSpin, [413](#)
- LogicBlockSelector
  - quickSpin, [413](#)
- m\_blackLevel
  - spinChunkData, [450](#)
- m\_cRC
  - spinChunkData, [451](#)
- m\_counterValue
  - spinChunkData, [451](#)
- m\_encoderValue
  - spinChunkData, [451](#)
- m\_exposureEndLineStatusAll
  - spinChunkData, [451](#)
- m\_exposureTime
  - spinChunkData, [451](#)
- m\_framelD
  - spinChunkData, [451](#)
- m\_gain
  - spinChunkData, [451](#)
- m\_height
  - spinChunkData, [451](#)
- m\_image
  - spinChunkData, [452](#)
- m\_inferenceConfidence
  - spinChunkData, [452](#)
- m\_inferenceResult
  - spinChunkData, [452](#)
- m\_linePitch
  - spinChunkData, [452](#)
- m\_lineStatusAll
  - spinChunkData, [452](#)
- m\_offsetX
  - spinChunkData, [452](#)
- m\_offsetY
  - spinChunkData, [452](#)
- m\_partSelector
  - spinChunkData, [452](#)
- m\_pixelDynamicRangeMax
  - spinChunkData, [453](#)
- m\_pixelDynamicRangeMin
  - spinChunkData, [453](#)
- m\_scan3dAxisMax
  - spinChunkData, [453](#)
- m\_scan3dAxisMin
  - spinChunkData, [453](#)
- m\_scan3dCoordinateOffset
  - spinChunkData, [453](#)
- m\_scan3dCoordinateReferenceValue
  - spinChunkData, [453](#)
- m\_scan3dCoordinateScale
  - spinChunkData, [453](#)
- m\_scan3dInvalidDataValue
  - spinChunkData, [453](#)
- m\_scan3dTransformValue
  - spinChunkData, [454](#)
- m\_scanLineSelector
  - spinChunkData, [454](#)
- m\_sequencerSetActive
  - spinChunkData, [454](#)
- m\_serialDataLength
  - spinChunkData, [454](#)
- m\_streamChannelID

- spinChunkData, 454
- m\_timerValue
  - spinChunkData, 454
- m\_timestamp
  - spinChunkData, 454
- m\_timestampLatchValue
  - spinChunkData, 454
- m\_transferBlockID
  - spinChunkData, 455
- m\_transferQueueCurrentBlockCount
  - spinChunkData, 455
- m\_width
  - spinChunkData, 455
- major
  - spinLibraryVersion, 459
- MaxDeviceResetTime
  - quickSpin, 414
- minor
  - spinLibraryVersion, 459
- Node Access, 261
  - spinNodeDeregisterCallback, 262
  - spinNodeGetAccessMode, 262
  - spinNodeGetCachingMode, 263
  - spinNodeGetDescription, 263
  - spinNodeGetDisplayName, 264
  - spinNodeGetImposedAccessMode, 265
  - spinNodeGetImposedVisibility, 265
  - spinNodeGetName, 265
  - spinNodeGetNameSpace, 266
  - spinNodeGetPollingTime, 266
  - spinNodeGetToolTip, 267
  - spinNodeGetType, 267
  - spinNodeGetVisibility, 268
  - spinNodeInvalidateNode, 268
  - spinNodesAvailable, 269
  - spinNodesEqual, 269
  - spinNodesImplemented, 270
  - spinNodesReadable, 270
  - spinNodesWritable, 271
  - spinNodeRegisterCallback, 271
- Node Map Access, 258
  - spinNodeMapGetNode, 258
  - spinNodeMapGetNodeByIndex, 259
  - spinNodeMapGetNumNodes, 259
  - spinNodeMapPoll, 260
- OffsetX
  - quickSpin, 414
- OffsetY
  - quickSpin, 414
- POEStatus
  - quickSpinTLInterface, 443
- PacketResendRequestCount
  - quickSpin, 414
- PayloadSize
  - quickSpin, 414
- PixelColorFilter
  - quickSpin, 414
- PixelDynamicRangeMax
  - quickSpin, 414
- PixelDynamicRangeMin
  - quickSpin, 415
- PixelFormat
  - quickSpin, 415
- PixelFormatInfoID
  - quickSpin, 415
- PixelFormatInfoSelector
  - quickSpin, 415
- PixelSize
  - quickSpin, 415
- PowerSupplyCurrent
  - quickSpin, 415
- PowerSupplyVoltage
  - quickSpin, 415
- progressive
  - spinJPEGOption, 457
- quality
  - spinJPEGOption, 457
  - spinJPG2Option, 458
  - spinMJPEGOption, 460
- quickSpin, 336
  - aPAUSEMACCtrlFramesReceived, 351
  - aPAUSEMACCtrlFramesTransmitted, 351
  - AasRoiEnable, 348
  - AasRoiHeight, 348
  - AasRoiOffsetX, 348
  - AasRoiOffsetY, 348
  - AasRoiWidth, 348
  - AcquisitionAbort, 348
  - AcquisitionArm, 349
  - AcquisitionBurstFrameCount, 349
  - AcquisitionFrameCount, 349
  - AcquisitionFrameRate, 349
  - AcquisitionFrameRateEnable, 349
  - AcquisitionLineRate, 349
  - AcquisitionMode, 349
  - AcquisitionResultingFrameRate, 349
  - AcquisitionStart, 350
  - AcquisitionStatus, 350
  - AcquisitionStatusSelector, 350
  - AcquisitionStop, 350
  - ActionDeviceKey, 350
  - ActionGroupKey, 350
  - ActionGroupMask, 350
  - ActionQueueSize, 350
  - ActionSelector, 351
  - ActionUnconditionalMode, 351
  - AdaptiveCompressionEnable, 351
  - AdcBitDepth, 351
  - AutoAlgorithmSelector, 351
  - AutoExposureControlLoopDamping, 351
  - AutoExposureControlPriority, 352
  - AutoExposureEVCompensation, 352
  - AutoExposureExposureTimeLowerLimit, 352
  - AutoExposureExposureTimeUpperLimit, 352



- AutoExposureGainLowerLimit, [352](#)
- AutoExposureGainUpperLimit, [352](#)
- AutoExposureGreyValueLowerLimit, [352](#)
- AutoExposureGreyValueUpperLimit, [352](#)
- AutoExposureLightingMode, [353](#)
- AutoExposureMeteringMode, [353](#)
- AutoExposureTargetGreyValue, [353](#)
- AutoExposureTargetGreyValueAuto, [353](#)
- BalanceRatio, [353](#)
- BalanceRatioSelector, [353](#)
- BalanceWhiteAuto, [353](#)
- BalanceWhiteAutoDamping, [353](#)
- BalanceWhiteAutoLowerLimit, [354](#)
- BalanceWhiteAutoProfile, [354](#)
- BalanceWhiteAutoUpperLimit, [354](#)
- BinningHorizontal, [354](#)
- BinningHorizontalMode, [354](#)
- BinningSelector, [354](#)
- BinningVertical, [354](#)
- BinningVerticalMode, [354](#)
- BlackLevel, [355](#)
- BlackLevelAuto, [355](#)
- BlackLevelAutoBalance, [355](#)
- BlackLevelClampingEnable, [355](#)
- BlackLevelRaw, [355](#)
- BlackLevelSelector, [355](#)
- ChunkBlackLevel, [355](#)
- ChunkBlackLevelSelector, [355](#)
- ChunkCRC, [356](#)
- ChunkCounterSelector, [356](#)
- ChunkCounterValue, [356](#)
- ChunkEnable, [356](#)
- ChunkEncoderSelector, [356](#)
- ChunkEncoderStatus, [356](#)
- ChunkEncoderValue, [356](#)
- ChunkExposureEndLineStatusAll, [356](#)
- ChunkExposureTime, [357](#)
- ChunkExposureTimeSelector, [357](#)
- ChunkFrameID, [357](#)
- ChunkGain, [357](#)
- ChunkGainSelector, [357](#)
- ChunkHeight, [357](#)
- ChunkImage, [357](#)
- ChunkImageComponent, [357](#)
- ChunkInferenceConfidence, [358](#)
- ChunkInferenceResult, [358](#)
- ChunkLinePitch, [358](#)
- ChunkLineStatusAll, [358](#)
- ChunkModeActive, [358](#)
- ChunkOffsetX, [358](#)
- ChunkOffsetY, [358](#)
- ChunkPartSelector, [358](#)
- ChunkPixelDynamicRangeMax, [359](#)
- ChunkPixelDynamicRangeMin, [359](#)
- ChunkPixelFormat, [359](#)
- ChunkRegionID, [359](#)
- ChunkScan3dAxisMax, [359](#)
- ChunkScan3dAxisMin, [359](#)
- ChunkScan3dCoordinateOffset, [359](#)
- ChunkScan3dCoordinateReferenceSelector, [359](#)
- ChunkScan3dCoordinateReferenceValue, [360](#)
- ChunkScan3dCoordinateScale, [360](#)
- ChunkScan3dCoordinateSelector, [360](#)
- ChunkScan3dCoordinateSystem, [360](#)
- ChunkScan3dCoordinateSystemReference, [360](#)
- ChunkScan3dCoordinateTransformSelector, [360](#)
- ChunkScan3dDistanceUnit, [360](#)
- ChunkScan3dInvalidDataFlag, [360](#)
- ChunkScan3dInvalidDataValue, [361](#)
- ChunkScan3dOutputMode, [361](#)
- ChunkScan3dTransformValue, [361](#)
- ChunkScanLineSelector, [361](#)
- ChunkSelector, [361](#)
- ChunkSequencerSetActive, [361](#)
- ChunkSerialData, [361](#)
- ChunkSerialDataLength, [361](#)
- ChunkSerialReceiveOverflow, [362](#)
- ChunkSourceID, [362](#)
- ChunkStreamChannelID, [362](#)
- ChunkTimerSelector, [362](#)
- ChunkTimerValue, [362](#)
- ChunkTimestamp, [362](#)
- ChunkTimestampLatchValue, [362](#)
- ChunkTransferBlockID, [362](#)
- ChunkTransferQueueCurrentBlockCount, [363](#)
- ChunkTransferStreamID, [363](#)
- ChunkWidth, [363](#)
- CICongfiguration, [363](#)
- CITimeSlotsCount, [363](#)
- ColorTransformationEnable, [363](#)
- ColorTransformationSelector, [363](#)
- ColorTransformationValue, [363](#)
- ColorTransformationValueSelector, [364](#)
- CompressionRatio, [364](#)
- CounterDelay, [364](#)
- CounterDuration, [364](#)
- CounterEventActivation, [364](#)
- CounterEventSource, [364](#)
- CounterReset, [364](#)
- CounterResetActivation, [364](#)
- CounterResetSource, [365](#)
- CounterSelector, [365](#)
- CounterStatus, [365](#)
- CounterTriggerActivation, [365](#)
- CounterTriggerSource, [365](#)
- CounterValue, [365](#)
- CounterValueAtReset, [365](#)
- CxpConnectionSelector, [365](#)
- CxpConnectionTestErrorCount, [366](#)
- CxpConnectionTestMode, [366](#)
- CxpConnectionTestPacketCount, [366](#)
- CxpLinkConfiguration, [366](#)
- CxpLinkConfigurationPreferred, [366](#)
- CxpLinkConfigurationStatus, [366](#)
- CxpPoCxpAuto, [366](#)
- CxpPoCxpStatus, [366](#)

