

Spinnaker C

1.18.0.17

Generated by Doxygen 1.8.11

Contents

1	Introduction	1
2	Software Licensing Information	3
3	Module Index	5
3.1	Modules	5
4	Data Structure Index	7
4.1	Data Structures	7
5	File Index	9
5.1	File List	9
6	Module Documentation	11
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	43
6.2.2	Enumeration Type Documentation	43
6.2.2.1	spinAcquisitionModeEnums	43
6.2.2.2	spinAcquisitionStatusSelectorEnums	44

6.2.2.3	spinActionUnconditionalModeEnums	44
6.2.2.4	spinAdcBitDepthEnums	44
6.2.2.5	spinAutoAlgorithmSelectorEnums	44
6.2.2.6	spinAutoExposureControlPriorityEnums	45
6.2.2.7	spinAutoExposureLightingModeEnums	45
6.2.2.8	spinAutoExposureMeteringModeEnums	45
6.2.2.9	spinAutoExposureTargetGreyValueAutoEnums	46
6.2.2.10	spinBalanceRatioSelectorEnums	46
6.2.2.11	spinBalanceWhiteAutoEnums	46
6.2.2.12	spinBalanceWhiteAutoProfileEnums	46
6.2.2.13	spinBinningHorizontalModeEnums	47
6.2.2.14	spinBinningSelectorEnums	47
6.2.2.15	spinBinningVerticalModeEnums	47
6.2.2.16	spinBlackLevelAutoBalanceEnums	47
6.2.2.17	spinBlackLevelAutoEnums	48
6.2.2.18	spinBlackLevelSelectorEnums	48
6.2.2.19	spinChunkBlackLevelSelectorEnums	48
6.2.2.20	spinChunkCounterSelectorEnums	48
6.2.2.21	spinChunkEncoderSelectorEnums	49
6.2.2.22	spinChunkEncoderStatusEnums	49
6.2.2.23	spinChunkExposureTimeSelectorEnums	49
6.2.2.24	spinChunkGainSelectorEnums	50
6.2.2.25	spinChunkImageComponentEnums	50
6.2.2.26	spinChunkPixelFormatEnums	50
6.2.2.27	spinChunkRegionIDEnums	51
6.2.2.28	spinChunkScan3dCoordinateReferenceSelectorEnums	51
6.2.2.29	spinChunkScan3dCoordinateSelectorEnums	51
6.2.2.30	spinChunkScan3dCoordinateSystemEnums	51
6.2.2.31	spinChunkScan3dCoordinateSystemReferenceEnums	52
6.2.2.32	spinChunkScan3dCoordinateTransformSelectorEnums	52

6.2.2.33	spinChunkScan3dDistanceUnitEnums	52
6.2.2.34	spinChunkScan3dOutputModeEnums	53
6.2.2.35	spinChunkSelectorEnums	53
6.2.2.36	spinChunkSourceIDEnums	54
6.2.2.37	spinChunkTimerSelectorEnums	54
6.2.2.38	spinChunkTransferStreamIDEnums	54
6.2.2.39	spinCIConfigurationEnums	55
6.2.2.40	spinCITimeSlotsCountEnums	55
6.2.2.41	spinColorTransformationSelectorEnums	55
6.2.2.42	spinColorTransformationValueSelectorEnums	56
6.2.2.43	spinCounterEventActivationEnums	56
6.2.2.44	spinCounterEventSourceEnums	56
6.2.2.45	spinCounterResetActivationEnums	57
6.2.2.46	spinCounterResetSourceEnums	57
6.2.2.47	spinCounterSelectorEnums	58
6.2.2.48	spinCounterStatusEnums	58
6.2.2.49	spinCounterTriggerActivationEnums	58
6.2.2.50	spinCounterTriggerSourceEnums	59
6.2.2.51	spinCxpConnectionTestModeEnums	59
6.2.2.52	spinCxpLinkConfigurationEnums	59
6.2.2.53	spinCxpLinkConfigurationPreferredEnums	60
6.2.2.54	spinCxpLinkConfigurationStatusEnums	61
6.2.2.55	spinCxpPoCxpStatusEnums	62
6.2.2.56	spinDecimationHorizontalModeEnums	62
6.2.2.57	spinDecimationSelectorEnums	63
6.2.2.58	spinDecimationVerticalModeEnums	63
6.2.2.59	spinDefectCorrectionModeEnums	63
6.2.2.60	spinDeinterlacingEnums	63
6.2.2.61	spinDeviceCharacterSetEnums	64
6.2.2.62	spinDeviceClockSelectorEnums	64