- CxpPoCxpTripReset, [367](#)
- CxpPoCxpTurnOff, [367](#)
- DecimationHorizontal, [367](#)
- DecimationHorizontalMode, [367](#)
- DecimationSelector, [367](#)
- DecimationVertical, [367](#)
- DecimationVerticalMode, [367](#)
- DefectCorrectStaticEnable, [368](#)
- DefectCorrectionMode, [367](#)
- DefectTableApply, [368](#)
- DefectTableCoordinateX, [368](#)
- DefectTableCoordinateY, [368](#)
- DefectTableFactoryRestore, [368](#)
- DefectTableIndex, [368](#)
- DefectTablePixelCount, [368](#)
- DefectTableSave, [368](#)
- Deinterlacing, [369](#)
- DeviceCharacterSet, [369](#)
- DeviceClockFrequency, [369](#)
- DeviceClockSelector, [369](#)
- DeviceConnectionSelector, [369](#)
- DeviceConnectionSpeed, [369](#)
- DeviceConnectionStatus, [369](#)
- DeviceEventChannelCount, [369](#)
- DeviceFamilyName, [370](#)
- DeviceFeaturePersistenceEnd, [370](#)
- DeviceFeaturePersistenceStart, [370](#)
- DeviceFirmwareVersion, [370](#)
- DeviceGenCPVersionMajor, [370](#)
- DeviceGenCPVersionMinor, [370](#)
- DeviceID, [370](#)
- DeviceIndicatorMode, [370](#)
- DeviceLinkBandwidthReserve, [371](#)
- DeviceLinkCommandTimeout, [371](#)
- DeviceLinkConnectionCount, [371](#)
- DeviceLinkCurrentThroughput, [371](#)
- DeviceLinkHeartbeatMode, [371](#)
- DeviceLinkHeartbeatTimeout, [371](#)
- DeviceLinkSelector, [371](#)
- DeviceLinkSpeed, [371](#)
- DeviceLinkThroughputLimit, [372](#)
- DeviceLinkThroughputLimitMode, [372](#)
- DeviceManifestEntrySelector, [372](#)
- DeviceManifestPrimaryURL, [372](#)
- DeviceManifestSchemaMajorVersion, [372](#)
- DeviceManifestSchemaMinorVersion, [372](#)
- DeviceManifestSecondaryURL, [372](#)
- DeviceManifestXMLMajorVersion, [372](#)
- DeviceManifestXMLMinorVersion, [373](#)
- DeviceManifestXMLSubMinorVersion, [373](#)
- DeviceManufacturerInfo, [373](#)
- DeviceMaxThroughput, [373](#)
- DeviceModelName, [373](#)
- DevicePowerSupplySelector, [373](#)
- DeviceRegistersCheck, [373](#)
- DeviceRegistersEndianness, [373](#)
- DeviceRegistersStreamingEnd, [374](#)
- DeviceRegistersStreamingStart, [374](#)
- DeviceRegistersValid, [374](#)
- DeviceReset, [374](#)
- DeviceSFNCVersionMajor, [375](#)
- DeviceSFNCVersionMinor, [375](#)
- DeviceSFNCVersionSubMinor, [375](#)
- DeviceScanType, [374](#)
- DeviceSerialNumber, [374](#)
- DeviceSerialPortBaudRate, [374](#)
- DeviceSerialPortSelector, [374](#)
- DeviceStreamChannelCount, [375](#)
- DeviceStreamChannelEndianness, [375](#)
- DeviceStreamChannelLink, [375](#)
- DeviceStreamChannelPacketSize, [375](#)
- DeviceStreamChannelSelector, [375](#)
- DeviceStreamChannelType, [376](#)
- DeviceTLType, [376](#)
- DeviceTLVersionMajor, [376](#)
- DeviceTLVersionMinor, [376](#)
- DeviceTLVersionSubMinor, [376](#)
- DeviceTapGeometry, [376](#)
- DeviceTemperature, [376](#)
- DeviceTemperatureSelector, [376](#)
- DeviceType, [377](#)
- DeviceUptime, [377](#)
- DeviceUserID, [377](#)
- DeviceVendorName, [377](#)
- DeviceVersion, [377](#)
- EncoderDivider, [377](#)
- EncoderMode, [377](#)
- EncoderOutputMode, [377](#)
- EncoderReset, [378](#)
- EncoderResetActivation, [378](#)
- EncoderResetSource, [378](#)
- EncoderSelector, [378](#)
- EncoderSourceA, [378](#)
- EncoderSourceB, [378](#)
- EncoderStatus, [378](#)
- EncoderTimeout, [378](#)
- EncoderValue, [379](#)
- EncoderValueAtReset, [379](#)
- EnumerationCount, [379](#)
- EventAcquisitionEnd, [379](#)
- EventAcquisitionEndFrameID, [379](#)
- EventAcquisitionEndTimestamp, [379](#)
- EventAcquisitionError, [379](#)
- EventAcquisitionErrorFrameID, [379](#)
- EventAcquisitionErrorTimestamp, [380](#)
- EventAcquisitionStart, [380](#)
- EventAcquisitionStartFrameID, [380](#)
- EventAcquisitionStartTimestamp, [380](#)
- EventAcquisitionTransferEnd, [380](#)
- EventAcquisitionTransferEndFrameID, [380](#)
- EventAcquisitionTransferEndTimestamp, [380](#)
- EventAcquisitionTransferStart, [380](#)
- EventAcquisitionTransferStartFrameID, [381](#)
- EventAcquisitionTransferStartTimestamp, [381](#)
- EventAcquisitionTrigger, [381](#)
- EventAcquisitionTriggerFrameID, [381](#)

- EventAcquisitionTriggerTimestamp, 381
- EventActionLate, 381
- EventActionLateFrameID, 381
- EventActionLateTimestamp, 381
- EventCounter0End, 382
- EventCounter0EndFrameID, 382
- EventCounter0EndTimestamp, 382
- EventCounter0Start, 382
- EventCounter0StartFrameID, 382
- EventCounter0StartTimestamp, 382
- EventCounter1End, 382
- EventCounter1EndFrameID, 382
- EventCounter1EndTimestamp, 383
- EventCounter1Start, 383
- EventCounter1StartFrameID, 383
- EventCounter1StartTimestamp, 383
- EventEncoder0Restarted, 383
- EventEncoder0RestartedFrameID, 383
- EventEncoder0RestartedTimestamp, 383
- EventEncoder0Stopped, 383
- EventEncoder0StoppedFrameID, 384
- EventEncoder0StoppedTimestamp, 384
- EventEncoder1Restarted, 384
- EventEncoder1RestartedFrameID, 384
- EventEncoder1RestartedTimestamp, 384
- EventEncoder1Stopped, 384
- EventEncoder1StoppedFrameID, 384
- EventEncoder1StoppedTimestamp, 384
- EventError, 385
- EventErrorCode, 385
- EventErrorFrameID, 385
- EventErrorTimestamp, 385
- EventExposureEnd, 385
- EventExposureEndFrameID, 385
- EventExposureEndTimestamp, 385
- EventExposureStart, 385
- EventExposureStartFrameID, 386
- EventExposureStartTimestamp, 386
- EventFrameBurstEnd, 386
- EventFrameBurstEndFrameID, 386
- EventFrameBurstEndTimestamp, 386
- EventFrameBurstStart, 386
- EventFrameBurstStartFrameID, 386
- EventFrameBurstStartTimestamp, 386
- EventFrameEnd, 387
- EventFrameEndFrameID, 387
- EventFrameEndTimestamp, 387
- EventFrameStart, 387
- EventFrameStartFrameID, 387
- EventFrameStartTimestamp, 387
- EventFrameTransferEnd, 387
- EventFrameTransferEndFrameID, 387
- EventFrameTransferEndTimestamp, 388
- EventFrameTransferStart, 388
- EventFrameTransferStartFrameID, 388
- EventFrameTransferStartTimestamp, 388
- EventFrameTrigger, 388
- EventFrameTriggerFrameID, 388
- EventFrameTriggerTimestamp, 388
- EventLine0AnyEdge, 388
- EventLine0AnyEdgeFrameID, 389
- EventLine0AnyEdgeTimestamp, 389
- EventLine0FallingEdge, 389
- EventLine0FallingEdgeFrameID, 389
- EventLine0FallingEdgeTimestamp, 389
- EventLine0RisingEdge, 389
- EventLine0RisingEdgeFrameID, 389
- EventLine0RisingEdgeTimestamp, 389
- EventLine1AnyEdge, 390
- EventLine1AnyEdgeFrameID, 390
- EventLine1AnyEdgeTimestamp, 390
- EventLine1FallingEdge, 390
- EventLine1FallingEdgeFrameID, 390
- EventLine1FallingEdgeTimestamp, 390
- EventLine1RisingEdge, 390
- EventLine1RisingEdgeFrameID, 390
- EventLine1RisingEdgeTimestamp, 391
- EventLinkSpeedChange, 391
- EventLinkSpeedChangeFrameID, 391
- EventLinkSpeedChangeTimestamp, 391
- EventLinkTrigger0, 391
- EventLinkTrigger0FrameID, 391
- EventLinkTrigger0Timestamp, 391
- EventLinkTrigger1, 391
- EventLinkTrigger1FrameID, 392
- EventLinkTrigger1Timestamp, 392
- EventNotification, 392
- EventSelector, 392
- EventSequencerSetChange, 392
- EventSequencerSetChangeFrameID, 392
- EventSequencerSetChangeTimestamp, 392
- EventSerialData, 392
- EventSerialDataLength, 393
- EventSerialPortReceive, 393
- EventSerialPortReceiveTimestamp, 393
- EventSerialReceiveOverflow, 393
- EventStream0TransferBlockEnd, 393
- EventStream0TransferBlockEndFrameID, 393
- EventStream0TransferBlockEndTimestamp, 393
- EventStream0TransferBlockStart, 393
- EventStream0TransferBlockStartFrameID, 394
- EventStream0TransferBlockStartTimestamp, 394
- EventStream0TransferBlockTrigger, 394
- EventStream0TransferBlockTriggerFrameID, 394
- EventStream0TransferBlockTriggerTimestamp, 394
- EventStream0TransferBurstEnd, 394
- EventStream0TransferBurstEndFrameID, 394
- EventStream0TransferBurstEndTimestamp, 394
- EventStream0TransferBurstStart, 395
- EventStream0TransferBurstStartFrameID, 395
- EventStream0TransferBurstStartTimestamp, 395
- EventStream0TransferEnd, 395
- EventStream0TransferEndFrameID, 395
- EventStream0TransferEndTimestamp, 395
- EventStream0TransferOverflow, 395
- EventStream0TransferOverflowFrameID, 395