6.2.2.63	spinDeviceConnectionStatusEnums	64
6.2.2.64	spinDeviceIndicatorModeEnums	64
6.2.2.65	spinDeviceLinkHeartbeatModeEnums	65
6.2.2.66	spinDeviceLinkThroughputLimitModeEnums	65
6.2.2.67	spinDevicePowerSupplySelectorEnums	65
6.2.2.68	spinDeviceRegistersEndiannessEnums	65
6.2.2.69	spinDeviceScanTypeEnums	65
6.2.2.70	spinDeviceSerialPortBaudRateEnums	66
6.2.2.71	spinDeviceSerialPortSelectorEnums	66
6.2.2.72	spinDeviceStreamChannelEndiannessEnums	66
6.2.2.73	spinDeviceStreamChannelTypeEnums	66
6.2.2.74	spinDeviceTapGeometryEnums	67
6.2.2.75	spinDeviceTemperatureSelectorEnums	68
6.2.2.76	spinDeviceTLTypeEnums	68
6.2.2.77	spinDeviceTypeEnums	68
6.2.2.78	spinEncoderModeEnums	69
6.2.2.79	spinEncoderOutputModeEnums	69
6.2.2.80	spinEncoderResetActivationEnums	69
6.2.2.81	spinEncoderResetSourceEnums	70
6.2.2.82	spinEncoderSelectorEnums	71
6.2.2.83	spinEncoderSourceAEnums	71
6.2.2.84	spinEncoderSourceBEnums	71
6.2.2.85	spinEncoderStatusEnums	71
6.2.2.86	spinEventNotificationEnums	72
6.2.2.87	spinEventSelectorEnums	72
6.2.2.88	spinExposureActiveModeEnums	72
6.2.2.89	spinExposureAutoEnums	72
6.2.2.90	spinExposureModeEnums	73
6.2.2.91	spinExposureTimeModeEnums	73
6.2.2.92	spinExposureTimeSelectorEnums	73

6.2.2.93	spinFileOpenModeEnums	74
6.2.2.94	spinFileOperationSelectorEnums	74
6.2.2.95	spinFileOperationStatusEnums	74
6.2.2.96	spinFileSelectorEnums	74
6.2.2.97	spinGainAutoBalanceEnums	75
6.2.2.98	spinGainAutoEnums	75
6.2.2.99	spinGainSelectorEnums	75
6.2.2.100	spinGevCCPEnums	75
6.2.2.101	spinGevCurrentPhysicalLinkConfigurationEnums	76
6.2.2.102	spinGevGVCPExtendedStatusCodesSelectorEnums	76
6.2.2.103	spinGevGVSPExtendedIDModeEnums	76
6.2.2.104	spinGevIEEE1588ClockAccuracyEnums	76
6.2.2.105	spinGevIEEE1588ModeEnums	77
6.2.2.106	spinGevIEEE1588StatusEnums	77
6.2.2.107	spinGevIPConfigurationStatusEnums	77
6.2.2.108	spinGevPhysicalLinkConfigurationEnums	77
6.2.2.109	spinGevSupportedOptionSelectorEnums	78
6.2.2.110	spinImageComponentSelectorEnums	78
6.2.2.111	spinImageCompressionJPEGFormatOptionEnums	79
6.2.2.112	spinImageCompressionModeEnums	79
6.2.2.113	spinImageCompressionRateOptionEnums	79
6.2.2.114	spinLineFormatEnums	80
6.2.2.115	spinLineInputFilterSelectorEnums	80
6.2.2.116	spinLineModeEnums	80
6.2.2.117	spinLineSelectorEnums	80
6.2.2.118	spinLineSourceEnums	81
6.2.2.119	spinLogicBlockLUTInputActivationEnums	81
6.2.2.120	spinLogicBlockLUTInputSelectorEnums	81
6.2.2.121	spinLogicBlockLUTInputSourceEnums	82
6.2.2.122	spinLogicBlockLUTSelectorEnums	82

6.2.2.123 spinLogicBlockSelectorEnums	82
6.2.2.124 spinLUTSelectorEnums	83
6.2.2.125 spinPixelColorFilterEnums	83
6.2.2.126 spinPixelFormatEnums	83
6.2.2.127 spinPixelFormatInfoSelectorEnums	89
6.2.2.128 spinPixelSizeEnums	94
6.2.2.129 spinRegionDestinationEnums	94
6.2.2.130 spinRegionModeEnums	95
6.2.2.131 spinRegionSelectorEnums	95
6.2.2.132 spinRgbTransformLightSourceEnums	95
6.2.2.133 spinScan3dCoordinateReferenceSelectorEnums	96
6.2.2.134 spinScan3dCoordinateSelectorEnums	96
6.2.2.135 spinScan3dCoordinateSystemEnums	96
6.2.2.136 spinScan3dCoordinateSystemReferenceEnums	96
6.2.2.137 spinScan3dCoordinateTransformSelectorEnums	97
6.2.2.138 spinScan3dDistanceUnitEnums	97
6.2.2.139 spinScan3dOutputModeEnums	97
6.2.2.140 spinSensorDigitizationTapsEnums	98
6.2.2.141 spinSensorShutterModeEnums	98
6.2.2.142 spinSensorTapsEnums	98
6.2.2.143 spinSequencerConfigurationModeEnums	99
6.2.2.144 spinSequencerConfigurationValidEnums	99
6.2.2.145 spinSequencerModeEnums	99
6.2.2.146 spinSequencerSetValidEnums	99
6.2.2.147 spinSequencerTriggerActivationEnums	99
6.2.2.148 spinSequencerTriggerSourceEnums	100
6.2.2.149 spinSerialPortBaudRateEnums	100
6.2.2.150 spinSerialPortParityEnums	100
6.2.2.151 spinSerialPortSelectorEnums	101
6.2.2.152 spinSerialPortSourceEnums	101

6.2.2.153 spinSerialPortStopBitsEnums	101
6.2.2.154 spinSoftwareSignalSelectorEnums	101
6.2.2.155 spinSourceSelectorEnums	102
6.2.2.156 spinTestPatternEnums	102
6.2.2.157 spinTestPatternGeneratorSelectorEnums	102
6.2.2.158 spinTimerSelectorEnums	102
6.2.2.159 spinTimerStatusEnums	103
6.2.2.160 spinTimerTriggerActivationEnums	103
6.2.2.161 spinTimerTriggerSourceEnums	103
6.2.2.162 spinTransferComponentSelectorEnums	104
6.2.2.163 spinTransferControlModeEnums	105
6.2.2.164 spinTransferOperationModeEnums	105
6.2.2.165 spinTransferQueueModeEnums	105
6.2.2.166 spinTransferSelectorEnums	106
6.2.2.167 spinTransferStatusSelectorEnums	106
6.2.2.168 spinTransferTriggerActivationEnums	106
6.2.2.169 spinTransferTriggerModeEnums	107
6.2.2.170 spinTransferTriggerSelectorEnums	107
6.2.2.171 spinTransferTriggerSourceEnums	107
6.2.2.172 spinTriggerActivationEnums	108
6.2.2.173 spinTriggerModeEnums	108
6.2.2.174 spinTriggerOverlapEnums	109
6.2.2.175 spinTriggerSelectorEnums	109
6.2.2.176 spinTriggerSourceEnums	109
6.2.2.177 spinUserOutputSelectorEnums	110
6.2.2.178 spinUserSetDefaultEnums	110
6.2.2.179 spinUserSetSelectorEnums	110
6.2.2.180 spinWhiteClipSelectorEnums	110
6.3 Chunk Data Structures	111
6.3.1 Detailed Description	111