- EventStream0TransferOverflowTimestamp, 396
- EventStream0TransferPause, 396
- EventStream0TransferPauseFrameID, 396
- EventStream0TransferPauseTimestamp, 396
- EventStream0TransferResume, 396
- EventStream0TransferResumeFrameID, 396
- EventStream0TransferResumeTimestamp, 396
- EventStream0TransferStart, 396
- EventStream0TransferStartFrameID, 397
- EventStream0TransferStartTimestamp, 397
- EventTest, 397
- EventTestTimestamp, 397
- EventTimer0End, 397
- EventTimer0EndFrameID, 397
- EventTimer0EndTimestamp, 397
- EventTimer0Start, 397
- EventTimer0StartFrameID, 398
- EventTimer0StartTimestamp, 398
- EventTimer1End, 398
- EventTimer1EndFrameID, 398
- EventTimer1EndTimestamp, 398
- EventTimer1Start, 398
- EventTimer1StartFrameID, 398
- EventTimer1StartTimestamp, 398
- ExposureActiveMode, 399
- ExposureAuto, 399
- ExposureMode, 399
- ExposureTime, 399
- ExposureTimeMode, 399
- ExposureTimeSelector, 399
- FactoryReset, 399
- FileAccessBuffer, 399
- FileAccessLength, 400
- FileAccessOffset, 400
- FileOpenMode, 400
- FileOperationExecute, 400
- FileOperationResult, 400
- FileOperationSelector, 400
- FileOperationStatus, 400
- FileSelector, 400
- FileSize, 401
- Gain, 401
- GainAuto, 401
- GainAutoBalance, 401
- GainSelector, 401
- Gamma, 401
- GammaEnable, 401
- GevActiveLinkCount, 401
- GevCCP, 402
- GevCurrentDefaultGateway, 402
- GevCurrentIPAddress, 402
- GevCurrentIPConfigurationDHCP, 402
- GevCurrentIPConfigurationLLA, 402
- GevCurrentIPConfigurationPersistentIP, 402
- GevCurrentPhysicalLinkConfiguration, 402
- GevCurrentSubnetMask, 402
- GevDiscoveryAckDelay, 403
- GevFirstURL, 403
- GevGVCPExtendedStatusCodes, 403
- GevGVCPExtendedStatusCodesSelector, 403
- GevGVCPHeartbeatDisable, 403
- GevGVCPPendingAck, 403
- GevGVCPPendingTimeout, 403
- GevGVSPExtendedIDMode, 403
- GevHeartbeatTimeout, 404
- GevIEEE1588, 404
- GevIEEE1588ClockAccuracy, 404
- GevIEEE1588Mode, 404
- GevIEEE1588Status, 404
- GevIPConfigurationStatus, 404
- GevInterfaceSelector, 404
- GevMACAddress, 404
- GevMCDA, 405
- GevMCPHostPort, 405
- GevMCRC, 405
- GevMCSP, 405
- GevMCTT, 405
- GevNumberOfInterfaces, 405
- GevPAUSEFrameReception, 405
- GevPAUSEFrameTransmission, 405
- GevPersistentDefaultGateway, 406
- GevPersistentIPAddress, 406
- GevPersistentSubnetMask, 406
- GevPhysicalLinkConfiguration, 406
- GevPrimaryApplicationIPAddress, 406
- GevPrimaryApplicationSocket, 406
- GevPrimaryApplicationSwitchoverKey, 406
- GevSCCFGAllInTransmission, 406
- GevSCCFGExtendedChunkData, 407
- GevSCCFGPacketResendDestination, 407
- GevSCCFGUnconditionalStreaming, 407
- GevSCDA, 407
- GevSCPDDirection, 407
- GevSCPHostPort, 407
- GevSCPInterfaceIndex, 407
- GevSCPSBigEndian, 408
- GevSCPSDoNotFragment, 408
- GevSCPSFireTestPacket, 408
- GevSCPSPacketSize, 408
- GevSCPD, 407
- GevSCSP, 408
- GevSCZoneConfigurationLock, 408
- GevSCZoneCount, 408
- GevSCZoneDirectionAll, 408
- GevSecondURL, 409
- GevStreamChannelSelector, 409
- GevSupportedOption, 409
- GevSupportedOptionSelector, 409
- GevTimestampTickFrequency, 409
- GuiXmlManifestAddress, 409
- Height, 409
- HeightMax, 409
- ImageComponentEnable, 410
- ImageComponentSelector, 410
- ImageCompressionBitrate, 410
- ImageCompressionJPEGFormatOption, 410

- ImageCompressionMode, [410](#)
- ImageCompressionQuality, [410](#)
- ImageCompressionRateOption, [410](#)
- IspEnable, [410](#)
- LUTEnable, [413](#)
- LUTIndex, [413](#)
- LUTSelector, [413](#)
- LUTValue, [413](#)
- LUTValueAll, [414](#)
- LineFilterWidth, [411](#)
- LineFormat, [411](#)
- LineInputFilterSelector, [411](#)
- LineInverter, [411](#)
- LineMode, [411](#)
- LinePitch, [411](#)
- LineSelector, [411](#)
- LineSource, [411](#)
- LineStatus, [412](#)
- LineStatusAll, [412](#)
- LinkErrorCount, [412](#)
- LinkUptime, [412](#)
- LogicBlockLUTInputActivation, [412](#)
- LogicBlockLUTInputSelector, [412](#)
- LogicBlockLUTInputSource, [412](#)
- LogicBlockLUTOutputValue, [412](#)
- LogicBlockLUTOutputValueAll, [413](#)
- LogicBlockLUTRowIndex, [413](#)
- LogicBlockLUTSelector, [413](#)
- LogicBlockSelector, [413](#)
- MaxDeviceResetTime, [414](#)
- OffsetX, [414](#)
- OffsetY, [414](#)
- PacketResendRequestCount, [414](#)
- PayloadSize, [414](#)
- PixelColorFilter, [414](#)
- PixelDynamicRangeMax, [414](#)
- PixelDynamicRangeMin, [415](#)
- PixelFormat, [415](#)
- PixelFormatInfoID, [415](#)
- PixelFormatInfoSelector, [415](#)
- PixelSize, [415](#)
- PowerSupplyCurrent, [415](#)
- PowerSupplyVoltage, [415](#)
- RegionDestination, [415](#)
- RegionMode, [416](#)
- RegionSelector, [416](#)
- ReverseX, [416](#)
- ReverseY, [416](#)
- RgbTransformLightSource, [416](#)
- Saturation, [416](#)
- SaturationEnable, [416](#)
- Scan3dAxisMax, [416](#)
- Scan3dAxisMin, [417](#)
- Scan3dCoordinateOffset, [417](#)
- Scan3dCoordinateReferenceSelector, [417](#)
- Scan3dCoordinateReferenceValue, [417](#)
- Scan3dCoordinateScale, [417](#)
- Scan3dCoordinateSelector, [417](#)
- Scan3dCoordinateSystem, [417](#)
- Scan3dCoordinateSystemReference, [417](#)
- Scan3dCoordinateTransformSelector, [418](#)
- Scan3dDistanceUnit, [418](#)
- Scan3dInvalidDataFlag, [418](#)
- Scan3dInvalidDataValue, [418](#)
- Scan3dOutputMode, [418](#)
- Scan3dTransformValue, [418](#)
- SensorDescription, [418](#)
- SensorDigitizationTaps, [418](#)
- SensorHeight, [419](#)
- SensorShutterMode, [419](#)
- SensorTaps, [419](#)
- SensorWidth, [419](#)
- SequencerConfigurationMode, [419](#)
- SequencerConfigurationValid, [419](#)
- SequencerFeatureEnable, [419](#)
- SequencerMode, [419](#)
- SequencerPathSelector, [420](#)
- SequencerSetActive, [420](#)
- SequencerSetLoad, [420](#)
- SequencerSetNext, [420](#)
- SequencerSetSave, [420](#)
- SequencerSetSelector, [420](#)
- SequencerSetStart, [420](#)
- SequencerSetValid, [420](#)
- SequencerTriggerActivation, [421](#)
- SequencerTriggerSource, [421](#)
- SerialPortBaudRate, [421](#)
- SerialPortDataBits, [421](#)
- SerialPortParity, [421](#)
- SerialPortSelector, [421](#)
- SerialPortSource, [421](#)
- SerialPortStopBits, [421](#)
- SerialReceiveFramingErrorCount, [422](#)
- SerialReceiveParityErrorCount, [422](#)
- SerialReceiveQueueClear, [422](#)
- SerialReceiveQueueCurrentCharacterCount, [422](#)
- SerialReceiveQueueMaxCharacterCount, [422](#)
- SerialTransmitQueueCurrentCharacterCount, [422](#)
- SerialTransmitQueueMaxCharacterCount, [422](#)
- Sharpening, [422](#)
- SharpeningAuto, [423](#)
- SharpeningEnable, [423](#)
- SharpeningThreshold, [423](#)
- SoftwareSignalPulse, [423](#)
- SoftwareSignalSelector, [423](#)
- SourceCount, [423](#)
- SourceSelector, [423](#)
- TLParamsLocked, [426](#)
- Test0001, [423](#)
- TestEventGenerate, [424](#)
- TestPattern, [424](#)
- TestPatternGeneratorSelector, [424](#)
- TestPendingAck, [424](#)
- TimerDelay, [424](#)
- TimerDuration, [424](#)
- TimerReset, [424](#)

- TimerSelector, [424](#)
- TimerStatus, [425](#)
- TimerTriggerActivation, [425](#)
- TimerTriggerSource, [425](#)
- TimerValue, [425](#)
- Timestamp, [425](#)
- TimestampLatch, [425](#)
- TimestampLatchValue, [425](#)
- TimestampReset, [425](#)
- TransferAbort, [426](#)
- TransferBlockCount, [426](#)
- TransferBurstCount, [426](#)
- TransferComponentSelector, [426](#)
- TransferControlMode, [426](#)
- TransferOperationMode, [426](#)
- TransferPause, [426](#)
- TransferQueueCurrentBlockCount, [427](#)
- TransferQueueMaxBlockCount, [427](#)
- TransferQueueMode, [427](#)
- TransferQueueOverflowCount, [427](#)
- TransferResume, [427](#)
- TransferSelector, [427](#)
- TransferStart, [427](#)
- TransferStatus, [427](#)
- TransferStatusSelector, [428](#)
- TransferStop, [428](#)
- TransferStreamChannel, [428](#)
- TransferTriggerActivation, [428](#)
- TransferTriggerMode, [428](#)
- TransferTriggerSelector, [428](#)
- TransferTriggerSource, [428](#)
- TriggerActivation, [428](#)
- TriggerDelay, [429](#)
- TriggerDivider, [429](#)
- TriggerEventTest, [429](#)
- TriggerMode, [429](#)
- TriggerMultiplier, [429](#)
- TriggerOverlap, [429](#)
- TriggerSelector, [429](#)
- TriggerSoftware, [429](#)
- TriggerSource, [430](#)
- UserOutputSelector, [430](#)
- UserOutputValue, [430](#)
- UserOutputValueAll, [430](#)
- UserOutputValueAllMask, [430](#)
- UserSetDefault, [430](#)
- UserSetFeatureEnable, [430](#)
- UserSetLoad, [430](#)
- UserSetSave, [431](#)
- UserSetSelector, [431](#)
- V3\_3Enable, [431](#)
- WhiteClip, [431](#)
- WhiteClipSelector, [431](#)
- Width, [431](#)
- WidthMax, [431](#)
- QuickSpin Access, [130](#)
- quickSpinInit, [130](#)
- quickSpinInitEx, [130](#)
- quickSpinTLDeviceInit, [131](#)
- quickSpinTLInterfaceInit, [131](#)
- quickSpinTLStreamInit, [131](#)
- quickSpinTLSystemInit, [131](#)
- quickSpinBooleanNode
  - QuickSpinDefsC.h, [500](#)
- quickSpinCommandNode
  - QuickSpinDefsC.h, [500](#)
- QuickSpinDefsC.h
  - quickSpinBooleanNode, [500](#)
  - quickSpinCommandNode, [500](#)
  - quickSpinEnumerationNode, [500](#)
  - quickSpinFloatNode, [500](#)
  - quickSpinIntegerNode, [501](#)
  - quickSpinRegisterNode, [501](#)
  - quickSpinStringNode, [501](#)
- quickSpinEnumerationNode
  - QuickSpinDefsC.h, [500](#)
- quickSpinFloatNode
  - QuickSpinDefsC.h, [500](#)
- quickSpinInit
  - QuickSpin Access, [130](#)
- quickSpinInitEx
  - QuickSpin Access, [130](#)
- quickSpinIntegerNode
  - QuickSpinDefsC.h, [501](#)
- quickSpinRegisterNode
  - QuickSpinDefsC.h, [501](#)
- quickSpinStringNode
  - QuickSpinDefsC.h, [501](#)
- quickSpinTLDevice, [432](#)
  - DeviceAccessStatus, [433](#)
  - DeviceCurrentSpeed, [433](#)
  - DeviceDisplayName, [433](#)
  - DeviceDriverVersion, [433](#)
  - DeviceEndiannessMechanism, [433](#)
  - DeviceID, [433](#)
  - DeviceInstanceId, [433](#)
  - DevicesUpdater, [433](#)
  - DeviceLinkSpeed, [434](#)
  - DeviceLocation, [434](#)
  - DeviceModelName, [434](#)
  - DeviceMulticastMonitorMode, [434](#)
  - DeviceSerialNumber, [434](#)
  - DeviceType, [434](#)
  - DeviceU3VProtocol, [434](#)
  - DeviceUserID, [434](#)
  - DeviceVendorName, [435](#)
  - DeviceVersion, [435](#)
  - GUIXMLLocation, [437](#)
  - GUIXMLPath, [437](#)
  - GenICamXMLLocation, [435](#)
  - GenICamXMLPath, [435](#)
  - GevCCP, [435](#)
  - GevDeviceDiscoverMaximumPacketSize, [435](#)
  - GevDeviceForceIP, [435](#)
  - GevDeviceGateway, [435](#)
  - GevDeviceIPAddress, [436](#)



- GevDevicesWrongSubnet, [436](#)
- GevDeviceMACAddress, [436](#)
- GevDeviceMaximumPacketSize, [436](#)
- GevDeviceMaximumRetryCount, [436](#)
- GevDeviceModelsBigEndian, [436](#)
- GevDevicePort, [436](#)
- GevDeviceReadAndWriteTimeout, [436](#)
- GevDeviceSubnetMask, [437](#)
- GevVersionMajor, [437](#)
- GevVersionMinor, [437](#)
- quickSpinTLDeviceInit
  - QuickSpin Access, [131](#)
- quickSpinTLInterface, [438](#)
  - ActionCommand, [438](#)
  - AutoForceIP, [439](#)
  - DeviceAccessStatus, [439](#)
  - DeviceCount, [439](#)
  - DeviceID, [439](#)
  - DeviceModelName, [439](#)
  - DeviceSelector, [439](#)
  - DeviceUnlock, [439](#)
  - DeviceUpdateList, [439](#)
  - DeviceVendorName, [440](#)
  - FilterDriverStatus, [440](#)
  - GevActionDeviceKey, [440](#)
  - GevActionGroupKey, [440](#)
  - GevActionGroupMask, [440](#)
  - GevActionTime, [440](#)
  - GevDeviceIPAddress, [440](#)
  - GevDeviceMACAddress, [440](#)
  - GevDeviceSubnetMask, [441](#)
  - GevInterfaceGateway, [441](#)
  - GevInterfaceIPAddress, [441](#)
  - GevInterfaceMACAddress, [441](#)
  - GevInterfaceMTU, [441](#)
  - GevInterfaceReceiveLinkSpeed, [441](#)
  - GevInterfaceSubnetMask, [441](#)
  - GevInterfaceTransmitLinkSpeed, [441](#)
  - HostAdapterDriverVersion, [442](#)
  - HostAdapterName, [442](#)
  - HostAdapterVendor, [442](#)
  - IncompatibleDeviceCount, [442](#)
  - IncompatibleDeviceID, [442](#)
  - IncompatibleDeviceModelName, [442](#)
  - IncompatibleDeviceSelector, [442](#)
  - IncompatibleDeviceVendorName, [442](#)
  - IncompatibleGevDeviceIPAddress, [443](#)
  - IncompatibleGevDeviceMACAddress, [443](#)
  - IncompatibleGevDeviceSubnetMask, [443](#)
  - InterfaceDisplayName, [443](#)
  - InterfaceID, [443](#)
  - InterfaceType, [443](#)
  - POEStatus, [443](#)
- quickSpinTLInterfaceInit
  - QuickSpin Access, [131](#)
- quickSpinTLStream, [444](#)
  - GevFailedPacketCount, [444](#)
  - GevMaximumNumberResendBuffers, [444](#)
  - GevMaximumNumberResendRequests, [445](#)
  - GevPacketResendMode, [445](#)
  - GevPacketResendTimeout, [445](#)
  - GevResendPacketCount, [445](#)
  - GevResendRequestCount, [445](#)
  - GevTotalPacketCount, [445](#)
  - StreamBlockTransferSize, [445](#)
  - StreamBufferCountManual, [445](#)
  - StreamBufferCountMax, [446](#)
  - StreamBufferCountMode, [446](#)
  - StreamBufferCountResult, [446](#)
  - StreamBufferHandlingMode, [446](#)
  - StreamBufferUnderrunCount, [446](#)
  - StreamCRCCheckEnable, [446](#)
  - StreamDefaultBufferCount, [446](#)
  - StreamDefaultBufferCountMax, [446](#)
  - StreamDefaultBufferCountMode, [447](#)
  - StreamFailedBufferCount, [447](#)
  - StreamID, [447](#)
  - StreamTotalBufferCount, [447](#)
  - StreamType, [447](#)
- quickSpinTLStreamInit
  - QuickSpin Access, [131](#)
- quickSpinTLSystem, [447](#)
  - EnumerateGEVInterfaces, [448](#)
- quickSpinTLSystemInit
  - QuickSpin Access, [131](#)
- RegionDestination
  - quickSpin, [415](#)
- RegionMode
  - quickSpin, [416](#)
- RegionSelector
  - quickSpin, [416](#)
- reserved
  - spinAVIOption, [448](#)
  - spinBMPOption, [449](#)
  - spinH264Option, [456](#)
  - spinJPEGOption, [457](#)
  - spinJPG2Option, [458](#)
  - spinMJPEGOption, [460](#)
  - spinPGMOption, [461](#)
  - spinPNGOption, [462](#)
  - spinPPMOption, [463](#)
  - spinTIFFOption, [464](#)
- ReverseX
  - quickSpin, [416](#)
- ReverseY
  - quickSpin, [416](#)
- RgbTransformLightSource
  - quickSpin, [416](#)
- SPINNAKERC\_API\_DEPRECATED
  - AVIRecorder Access, [236](#), [237](#)
- SPINNAKERC\_API
  - SpinnakerPlatformC.h, [523](#)
- Saturation
  - quickSpin, [416](#)
- SaturationEnable

- quickSpin, [416](#)
- Scan3dAxisMax
  - quickSpin, [416](#)
- Scan3dAxisMin
  - quickSpin, [417](#)
- Scan3dCoordinateOffset
  - quickSpin, [417](#)
- Scan3dCoordinateReferenceSelector
  - quickSpin, [417](#)
- Scan3dCoordinateReferenceValue
  - quickSpin, [417](#)
- Scan3dCoordinateScale
  - quickSpin, [417](#)
- Scan3dCoordinateSelector
  - quickSpin, [417](#)
- Scan3dCoordinateSystem
  - quickSpin, [417](#)
- Scan3dCoordinateSystemReference
  - quickSpin, [417](#)
- Scan3dCoordinateTransformSelector
  - quickSpin, [418](#)
- Scan3dDistanceUnit
  - quickSpin, [418](#)
- Scan3dInvalidDataFlag
  - quickSpin, [418](#)
- Scan3dInvalidDataValue
  - quickSpin, [418](#)
- Scan3dOutputMode
  - quickSpin, [418](#)
- Scan3dTransformValue
  - quickSpin, [418](#)
- SensorDescription
  - quickSpin, [418](#)
- SensorDigitizationTaps
  - quickSpin, [418](#)
- SensorHeight
  - quickSpin, [419](#)
- SensorShutterMode
  - quickSpin, [419](#)
- SensorTaps
  - quickSpin, [419](#)
- SensorWidth
  - quickSpin, [419](#)
- SequencerConfigurationMode
  - quickSpin, [419](#)
- SequencerConfigurationValid
  - quickSpin, [419](#)
- SequencerFeatureEnable
  - quickSpin, [419](#)
- SequencerMode
  - quickSpin, [419](#)
- SequencerPathSelector
  - quickSpin, [420](#)
- SequencerSetActive
  - quickSpin, [420](#)
- SequencerSetLoad
  - quickSpin, [420](#)
- SequencerSetNext
  - quickSpin, [420](#)
- SequencerSetSave
  - quickSpin, [420](#)
- SequencerSetSelector
  - quickSpin, [420](#)
- SequencerSetStart
  - quickSpin, [420](#)
- SequencerSetValid
  - quickSpin, [420](#)
- SequencerTriggerActivation
  - quickSpin, [421](#)
- SequencerTriggerSource
  - quickSpin, [421](#)
- SerialPortBaudRate
  - quickSpin, [421](#)
- SerialPortDataBits
  - quickSpin, [421](#)
- SerialPortParity
  - quickSpin, [421](#)
- SerialPortSelector
  - quickSpin, [421](#)
- SerialPortSource
  - quickSpin, [421](#)
- SerialPortStopBits
  - quickSpin, [421](#)
- SerialReceiveFramingErrorCount
  - quickSpin, [422](#)
- SerialReceiveParityErrorCount
  - quickSpin, [422](#)
- SerialReceiveQueueClear
  - quickSpin, [422](#)
- SerialReceiveQueueCurrentCharacterCount
  - quickSpin, [422](#)
- SerialReceiveQueueMaxCharacterCount
  - quickSpin, [422](#)
- SerialTransmitQueueCurrentCharacterCount
  - quickSpin, [422](#)
- SerialTransmitQueueMaxCharacterCount
  - quickSpin, [422](#)
- Sharpening
  - quickSpin, [422](#)
- SharpeningAuto
  - quickSpin, [423](#)
- SharpeningEnable
  - quickSpin, [423](#)
- SharpeningThreshold
  - quickSpin, [423](#)
- SoftwareSignalPulse
  - quickSpin, [423](#)
- SoftwareSignalSelector
  - quickSpin, [423](#)
- SourceCount
  - quickSpin, [423](#)
- SourceSelector
  - quickSpin, [423](#)
- spinAVIOption, [448](#)
  - frameRate, [448](#)
  - reserved, [448](#)



- spinAVIRecorder
  - Spinnaker C Handles, [241](#)
- spinAccessMode
  - Spinnaker C GenICam Enumerations, [312](#)
- spinAcquisitionModeEnums
  - Camera Enumerations, [45](#)
- spinAcquisitionStatusSelectorEnums
  - Camera Enumerations, [45](#)
- spinActionUnconditionalModeEnums
  - Camera Enumerations, [46](#)
- spinAdcBitDepthEnums
  - Camera Enumerations, [46](#)
- spinArrivalEvent
  - Spinnaker C Handles, [241](#)
- spinArrivalEventCreate
  - Event Access, [212](#)
- spinArrivalEventDestroy
  - Event Access, [213](#)
- spinArrivalEventFunction
  - Spinnaker C Function Signatures, [244](#)
- spinAutoAlgorithmSelectorEnums
  - Camera Enumerations, [46](#)
- spinAutoExposureControlPriorityEnums
  - Camera Enumerations, [47](#)
- spinAutoExposureLightingModeEnums
  - Camera Enumerations, [47](#)
- spinAutoExposureMeteringModeEnums
  - Camera Enumerations, [47](#)
- spinAutoExposureTargetGreyValueAutoEnums
  - Camera Enumerations, [48](#)
- spinBMPOption, [449](#)
  - indexedColor\_8bit, [449](#)
  - reserved, [449](#)
- spinBalanceRatioSelectorEnums
  - Camera Enumerations, [48](#)
- spinBalanceWhiteAutoEnums
  - Camera Enumerations, [49](#)
- spinBalanceWhiteAutoProfileEnums
  - Camera Enumerations, [49](#)
- spinBinningHorizontalModeEnums
  - Camera Enumerations, [49](#)
- spinBinningSelectorEnums
  - Camera Enumerations, [50](#)
- spinBinningVerticalModeEnums
  - Camera Enumerations, [50](#)
- spinBlackLevelAutoBalanceEnums
  - Camera Enumerations, [50](#)
- spinBlackLevelAutoEnums
  - Camera Enumerations, [51](#)
- spinBlackLevelSelectorEnums
  - Camera Enumerations, [51](#)
- spinBooleanGetValue
  - IBoolean Access, [297](#)
- spinBooleanSetValue
  - IBoolean Access, [298](#)
- spinCachingMode
  - Spinnaker C GenICam Enumerations, [313](#)
- spinCamera
  - Spinnaker C Handles, [241](#)
- spinCameraBeginAcquisition
  - Camera Access, [172](#)
- spinCameraDelInit
  - Camera Access, [173](#)
- spinCameraDiscoverMaxPacketSize
  - Spinnaker C API, [133](#)
- spinCameraEndAcquisition
  - Camera Access, [173](#)
- spinCameraForcelP
  - SpinnakerC.h, [510](#)
- spinCameraGetAccessMode
  - Camera Access, [173](#)
- spinCameraGetGuiXml
  - Camera Access, [174](#)
- spinCameraGetNextImage
  - Camera Access, [174](#)
- spinCameraGetNextImageEx
  - Camera Access, [175](#)
- spinCameraGetNodeMap
  - Camera Access, [175](#)
- spinCameraGetTLDeviceNodeMap
  - Camera Access, [176](#)
- spinCameraGetTLStreamNodeMap
  - Camera Access, [176](#)
- spinCameraGetUniqueID
  - Camera Access, [177](#)
- spinCameraInit
  - Camera Access, [177](#)
- spinCameralSInitialized
  - Camera Access, [178](#)
- spinCameralSStreaming
  - Camera Access, [178](#)
- spinCameralSValid
  - Camera Access, [179](#)
- spinCameraList
  - Spinnaker C Handles, [241](#)
- spinCameraListAppend
  - CameraList Access, [157](#)
- spinCameraListClear
  - CameraList Access, [158](#)
- spinCameraListCreateEmpty
  - CameraList Access, [158](#)
- spinCameraListDestroy
  - CameraList Access, [159](#)
- spinCameraListGet
  - CameraList Access, [159](#)
- spinCameraListGetBySerial
  - CameraList Access, [160](#)
- spinCameraListGetSize
  - CameraList Access, [160](#)
- spinCameraListRemove
  - CameraList Access, [161](#)
- spinCameraListRemoveBySerial
  - CameraList Access, [161](#)
- spinCameraReadPort
  - Camera Access, [179](#)
- spinCameraRegisterDeviceEvent