6.4	Spinnaker C QuickSpin API	112
6.4.1	Detailed Description	112
6.5	QuickSpin Access	113
6.5.1	Detailed Description	113
6.5.2	Function Documentation	113
6.5.2.1	quickSpinInit(spinCamera hCamera, quickSpin *pQuickSpin)	113
6.5.2.2	quickSpinInitEx(spinCamera hCamera, quickSpin *pQuickSpin, quickSpinTL↔ Device *pQuickSpinTLDevice, quickSpinTLStream *pQuickSpinTLStream) . . .	113
6.5.2.3	quickSpinTLDeviceInit(spinCamera hCamera, quickSpinTLDevice *pQuick↔ SpinTLDevice)	113
6.5.2.4	quickSpinTLInterfaceInit(spinInterface hInterface, quickSpinTLInterface *p↔ QuickSpinTLInterface)	113
6.5.2.5	quickSpinTLStreamInit(spinCamera hCamera, quickSpinTLStream *pQuick↔ SpinTLStream)	113
6.6	Spinnaker C API	114
6.6.1	Detailed Description	115
6.6.2	Function Documentation	115
6.6.2.1	spinCameraDiscoverMaxPacketSize(spinCamera hCamera, unsigned int *p↔ MaxPacketSize)	115
6.7	Error Handling	116
6.7.1	Detailed Description	116
6.7.2	Function Documentation	116
6.7.2.1	spinErrorGetLast(spinError *pError)	116
6.7.2.2	spinErrorGetLastBuildDate(char *pBuf, size_t *pBufLen)	117
6.7.2.3	spinErrorGetLastBuildTime(char *pBuf, size_t *pBufLen)	117
6.7.2.4	spinErrorGetLastFileName(char *pBuf, size_t *pBufLen)	117
6.7.2.5	spinErrorGetLastFullMessage(char *pBuf, size_t *pBufLen)	118
6.7.2.6	spinErrorGetLastFunctionName(char *pBuf, size_t *pBufLen)	118
6.7.2.7	spinErrorGetLastLineNumber(int64_t *pLineNum)	119
6.7.2.8	spinErrorGetLastMessage(char *pBuf, size_t *pBufLen)	119
6.8	System Access	120
6.8.1	Detailed Description	121
6.8.2	Function Documentation	121

6.8.2.1	spinSystemGetCameras(spinSystem hSystem, spinCameraList hCameraList)	121
6.8.2.2	spinSystemGetCamerasEx(spinSystem hSystem, bool8_t bUpdateInterfaces, bool8_t bUpdateCameras, spinCameraList hCameraList)	122
6.8.2.3	spinSystemGetInstance(spinSystem *phSystem)	122
6.8.2.4	spinSystemGetInterfaces(spinSystem hSystem, spinInterfaceList hInterfaceList)	122
6.8.2.5	spinSystemGetLibraryVersion(spinSystem hSystem, spinLibraryVersion *hLibraryVersion)	123
6.8.2.6	spinSystemGetLoggingLevel(spinSystem hSystem, spinnakerLogLevel *pLogLevel)	123
6.8.2.7	spinSystemIsInUse(spinSystem hSystem, bool8_t *pbIsInUse)	123
6.8.2.8	spinSystemRegisterArrivalEvent(spinSystem hSystem, spinArrivalEvent hArrivalEvent)	124
6.8.2.9	spinSystemRegisterInterfaceEvent(spinSystem hSystem, spinInterfaceEvent hInterfaceEvent)	124
6.8.2.10	spinSystemRegisterLogEvent(spinSystem hSystem, spinLogEvent hLogEvent)	125
6.8.2.11	spinSystemRegisterRemovalEvent(spinSystem hSystem, spinRemovalEvent hRemovalEvent)	125
6.8.2.12	spinSystemReleaseInstance(spinSystem hSystem)	125
6.8.2.13	spinSystemSendActionCommand(spinSystem hSystem, size_t iDeviceKey, size_t iGroupKey, size_t iGroupMask, size_t iActionTime, size_t *piResultSize, actionCommandResult results[])	126
6.8.2.14	spinSystemSetLoggingLevel(spinSystem hSystem, spinnakerLogLevel logLevel)	126
6.8.2.15	spinSystemUnregisterAllLogEvents(spinSystem hSystem)	127
6.8.2.16	spinSystemUnregisterArrivalEvent(spinSystem hSystem, spinArrivalEvent hArrivalEvent)	127
6.8.2.17	spinSystemUnregisterInterfaceEvent(spinSystem hSystem, spinInterfaceEvent hInterfaceEvent)	127
6.8.2.18	spinSystemUnregisterLogEvent(spinSystem hSystem, spinLogEvent hLogEvent)	128
6.8.2.19	spinSystemUnregisterRemovalEvent(spinSystem hSystem, spinRemovalEvent hRemovalEvent)	128
6.8.2.20	spinSystemUpdateCameras(spinSystem hSystem, bool8_t *pbChanged)	129
6.8.2.21	spinSystemUpdateCamerasEx(spinSystem hSystem, bool8_t bUpdateInterfaces, bool8_t *pbChanged)	129
6.9	InterfaceList Access	130
6.9.1	Detailed Description	130
6.9.2	Function Documentation	130