- Camera Access, [179](#)
- spinCameraRegisterDeviceEventEx
  - Camera Access, [180](#)
- spinCameraRegisterImageEvent
  - Camera Access, [180](#)
- spinCameraRelease
  - Camera Access, [181](#)
- spinCameraUnregisterDeviceEvent
  - Camera Access, [181](#)
- spinCameraUnregisterImageEvent
  - Camera Access, [182](#)
- spinCameraWritePort
  - Camera Access, [182](#)
- spinCategoryGetFeatureByIndex
  - ICategory Access, [301](#)
- spinCategoryGetNumFeatures
  - ICategory Access, [302](#)
- spinChunkBlackLevelSelectorEnums
  - Camera Enumerations, [51](#)
- spinChunkCounterSelectorEnums
  - Camera Enumerations, [51](#)
- spinChunkData, [450](#)
  - m\_blackLevel, [450](#)
  - m\_cRC, [451](#)
  - m\_counterValue, [451](#)
  - m\_encoderValue, [451](#)
  - m\_exposureEndLineStatusAll, [451](#)
  - m\_exposureTime, [451](#)
  - m\_frameID, [451](#)
  - m\_gain, [451](#)
  - m\_height, [451](#)
  - m\_image, [452](#)
  - m\_inferenceConfidence, [452](#)
  - m\_inferenceResult, [452](#)
  - m\_linePitch, [452](#)
  - m\_lineStatusAll, [452](#)
  - m\_offsetX, [452](#)
  - m\_offsetY, [452](#)
  - m\_partSelector, [452](#)
  - m\_pixelDynamicRangeMax, [453](#)
  - m\_pixelDynamicRangeMin, [453](#)
  - m\_scan3dAxisMax, [453](#)
  - m\_scan3dAxisMin, [453](#)
  - m\_scan3dCoordinateOffset, [453](#)
  - m\_scan3dCoordinateReferenceValue, [453](#)
  - m\_scan3dCoordinateScale, [453](#)
  - m\_scan3dInvalidDataValue, [453](#)
  - m\_scan3dTransformValue, [454](#)
  - m\_scanLineSelector, [454](#)
  - m\_sequencerSetActive, [454](#)
  - m\_serialDataLength, [454](#)
  - m\_streamChannelID, [454](#)
  - m\_timerValue, [454](#)
  - m\_timestamp, [454](#)
  - m\_timestampLatchValue, [454](#)
  - m\_transferBlockID, [455](#)
  - m\_transferQueueCurrentBlockCount, [455](#)
  - m\_width, [455](#)
- spinChunkEncoderSelectorEnums
  - Camera Enumerations, [52](#)
- spinChunkEncoderStatusEnums
  - Camera Enumerations, [52](#)
- spinChunkExposureTimeSelectorEnums
  - Camera Enumerations, [52](#)
- spinChunkGainSelectorEnums
  - Camera Enumerations, [53](#)
- spinChunkImageComponentEnums
  - Camera Enumerations, [53](#)
- spinChunkPixelFormatEnums
  - Camera Enumerations, [54](#)
- spinChunkRegionIDEnums
  - Camera Enumerations, [54](#)
- spinChunkScan3dCoordinateReferenceSelectorEnums
  - Camera Enumerations, [54](#)
- spinChunkScan3dCoordinateSelectorEnums
  - Camera Enumerations, [55](#)
- spinChunkScan3dCoordinateSystemEnums
  - Camera Enumerations, [55](#)
- spinChunkScan3dCoordinateSystemReferenceEnums
  - Camera Enumerations, [55](#)
- spinChunkScan3dCoordinateTransformSelectorEnums
  - Camera Enumerations, [56](#)
- spinChunkScan3dDistanceUnitEnums
  - Camera Enumerations, [56](#)
- spinChunkScan3dOutputModeEnums
  - Camera Enumerations, [57](#)
- spinChunkSelectorEnums
  - Camera Enumerations, [57](#)
- spinChunkSourceIDEnums
  - Camera Enumerations, [58](#)
- spinChunkTimerSelectorEnums
  - Camera Enumerations, [58](#)
- spinChunkTransferStreamIDEnums
  - Camera Enumerations, [59](#)
- spinCIConfigurationEnums
  - Camera Enumerations, [59](#)
- spinCITimeSlotsCountEnums
  - Camera Enumerations, [59](#)
- spinColorProcessingAlgorithm
  - Spinnaker C Enumerations, [248](#)
- spinColorTransformationSelectorEnums
  - Camera Enumerations, [60](#)
- spinColorTransformationValueSelectorEnums
  - Camera Enumerations, [60](#)
- spinCommandExecute
  - ICommand Access, [299](#)
- spinCommandIsDone
  - ICommand Access, [300](#)
- spinCompressionMethod
  - Spinnaker C Structures, [255](#)
- spinCounterEventActivationEnums
  - Camera Enumerations, [61](#)
- spinCounterEventSourceEnums
  - Camera Enumerations, [61](#)
- spinCounterResetActivationEnums
  - Camera Enumerations, [62](#)

- spinCounterResetSourceEnums
  - Camera Enumerations, [62](#)
- spinCounterSelectorEnums
  - Camera Enumerations, [62](#)
- spinCounterStatusEnums
  - Camera Enumerations, [63](#)
- spinCounterTriggerActivationEnums
  - Camera Enumerations, [63](#)
- spinCounterTriggerSourceEnums
  - Camera Enumerations, [63](#)
- spinCxpConnectionTestModeEnums
  - Camera Enumerations, [64](#)
- spinCxpLinkConfigurationEnums
  - Camera Enumerations, [64](#)
- spinCxpLinkConfigurationPreferredEnums
  - Camera Enumerations, [65](#)
- spinCxpLinkConfigurationStatusEnums
  - Camera Enumerations, [66](#)
- spinCxpPoCxpStatusEnums
  - Camera Enumerations, [67](#)
- spinDecimationHorizontalModeEnums
  - Camera Enumerations, [68](#)
- spinDecimationSelectorEnums
  - Camera Enumerations, [68](#)
- spinDecimationVerticalModeEnums
  - Camera Enumerations, [68](#)
- spinDefectCorrectionModeEnums
  - Camera Enumerations, [68](#)
- spinDeinterlacingEnums
  - Camera Enumerations, [69](#)
- spinDeviceCharacterSetEnums
  - Camera Enumerations, [69](#)
- spinDeviceClockSelectorEnums
  - Camera Enumerations, [69](#)
- spinDeviceConnectionStatusEnums
  - Camera Enumerations, [70](#)
- spinDeviceEvent
  - Spinnaker C Handles, [241](#)
- spinDeviceEventCreate
  - Event Access, [213](#)
- spinDeviceEventData
  - Spinnaker C Handles, [242](#)
- spinDeviceEventDestroy
  - Event Access, [214](#)
- spinDeviceEventFunction
  - Spinnaker C Function Signatures, [244](#)
- spinDeviceEventGetId
  - Device Event Data Access, [233](#)
- spinDeviceEventGetName
  - Device Event Data Access, [234](#)
- spinDeviceEventGetPayloadData
  - Device Event Data Access, [234](#)
- spinDeviceEventGetPayloadDataSize
  - Device Event Data Access, [235](#)
- spinDeviceIndicatorModeEnums
  - Camera Enumerations, [70](#)
- spinDeviceLinkHeartbeatModeEnums
  - Camera Enumerations, [70](#)
- spinDeviceLinkThroughputLimitModeEnums
  - Camera Enumerations, [72](#)
- spinDevicePowerSupplySelectorEnums
  - Camera Enumerations, [72](#)
- spinDeviceRegistersEndiannessEnums
  - Camera Enumerations, [72](#)
- spinDeviceScanTypeEnums
  - Camera Enumerations, [73](#)
- spinDeviceSerialPortBaudRateEnums
  - Camera Enumerations, [73](#)
- spinDeviceSerialPortSelectorEnums
  - Camera Enumerations, [73](#)
- spinDeviceStreamChannelEndiannessEnums
  - Camera Enumerations, [73](#)
- spinDeviceStreamChannelTypeEnums
  - Camera Enumerations, [74](#)
- spinDeviceTLTypeEnums
  - Camera Enumerations, [76](#)
- spinDeviceTapGeometryEnums
  - Camera Enumerations, [74](#)
- spinDeviceTemperatureSelectorEnums
  - Camera Enumerations, [75](#)
- spinDeviceTypeEnums
  - Camera Enumerations, [76](#)
- spinDisplayNotation
  - Spinnaker C GenICam Enumerations, [313](#)
- spinEncoderModeEnums
  - Camera Enumerations, [76](#)
- spinEncoderOutputModeEnums
  - Camera Enumerations, [77](#)
- spinEncoderResetActivationEnums
  - Camera Enumerations, [77](#)
- spinEncoderResetSourceEnums
  - Camera Enumerations, [78](#)
- spinEncoderSelectorEnums
  - Camera Enumerations, [79](#)
- spinEncoderSourceAEnums
  - Camera Enumerations, [79](#)
- spinEncoderSourceBEnums
  - Camera Enumerations, [79](#)
- spinEncoderStatusEnums
  - Camera Enumerations, [80](#)
- spinEndianness
  - Spinnaker C GenICam Enumerations, [313](#)
- spinEnumerationEntryGetEnumValue
  - IEnumEntry Access, [294](#)
- spinEnumerationEntryGetIntValue
  - IEnumEntry Access, [295](#)
- spinEnumerationEntryGetSymbolic
  - IEnumEntry Access, [295](#)
- spinEnumerationGetCurrentEntry
  - IEnumeration Access, [290](#)
- spinEnumerationGetEntryByIndex
  - IEnumeration Access, [291](#)
- spinEnumerationGetEntryByName
  - IEnumeration Access, [291](#)
- spinEnumerationGetNumEntries
  - IEnumeration Access, [292](#)

- spinEnumerationSetEnumValue
  - IEnumeration Access, [292](#)
- spinEnumerationSetIntValue
  - IEnumeration Access, [293](#)
- spinError
  - Spinnaker C Enumerations, [249](#)
- spinErrorGetLast
  - Error Handling, [134](#)
- spinErrorGetLastBuildDate
  - Error Handling, [135](#)
- spinErrorGetLastBuildTime
  - Error Handling, [135](#)
- spinErrorGetLastFileName
  - Error Handling, [136](#)
- spinErrorGetLastFullMessage
  - Error Handling, [136](#)
- spinErrorGetLastFunctionName
  - Error Handling, [137](#)
- spinErrorGetLastLineNumber
  - Error Handling, [137](#)
- spinErrorGetLastMessage
  - Error Handling, [138](#)
- spinEventNotificationEnums
  - Camera Enumerations, [80](#)
- spinEventSelectorEnums
  - Camera Enumerations, [80](#)
- spinExposureActiveModeEnums
  - Camera Enumerations, [81](#)
- spinExposureAutoEnums
  - Camera Enumerations, [81](#)
- spinExposureModeEnums
  - Camera Enumerations, [81](#)
- spinExposureTimeModeEnums
  - Camera Enumerations, [82](#)
- spinExposureTimeSelectorEnums
  - Camera Enumerations, [82](#)
- spinFileOpenModeEnums
  - Camera Enumerations, [83](#)
- spinFileOperationSelectorEnums
  - Camera Enumerations, [83](#)
- spinFileOperationStatusEnums
  - Camera Enumerations, [83](#)
- spinFileSelectorEnums
  - Camera Enumerations, [84](#)
- spinFloatGetMax
  - IFloat Access, [285](#)
- spinFloatGetMin
  - IFloat Access, [286](#)
- spinFloatGetRepresentation
  - IFloat Access, [286](#)
- spinFloatGetUnit
  - IFloat Access, [287](#)
- spinFloatGetValue
  - IFloat Access, [287](#)
- spinFloatGetValueEx
  - IFloat Access, [288](#)
- spinFloatSetValue
  - IFloat Access, [288](#)
- spinFloatSetValueEx
  - IFloat Access, [289](#)
- spinGainAutoBalanceEnums
  - Camera Enumerations, [84](#)
- spinGainAutoEnums
  - Camera Enumerations, [84](#)
- spinGainSelectorEnums
  - Camera Enumerations, [85](#)
- spinGevCCPEnums
  - Camera Enumerations, [85](#)
- spinGevCurrentPhysicalLinkConfigurationEnums
  - Camera Enumerations, [85](#)
- spinGevGVCPExtendedStatusCodesSelectorEnums
  - Camera Enumerations, [85](#)
- spinGevGVSPExtendedIDModeEnums
  - Camera Enumerations, [86](#)
- spinGevIEEE1588ClockAccuracyEnums
  - Camera Enumerations, [86](#)
- spinGevIEEE1588ModeEnums
  - Camera Enumerations, [86](#)
- spinGevIEEE1588StatusEnums
  - Camera Enumerations, [87](#)
- spinGevIPConfigurationStatusEnums
  - Camera Enumerations, [87](#)
- spinGevPhysicalLinkConfigurationEnums
  - Camera Enumerations, [87](#)
- spinGevSupportedOptionSelectorEnums
  - Camera Enumerations, [88](#)
- spinH264Option, [455](#)
  - bitrate, [456](#)
  - frameRate, [456](#)
  - height, [456](#)
  - reserved, [456](#)
  - width, [456](#)
- spinImage
  - Spinnaker C Handles, [242](#)
- spinImageCalculateStatistics
  - Image Access, [185](#)
- spinImageCheckCRC
  - Image Access, [186](#)
- spinImageChunkDataGetFloatValue
  - Chunk data access, [239](#)
- spinImageChunkDataGetIntValue
  - Chunk data access, [239](#)
- spinImageComponentSelectorEnums
  - Camera Enumerations, [89](#)
- spinImageCompressionJPEGFormatOptionEnums
  - Camera Enumerations, [89](#)
- spinImageCompressionModeEnums
  - Camera Enumerations, [90](#)
- spinImageCompressionRateOptionEnums
  - Camera Enumerations, [90](#)
- spinImageConvert
  - Image Access, [186](#)
- spinImageConvertEx
  - Image Access, [187](#)
- spinImageCreate
  - Image Access, [187](#)

- spinImageCreateEmpty
  - Image Access, [188](#)
- spinImageCreateEx
  - Image Access, [188](#)
- spinImageDeepCopy
  - Image Access, [189](#)
- spinImageDestroy
  - Image Access, [189](#)
- spinImageEvent
  - Spinnaker C Handles, [242](#)
- spinImageEventCreate
  - Event Access, [214](#)
- spinImageEventDestroy
  - Event Access, [215](#)
- spinImageEventFunction
  - Spinnaker C Function Signatures, [244](#)
- spinImageFileFormat
  - Spinnaker C Enumerations, [250](#)
- spinImageGetBitsPerPixel
  - Image Access, [190](#)
- spinImageGetBufferSize
  - Image Access, [190](#)
- spinImageGetChunkLayoutID
  - Image Access, [191](#)
- spinImageGetColorProcessing
  - Image Access, [191](#)
- spinImageGetData
  - Image Access, [192](#)
- spinImageGetDefaultColorProcessing
  - Image Access, [192](#)
- spinImageGetFrameID
  - Image Access, [192](#)
- spinImageGetHeight
  - Image Access, [193](#)
- spinImageGetID
  - Image Access, [193](#)
- spinImageGetOffsetX
  - Image Access, [194](#)
- spinImageGetOffsetY
  - Image Access, [194](#)
- spinImageGetPaddingX
  - Image Access, [195](#)
- spinImageGetPaddingY
  - Image Access, [195](#)
- spinImageGetPayloadType
  - Image Access, [196](#)
- spinImageGetPixelFormat
  - Image Access, [196](#)
- spinImageGetPixelFormatName
  - Image Access, [197](#)
- spinImageGetPrivateData
  - Image Access, [197](#)
- spinImageGetSize
  - Image Access, [198](#)
- spinImageGetStatus
  - Image Access, [198](#)
- spinImageGetStatusDescription
  - Image Access, [199](#)
- spinImageGetStride
  - Image Access, [199](#)
- spinImageGetTLPayloadType
  - Image Access, [200](#)
- spinImageGetTLPixelFormat
  - Image Access, [201](#)
- spinImageGetTLPixelFormatNamespace
  - Image Access, [201](#)
- spinImageGetTimeStamp
  - Image Access, [200](#)
- spinImageGetValidPayloadSize
  - Image Access, [202](#)
- spinImageGetWidth
  - Image Access, [202](#)
- spinImageHasCRC
  - Image Access, [203](#)
- spinImageIsIncomplete
  - Image Access, [203](#)
- spinImageRelease
  - Image Access, [204](#)
- spinImageReset
  - Image Access, [204](#)
- spinImageResetEx
  - Image Access, [205](#)
- spinImageSave
  - Image Access, [206](#)
- spinImageSaveBmp
  - Image Access, [206](#)
- spinImageSaveFromExt
  - Image Access, [207](#)
- spinImageSaveJpeg
  - Image Access, [207](#)
- spinImageSaveJpg2
  - Image Access, [208](#)
- spinImageSavePgm
  - Image Access, [208](#)
- spinImageSavePng
  - Image Access, [209](#)
- spinImageSavePpm
  - Image Access, [209](#)
- spinImageSaveTiff
  - Image Access, [210](#)
- spinImageSetDefaultColorProcessing
  - Image Access, [210](#)
- spinImageStatistics
  - Spinnaker C Handles, [242](#)
- spinImageStatisticsCreate
  - ImageStatistics Access, [220](#)
- spinImageStatisticsDestroy
  - ImageStatistics Access, [220](#)
- spinImageStatisticsDisableAll
  - ImageStatistics Access, [220](#)
- spinImageStatisticsEnableAll
  - ImageStatistics Access, [221](#)
- spinImageStatisticsEnableGreyOnly
  - ImageStatistics Access, [221](#)
- spinImageStatisticsEnableHslOnly
  - ImageStatistics Access, [222](#)

- spinImageStatisticsEnableRgbOnly
  - ImageStatistics Access, [222](#)
- spinImageStatisticsGetAll
  - ImageStatistics Access, [223](#)
- spinImageStatisticsGetChannelStatus
  - ImageStatistics Access, [223](#)
- spinImageStatisticsGetHistogram
  - ImageStatistics Access, [224](#)
- spinImageStatisticsGetMean
  - ImageStatistics Access, [224](#)
- spinImageStatisticsGetNumPixelValues
  - ImageStatistics Access, [225](#)
- spinImageStatisticsGetPixelValueRange
  - ImageStatistics Access, [225](#)
- spinImageStatisticsGetRange
  - ImageStatistics Access, [226](#)
- spinImageStatisticsSetChannelStatus
  - ImageStatistics Access, [226](#)
- spinImageStatus
  - Spinnaker C Enumerations, [251](#)
- spinIncMode
  - Spinnaker C GenICam Enumerations, [314](#)
- spinInputDirection
  - Spinnaker C GenICam Enumerations, [314](#)
- spinIntegerGetInc
  - Integer Access, [280](#)
- spinIntegerGetMax
  - Integer Access, [281](#)
- spinIntegerGetMin
  - Integer Access, [281](#)
- spinIntegerGetRepresentation
  - Integer Access, [282](#)
- spinIntegerGetValue
  - Integer Access, [282](#)
- spinIntegerGetValueEx
  - Integer Access, [283](#)
- spinIntegerSetValue
  - Integer Access, [283](#)
- spinIntegerSetValueEx
  - Integer Access, [284](#)
- spinInterface
  - Spinnaker C Handles, [242](#)
- spinInterfaceEvent
  - Spinnaker C Handles, [242](#)
- spinInterfaceEventCreate
  - Event Access, [215](#)
- spinInterfaceEventDestroy
  - Event Access, [216](#)
- spinInterfaceGetCameras
  - Interface Access, [164](#)
- spinInterfaceGetCamerasEx
  - Interface Access, [164](#)
- spinInterfaceGetTLNodeMap
  - Interface Access, [165](#)
- spinInterfaceIsInUse
  - Interface Access, [165](#)
- spinInterfaceList
  - Spinnaker C Handles, [243](#)
- spinInterfaceListClear
  - InterfaceList Access, [153](#)
- spinInterfaceListCreateEmpty
  - InterfaceList Access, [154](#)
- spinInterfaceListDestroy
  - InterfaceList Access, [154](#)
- spinInterfaceListGet
  - InterfaceList Access, [155](#)
- spinInterfaceListGetSize
  - InterfaceList Access, [155](#)
- spinInterfaceRegisterArrivalEvent
  - Interface Access, [166](#)
- spinInterfaceRegisterInterfaceEvent
  - Interface Access, [166](#)
- spinInterfaceRegisterRemovalEvent
  - Interface Access, [167](#)
- spinInterfaceRelease
  - Interface Access, [167](#)
- spinInterfaceSendActionCommand
  - Interface Access, [168](#)
- spinInterfaceType
  - Spinnaker C GenICam Enumerations, [314](#)
- spinInterfaceUnregisterArrivalEvent
  - Interface Access, [168](#)
- spinInterfaceUnregisterInterfaceEvent
  - Interface Access, [169](#)
- spinInterfaceUnregisterRemovalEvent
  - Interface Access, [169](#)
- spinInterfaceUpdateCameras
  - Interface Access, [170](#)
- spinJPEGOption, [457](#)
  - progressive, [457](#)
  - quality, [457](#)
  - reserved, [457](#)
- spinJPG2Option, [458](#)
  - quality, [458](#)
  - reserved, [458](#)
- spinLUTSelectorEnums
  - Camera Enumerations, [94](#)
- spinLibraryVersion, [459](#)
  - build, [459](#)
  - major, [459](#)
  - minor, [459](#)
  - type, [459](#)
- spinLineFormatEnums
  - Camera Enumerations, [90](#)
- spinLineInputFilterSelectorEnums
  - Camera Enumerations, [91](#)
- spinLineModeEnums
  - Camera Enumerations, [91](#)
- spinLineSelectorEnums
  - Camera Enumerations, [91](#)
- spinLineSourceEnums
  - Camera Enumerations, [92](#)
- spinLinkType
  - Spinnaker C GenICam Enumerations, [315](#)
- spinLogDataGetCategoryName
  - Logging Event Data Access, [228](#)