6.9.2.1	<code>spinInterfaceListClear(spinInterfaceList hInterfaceList)</code>	130
6.9.2.2	<code>spinInterfaceListCreateEmpty(spinInterfaceList *phInterfaceList)</code>	131
6.9.2.3	<code>spinInterfaceListDestroy(spinInterfaceList hInterfaceList)</code>	131
6.9.2.4	<code>spinInterfaceListGet(spinInterfaceList hInterfaceList, size_t index, spinInterface *phInterface)</code>	131
6.9.2.5	<code>spinInterfaceListGetSize(spinInterfaceList hInterfaceList, size_t *pSize)</code>	132
6.10	CameraList Access	133
6.10.1	Detailed Description	133
6.10.2	Function Documentation	133
6.10.2.1	<code>spinCameraListAppend(spinCameraList hCameraListBase, spinCameraList h↔CameraListToAppend)</code>	133
6.10.2.2	<code>spinCameraListClear(spinCameraList hCameraList)</code>	134
6.10.2.3	<code>spinCameraListCreateEmpty(spinCameraList *phCameraList)</code>	134
6.10.2.4	<code>spinCameraListDestroy(spinCameraList hCameraList)</code>	134
6.10.2.5	<code>spinCameraListGet(spinCameraList hCameraList, size_t index, spinCamera *phCamera)</code>	135
6.10.2.6	<code>spinCameraListGetBySerial(spinCameraList hCameraList, const char *pSerial, spinCamera *phCamera)</code>	135
6.10.2.7	<code>spinCameraListGetSize(spinCameraList hCameraList, size_t *pSize)</code>	136
6.10.2.8	<code>spinCameraListRemove(spinCameraList hCameraList, size_t index)</code>	136
6.10.2.9	<code>spinCameraListRemoveBySerial(spinCameraList hCameraList, const char *p↔Serial)</code>	136
6.11	Interface Access	138
6.11.1	Detailed Description	139
6.11.2	Function Documentation	139
6.11.2.1	<code>spinInterfaceGetCameras(spinInterface hInterface, spinCameraList hCameraList)</code>	139
6.11.2.2	<code>spinInterfaceGetCamerasEx(spinInterface hInterface, bool8_t bUpdateCameras, spinCameraList hCameraList)</code>	139
6.11.2.3	<code>spinInterfaceGetTLNodeMap(spinInterface hInterface, spinNodeMapHandle *phNodeMap)</code>	140
6.11.2.4	<code>spinInterfaceIsInUse(spinInterface hInterface, bool8_t *pbIsInUse)</code>	140
6.11.2.5	<code>spinInterfaceRegisterArrivalEvent(spinInterface hInterface, spinArrivalEvent h↔ArrivalEvent)</code>	140