- spinLogDataGetLogMessage
  - Logging Event Data Access, [229](#)
- spinLogDataGetNDC
  - Logging Event Data Access, [229](#)
- spinLogDataGetPriority
  - Logging Event Data Access, [230](#)
- spinLogDataGetPriorityName
  - Logging Event Data Access, [230](#)
- spinLogDataGetThreadName
  - Logging Event Data Access, [231](#)
- spinLogDataGetTimestamp
  - Logging Event Data Access, [231](#)
- spinLogEvent
  - Spinnaker C Handles, [243](#)
- spinLogEventCreate
  - Event Access, [216](#)
- spinLogEventData
  - Spinnaker C Handles, [243](#)
- spinLogEventDestroy
  - Event Access, [217](#)
- spinLogEventFunction
  - Spinnaker C Function Signatures, [245](#)
- spinLogicBlockLUTInputActivationEnums
  - Camera Enumerations, [92](#)
- spinLogicBlockLUTInputSelectorEnums
  - Camera Enumerations, [93](#)
- spinLogicBlockLUTInputSourceEnums
  - Camera Enumerations, [93](#)
- spinLogicBlockLUTSelectorEnums
  - Camera Enumerations, [94](#)
- spinLogicBlockSelectorEnums
  - Camera Enumerations, [94](#)
- spinMJPGOption, [460](#)
  - frameRate, [460](#)
  - quality, [460](#)
  - reserved, [460](#)
- spinNameSpace
  - Spinnaker C GenICam Enumerations, [316](#)
- spinNodeCallbackFunction
  - Spinnaker C GenICam Handles, [308](#)
- spinNodeCallbackHandle
  - Spinnaker C GenICam Handles, [308](#)
- spinNodeDeregisterCallback
  - Node Access, [262](#)
- spinNodeFromString
  - IValue Access, [273](#)
- spinNodeFromStringEx
  - IValue Access, [274](#)
- spinNodeGetAccessMode
  - Node Access, [262](#)
- spinNodeGetCachingMode
  - Node Access, [263](#)
- spinNodeGetDescription
  - Node Access, [263](#)
- spinNodeGetDisplayName
  - Node Access, [264](#)
- spinNodeGetImposedAccessMode
  - Node Access, [265](#)
- spinNodeGetImposedVisibility
  - Node Access, [265](#)
- spinNodeGetName
  - Node Access, [265](#)
- spinNodeGetNameSpace
  - Node Access, [266](#)
- spinNodeGetPollingTime
  - Node Access, [266](#)
- spinNodeGetToolTip
  - Node Access, [267](#)
- spinNodeGetType
  - Node Access, [267](#)
- spinNodeGetVisibility
  - Node Access, [268](#)
- spinNodeHandle
  - Spinnaker C GenICam Handles, [308](#)
- spinNodeInvalidateNode
  - Node Access, [268](#)
- spinNodesAvailable
  - Node Access, [269](#)
- spinNodesEqual
  - Node Access, [269](#)
- spinNodesImplemented
  - Node Access, [270](#)
- spinNodesReadable
  - Node Access, [270](#)
- spinNodesWritable
  - Node Access, [271](#)
- spinNodeMapGetNode
  - Node Map Access, [258](#)
- spinNodeMapGetNodeByIndex
  - Node Map Access, [259](#)
- spinNodeMapGetNumNodes
  - Node Map Access, [259](#)
- spinNodeMapHandle
  - Spinnaker C GenICam Handles, [309](#)
- spinNodeMapPoll
  - Node Map Access, [260](#)
- spinNodeRegisterCallback
  - Node Access, [271](#)
- spinNodeToString
  - IValue Access, [274](#)
- spinNodeToStringEx
  - IValue Access, [275](#)
- spinNodeType
  - Spinnaker C GenICam Enumerations, [316](#)
- spinPGMOption, [461](#)
  - binaryFile, [461](#)
  - reserved, [461](#)
- spinPNGOption, [462](#)
  - compressionLevel, [462](#)
  - interlaced, [462](#)
  - reserved, [462](#)
- spinPPMOption, [463](#)
  - binaryFile, [463](#)
  - reserved, [463](#)
- spinPayloadTypeInfoIds
  - Spinnaker C Enumerations, [252](#)



- spinPixelFormatEnums
  - Camera Enumerations, [95](#)
- spinPixelFormatInfoSelectorEnums
  - Camera Enumerations, [101](#)
- spinPixelFormatNamespaceID
  - Spinnaker C Enumerations, [252](#)
- spinPixelSizeEnums
  - Camera Enumerations, [106](#)
- spinRegionDestinationEnums
  - Camera Enumerations, [107](#)
- spinRegionModeEnums
  - Camera Enumerations, [107](#)
- spinRegionSelectorEnums
  - Camera Enumerations, [108](#)
- spinRegisterGet
  - IRegister Access, [303](#)
- spinRegisterGetAddress
  - IRegister Access, [304](#)
- spinRegisterGetEx
  - IRegister Access, [304](#)
- spinRegisterGetLength
  - IRegister Access, [305](#)
- spinRegisterSet
  - IRegister Access, [306](#)
- spinRegisterSetEx
  - IRegister Access, [306](#)
- spinRegisterSetReference
  - IRegister Access, [307](#)
- spinRemovalEvent
  - Spinnaker C Handles, [243](#)
- spinRemovalEventCreate
  - Event Access, [217](#)
- spinRemovalEventDestroy
  - Event Access, [218](#)
- spinRemovalEventFunction
  - Spinnaker C Function Signatures, [245](#)
- spinRepresentation
  - Spinnaker C GenICam Enumerations, [317](#)
- spinRgbTransformLightSourceEnums
  - Camera Enumerations, [108](#)
- spinScan3dCoordinateReferenceSelectorEnums
  - Camera Enumerations, [108](#)
- spinScan3dCoordinateSelectorEnums
  - Camera Enumerations, [109](#)
- spinScan3dCoordinateSystemEnums
  - Camera Enumerations, [109](#)
- spinScan3dCoordinateSystemReferenceEnums
  - Camera Enumerations, [109](#)
- spinScan3dCoordinateTransformSelectorEnums
  - Camera Enumerations, [110](#)
- spinScan3dDistanceUnitEnums
  - Camera Enumerations, [110](#)
- spinScan3dOutputModeEnums
  - Camera Enumerations, [111](#)
- spinSensorDigitizationTapsEnums
  - Camera Enumerations, [111](#)
- spinSensorShutterModeEnums
  - Camera Enumerations, [112](#)
- spinSensorTapsEnums
  - Camera Enumerations, [112](#)
- spinSequencerConfigurationModeEnums
  - Camera Enumerations, [113](#)
- spinSequencerConfigurationValidEnums
  - Camera Enumerations, [113](#)
- spinSequencerModeEnums
  - Camera Enumerations, [113](#)
- spinSequencerSetValidEnums
  - Camera Enumerations, [113](#)
- spinSequencerTriggerActivationEnums
  - Camera Enumerations, [114](#)
- spinSequencerTriggerSourceEnums
  - Camera Enumerations, [114](#)
- spinSerialPortBaudRateEnums
  - Camera Enumerations, [114](#)
- spinSerialPortParityEnums
  - Camera Enumerations, [115](#)
- spinSerialPortSelectorEnums
  - Camera Enumerations, [115](#)
- spinSerialPortSourceEnums
  - Camera Enumerations, [116](#)
- spinSerialPortStopBitsEnums
  - Camera Enumerations, [116](#)
- spinSign
  - Spinnaker C GenICam Enumerations, [317](#)
- spinSlope
  - Spinnaker C GenICam Enumerations, [317](#)
- spinSoftwareSignalSelectorEnums
  - Camera Enumerations, [116](#)
- spinSourceSelectorEnums
  - Camera Enumerations, [117](#)
- spinStandardNameSpace
  - Spinnaker C GenICam Enumerations, [318](#)
- spinStatisticsChannel
  - Spinnaker C Enumerations, [253](#)
- spinStringGetMaxLength
  - String Access, [276](#)
- spinStringGetValue
  - String Access, [277](#)
- spinStringGetValueEx
  - String Access, [277](#)
- spinStringSetValue
  - String Access, [278](#)
- spinStringSetValueEx
  - String Access, [278](#)
- spinSystem
  - Spinnaker C Handles, [243](#)
- spinSystemGetCameras
  - System Access, [140](#)
- spinSystemGetCamerasEx
  - System Access, [141](#)
- spinSystemGetInstance
  - System Access, [141](#)
- spinSystemGetInterfaces
  - System Access, [143](#)



- spinSystemGetLibraryVersion
  - System Access, [143](#)
- spinSystemGetLoggingLevel
  - System Access, [143](#)
- spinSystemGetTLNodeMap
  - System Access, [144](#)
- spinSystemIsInUse
  - System Access, [144](#)
- spinSystemRegisterArrivalEvent
  - System Access, [145](#)
- spinSystemRegisterInterfaceEvent
  - System Access, [145](#)
- spinSystemRegisterLogEvent
  - System Access, [146](#)
- spinSystemRegisterRemovalEvent
  - System Access, [146](#)
- spinSystemReleaseInstance
  - System Access, [147](#)
- spinSystemSendActionCommand
  - System Access, [147](#)
- spinSystemSetLoggingLevel
  - System Access, [148](#)
- spinSystemUnregisterAllLogEvents
  - System Access, [149](#)
- spinSystemUnregisterArrivalEvent
  - System Access, [149](#)
- spinSystemUnregisterInterfaceEvent
  - System Access, [150](#)
- spinSystemUnregisterLogEvent
  - System Access, [150](#)
- spinSystemUnregisterRemovalEvent
  - System Access, [151](#)
- spinSystemUpdateCameras
  - System Access, [151](#)
- spinSystemUpdateCamerasEx
  - System Access, [152](#)
- spinTIFFOption, [463](#)
  - compression, [464](#)
  - reserved, [464](#)
- spinTLDeviceAccessStatusEnums
  - Transport Layer Enumerations, [325](#)
- spinTLDeviceCurrentSpeedEnums
  - Transport Layer Enumerations, [326](#)
- spinTLDeviceEndiannessMechanismEnums
  - Transport Layer Enumerations, [326](#)
- spinTLDeviceTypeEnums
  - Transport Layer Enumerations, [326](#)
- spinTLFilterDriverStatusEnums
  - Transport Layer Enumerations, [327](#)
- spinTLGUIXMLLocationEnums
  - Transport Layer Enumerations, [328](#)
- spinTLGenlCamXMLLocationEnums
  - Transport Layer Enumerations, [327](#)
- spinTLGevCCPEnums
  - Transport Layer Enumerations, [327](#)
- spinTLPOEStatusEnums
  - Transport Layer Enumerations, [328](#)
- spinTLStreamBufferCountModeEnums
  - Transport Layer Enumerations, [328](#)
- spinTLStreamBufferHandlingModeEnums
  - Transport Layer Enumerations, [329](#)
- spinTLStreamDefaultBufferCountModeEnums
  - Transport Layer Enumerations, [329](#)
- spinTLStreamTypeEnums
  - Transport Layer Enumerations, [330](#)
- spinTestPatternEnums
  - Camera Enumerations, [117](#)
- spinTestPatternGeneratorSelectorEnums
  - Camera Enumerations, [117](#)
- spinTimerSelectorEnums
  - Camera Enumerations, [118](#)
- spinTimerStatusEnums
  - Camera Enumerations, [118](#)
- spinTimerTriggerActivationEnums
  - Camera Enumerations, [118](#)
- spinTimerTriggerSourceEnums
  - Camera Enumerations, [119](#)
- spinTransferComponentSelectorEnums
  - Camera Enumerations, [120](#)
- spinTransferControlModeEnums
  - Camera Enumerations, [120](#)
- spinTransferOperationModeEnums
  - Camera Enumerations, [121](#)
- spinTransferQueueModeEnums
  - Camera Enumerations, [121](#)
- spinTransferSelectorEnums
  - Camera Enumerations, [121](#)
- spinTransferStatusSelectorEnums
  - Camera Enumerations, [122](#)
- spinTransferTriggerActivationEnums
  - Camera Enumerations, [122](#)
- spinTransferTriggerModeEnums
  - Camera Enumerations, [122](#)
- spinTransferTriggerSelectorEnums
  - Camera Enumerations, [123](#)
- spinTransferTriggerSourceEnums
  - Camera Enumerations, [123](#)
- spinTriggerActivationEnums
  - Camera Enumerations, [124](#)
- spinTriggerModeEnums
  - Camera Enumerations, [125](#)
- spinTriggerOverlapEnums
  - Camera Enumerations, [125](#)
- spinTriggerSelectorEnums
  - Camera Enumerations, [125](#)
- spinTriggerSourceEnums
  - Camera Enumerations, [125](#)
- spinUserOutputSelectorEnums
  - Camera Enumerations, [126](#)
- spinUserSetDefaultEnums
  - Camera Enumerations, [126](#)
- spinUserSetSelectorEnums
  - Camera Enumerations, [127](#)
- spinVideo
  - Spinnaker C Handles, [243](#)
  - SpinVideo Recording Access, [321](#)

- spinVideoAppend, [321](#)
- spinVideoClose, [321](#)
- spinVideoOpenH264, [322](#)
- spinVideoOpenMJPEG, [322](#)
- spinVideoOpenUncompressed, [322](#)
- spinVideoSetMaximumFileSize, [322](#)
- spinVideoAppend
  - SpinVideo Recording Access, [321](#)
- spinVideoClose
  - SpinVideo Recording Access, [321](#)
- spinVideoOpenH264
  - SpinVideo Recording Access, [322](#)
- spinVideoOpenMJPEG
  - SpinVideo Recording Access, [322](#)
- spinVideoOpenUncompressed
  - SpinVideo Recording Access, [322](#)
- spinVideoSetMaximumFileSize
  - SpinVideo Recording Access, [322](#)
- spinVisibility
  - Spinnaker C GenICam Enumerations, [318](#)
- spinWhiteClipSelectorEnums
  - Camera Enumerations, [127](#)
- spinXMLValidation
  - Spinnaker C GenICam Enumerations, [318](#)
- spinYesNo
  - Spinnaker C GenICam Enumerations, [320](#)
- Spinnaker C API, [132](#)
  - spinCameraDiscoverMaxPacketSize, [133](#)
- Spinnaker C Definitions, [11](#)
  - bool8\_t, [12](#)
  - False, [12](#)
  - True, [12](#)
- Spinnaker C Enumerations, [246](#)
  - spinColorProcessingAlgorithm, [248](#)
  - spinError, [249](#)
  - spinImageFileFormat, [250](#)
  - spinImageStatus, [251](#)
  - spinPayloadTypeInfoIDs, [252](#)
  - spinPixelFormatNamespaceID, [252](#)
  - spinStatisticsChannel, [253](#)
  - spinnakerLogLevel, [251](#)
- Spinnaker C Function Signatures, [244](#)
  - spinArrivalEventFunction, [244](#)
  - spinDeviceEventFunction, [244](#)
  - spinImageEventFunction, [244](#)
  - spinLogEventFunction, [245](#)
  - spinRemovalEventFunction, [245](#)
- Spinnaker C GenICam API, [256](#)
- Spinnaker C GenICam Enumerations, [310](#)
  - spinAccessMode, [312](#)
  - spinCachingMode, [313](#)
  - spinDisplayNotation, [313](#)
  - spinEndianess, [313](#)
  - spinIncMode, [314](#)
  - spinInputDirection, [314](#)
  - spinInterfaceType, [314](#)
  - spinLinkType, [315](#)
  - spinNameSpace, [316](#)
  - spinNodeType, [316](#)
  - spinRepresentation, [317](#)
  - spinSign, [317](#)
  - spinSlope, [317](#)
  - spinStandardNameSpace, [318](#)
  - spinVisibility, [318](#)
  - spinXMLValidation, [318](#)
  - spinYesNo, [320](#)
- Spinnaker C GenICam Handles, [308](#)
  - spinNodeCallbackFunction, [308](#)
  - spinNodeCallbackHandle, [308](#)
  - spinNodeHandle, [308](#)
  - spinNodeMapHandle, [309](#)
- Spinnaker C Handles, [240](#)
  - spinAVIRecorder, [241](#)
  - spinArrivalEvent, [241](#)
  - spinCamera, [241](#)
  - spinCameraList, [241](#)
  - spinDeviceEvent, [241](#)
  - spinDeviceEventData, [242](#)
  - spinImage, [242](#)
  - spinImageEvent, [242](#)
  - spinImageStatistics, [242](#)
  - spinInterface, [242](#)
  - spinInterfaceEvent, [242](#)
  - spinInterfaceList, [243](#)
  - spinLogEvent, [243](#)
  - spinLogEventData, [243](#)
  - spinRemovalEvent, [243](#)
  - spinSystem, [243](#)
  - spinVideo, [243](#)
- Spinnaker C QuickSpin API, [129](#)
- Spinnaker C Structures, [254](#)
  - actionCommandStatus, [255](#)
  - spinCompressionMethod, [255](#)
- SpinnakerC.h
  - spinCameraForceIP, [510](#)
- spinnakerLogLevel
  - Spinnaker C Enumerations, [251](#)
- SpinnakerPlatformC.h
  - SPINNAKERC\_API, [523](#)
- Status
  - actionCommandResult, [335](#)
- StreamBlockTransferSize
  - quickSpinTLStream, [445](#)
- StreamBufferCountManual
  - quickSpinTLStream, [445](#)
- StreamBufferCountMax
  - quickSpinTLStream, [446](#)
- StreamBufferCountMode
  - quickSpinTLStream, [446](#)
- StreamBufferCountResult
  - quickSpinTLStream, [446](#)
- StreamBufferHandlingMode
  - quickSpinTLStream, [446](#)
- StreamBufferUnderrunCount
  - quickSpinTLStream, [446](#)
- StreamCRCCheckEnable

- quickSpinTLStream, [446](#)
- StreamDefaultBufferCount
  - quickSpinTLStream, [446](#)
- StreamDefaultBufferCountMax
  - quickSpinTLStream, [446](#)
- StreamDefaultBufferCountMode
  - quickSpinTLStream, [447](#)
- StreamFailedBufferCount
  - quickSpinTLStream, [447](#)
- StreamID
  - quickSpinTLStream, [447](#)
- StreamTotalBufferCount
  - quickSpinTLStream, [447](#)
- StreamType
  - quickSpinTLStream, [447](#)
- String Access, [276](#)
  - spinStringGetMaxLength, [276](#)
  - spinStringGetValue, [277](#)
  - spinStringGetValueEx, [277](#)
  - spinStringSetValue, [278](#)
  - spinStringSetValueEx, [278](#)
- System Access, [139](#)
  - spinSystemGetCameras, [140](#)
  - spinSystemGetCamerasEx, [141](#)
  - spinSystemGetInstance, [141](#)
  - spinSystemGetInterfaces, [143](#)
  - spinSystemGetLibraryVersion, [143](#)
  - spinSystemGetLoggingLevel, [143](#)
  - spinSystemGetTLNodeMap, [144](#)
  - spinSystemIsInUse, [144](#)
  - spinSystemRegisterArrivalEvent, [145](#)
  - spinSystemRegisterInterfaceEvent, [145](#)
  - spinSystemRegisterLogEvent, [146](#)
  - spinSystemRegisterRemovalEvent, [146](#)
  - spinSystemReleaseInstance, [147](#)
  - spinSystemSendActionCommand, [147](#)
  - spinSystemSetLoggingLevel, [148](#)
  - spinSystemUnregisterAllLogEvents, [149](#)
  - spinSystemUnregisterArrivalEvent, [149](#)
  - spinSystemUnregisterInterfaceEvent, [150](#)
  - spinSystemUnregisterLogEvent, [150](#)
  - spinSystemUnregisterRemovalEvent, [151](#)
  - spinSystemUpdateCameras, [151](#)
  - spinSystemUpdateCamerasEx, [152](#)
- TLDevice Structures, [331](#)
- TLInterface Structures, [332](#)
- TLParamsLocked
  - quickSpin, [426](#)
- TLStream Structures, [333](#)
- TLSystem Structures, [334](#)
- Test0001
  - quickSpin, [423](#)
- TestEventGenerate
  - quickSpin, [424](#)
- TestPattern
  - quickSpin, [424](#)
- TestPatternGeneratorSelector
  - quickSpin, [424](#)
- TestPendingAck
  - quickSpin, [424](#)
- TimerDelay
  - quickSpin, [424](#)
- TimerDuration
  - quickSpin, [424](#)
- TimerReset
  - quickSpin, [424](#)
- TimerSelector
  - quickSpin, [424](#)
- TimerStatus
  - quickSpin, [425](#)
- TimerTriggerActivation
  - quickSpin, [425](#)
- TimerTriggerSource
  - quickSpin, [425](#)
- TimerValue
  - quickSpin, [425](#)
- Timestamp
  - quickSpin, [425](#)
- TimestampLatch
  - quickSpin, [425](#)
- TimestampLatchValue
  - quickSpin, [425](#)
- TimestampReset
  - quickSpin, [425](#)
- TransferAbort
  - quickSpin, [426](#)
- TransferBlockCount
  - quickSpin, [426](#)
- TransferBurstCount
  - quickSpin, [426](#)
- TransferComponentSelector
  - quickSpin, [426](#)
- TransferControlMode
  - quickSpin, [426](#)
- TransferOperationMode
  - quickSpin, [426](#)
- TransferPause
  - quickSpin, [426](#)
- TransferQueueCurrentBlockCount
  - quickSpin, [427](#)
- TransferQueueMaxBlockCount
  - quickSpin, [427](#)
- TransferQueueMode
  - quickSpin, [427](#)
- TransferQueueOverflowCount
  - quickSpin, [427](#)
- TransferResume
  - quickSpin, [427](#)
- TransferSelector
  - quickSpin, [427](#)
- TransferStart
  - quickSpin, [427](#)
- TransferStatus
  - quickSpin, [427](#)
- TransferStatusSelector
  - quickSpin, [428](#)

- TransferStop
  - quickSpin, [428](#)
- TransferStreamChannel
  - quickSpin, [428](#)
- TransferTriggerActivation
  - quickSpin, [428](#)
- TransferTriggerMode
  - quickSpin, [428](#)
- TransferTriggerSelector
  - quickSpin, [428](#)
- TransferTriggerSource
  - quickSpin, [428](#)
- Transport Layer Enumerations, [324](#)
  - spinTLDeviceAccessStatusEnums, [325](#)
  - spinTLDeviceCurrentSpeedEnums, [326](#)
  - spinTLDeviceEndianessMechanismEnums, [326](#)
  - spinTLDeviceTypeEnums, [326](#)
  - spinTLFilterDriverStatusEnums, [327](#)
  - spinTLGUIXMLLocationEnums, [328](#)
  - spinTLGenICamXMLLocationEnums, [327](#)
  - spinTLGevCCPEnums, [327](#)
  - spinTLPOEStatusEnums, [328](#)
  - spinTLStreamBufferCountModeEnums, [328](#)
  - spinTLStreamBufferHandlingModeEnums, [329](#)
  - spinTLStreamDefaultBufferCountModeEnums, [329](#)
  - spinTLStreamTypeEnum, [330](#)
- TriggerActivation
  - quickSpin, [428](#)
- TriggerDelay
  - quickSpin, [429](#)
- TriggerDivider
  - quickSpin, [429](#)
- TriggerEventTest
  - quickSpin, [429](#)
- TriggerMode
  - quickSpin, [429](#)
- TriggerMultiplier
  - quickSpin, [429](#)
- TriggerOverlap
  - quickSpin, [429](#)
- TriggerSelector
  - quickSpin, [429](#)
- TriggerSoftware
  - quickSpin, [429](#)
- TriggerSource
  - quickSpin, [430](#)
- True
  - Spinnaker C Definitions, [12](#)
- type
  - spinLibraryVersion, [459](#)
- UserOutputSelector
  - quickSpin, [430](#)
- UserOutputValue
  - quickSpin, [430](#)
- UserOutputValueAll
  - quickSpin, [430](#)
- UserOutputValueAllMask
  - quickSpin, [430](#)
- UserSetDefault
  - quickSpin, [430](#)
- UserSetFeatureEnable
  - quickSpin, [430](#)
- UserSetLoad
  - quickSpin, [430](#)
- UserSetSave
  - quickSpin, [431](#)
- UserSetSelector
  - quickSpin, [431](#)
- V3\_3Enable
  - quickSpin, [431](#)
- WhiteClip
  - quickSpin, [431](#)
- WhiteClipSelector
  - quickSpin, [431](#)
- Width
  - quickSpin, [431](#)
- width
  - spinH264Option, [456](#)
- WidthMax
  - quickSpin, [431](#)