6.11.2.6	<code>spinInterfaceRegisterInterfaceEvent(spinInterface hInterface, spinInterfaceEvent hInterfaceEvent)</code>	141
6.11.2.7	<code>spinInterfaceRegisterRemovalEvent(spinInterface hInterface, spinRemovalEvent hRemovalEvent)</code>	141
6.11.2.8	<code>spinInterfaceRelease(spinInterface hInterface)</code>	142
6.11.2.9	<code>spinInterfaceSendActionCommand(spinInterface hInterface, size_t iDeviceKey, size_t iGroupKey, size_t iGroupMask, size_t iActionTime, size_t *piResultSize, actionCommandResult results[])</code>	142
6.11.2.10	<code>spinInterfaceUnregisterArrivalEvent(spinInterface hInterface, spinArrivalEvent hArrivalEvent)</code>	142
6.11.2.11	<code>spinInterfaceUnregisterInterfaceEvent(spinInterface hInterface, spinInterface↵Event hInterfaceEvent)</code>	143
6.11.2.12	<code>spinInterfaceUnregisterRemovalEvent(spinInterface hInterface, spinRemoval↵Event hRemovalEvent)</code>	143
6.11.2.13	<code>spinInterfaceUpdateCameras(spinInterface hInterface, bool8_t *pbChanged)</code>	144
6.12	Camera Access	145
6.12.1	Detailed Description	146
6.12.2	Function Documentation	146
6.12.2.1	<code>spinCameraBeginAcquisition(spinCamera hCamera)</code>	146
6.12.2.2	<code>spinCameraDeInit(spinCamera hCamera)</code>	146
6.12.2.3	<code>spinCameraEndAcquisition(spinCamera hCamera)</code>	147
6.12.2.4	<code>spinCameraGetAccessMode(spinCamera hCamera, spinAccessMode *p↵AccessMode)</code>	147
6.12.2.5	<code>spinCameraGetGuiXml(spinCamera hCamera, char *pBuf, size_t *pBufLen)</code>	148
6.12.2.6	<code>spinCameraGetNextImage(spinCamera hCamera, spinImage *phImage)</code>	148
6.12.2.7	<code>spinCameraGetNextImageEx(spinCamera hCamera, uint64_t grabTimeout, spinImage *phImage)</code>	148
6.12.2.8	<code>spinCameraGetNodeMap(spinCamera hCamera, spinNodeMapHandle *ph↵NodeMap)</code>	149
6.12.2.9	<code>spinCameraGetTLDeviceNodeMap(spinCamera hCamera, spinNodeMapHandle *phNodeMap)</code>	149
6.12.2.10	<code>spinCameraGetTLStreamNodeMap(spinCamera hCamera, spinNodeMapHandle *phNodeMap)</code>	150
6.12.2.11	<code>spinCameraGetUniqueID(spinCamera hCamera, char *pBuf, size_t *pBufLen)</code>	150
6.12.2.12	<code>spinCameraInit(spinCamera hCamera)</code>	150
6.12.2.13	<code>spinCameraIsInitialized(spinCamera hCamera, bool8_t *pbInit)</code>	151

6.12.2.14	<code>spinCamerasStreaming(spinCamera hCamera, bool8_t *pbIsStreaming)</code>	151
6.12.2.15	<code>spinCamerasValid(spinCamera hCamera, bool8_t *pbValid)</code>	151
6.12.2.16	<code>spinCameraReadPort(spinCamera hCamera, uint64_t iAddress, void *pBuffer, size_t iSize)</code>	152
6.12.2.17	<code>spinCameraRegisterDeviceEvent(spinCamera hCamera, spinDeviceEvent h↔ DeviceEvent)</code>	152
6.12.2.18	<code>spinCameraRegisterDeviceEventEx(spinCamera hCamera, spinDeviceEvent h↔ DeviceEvent, const char *pName)</code>	152
6.12.2.19	<code>spinCameraRegisterImageEvent(spinCamera hCamera, spinImageEvent h↔ ImageEvent)</code>	153
6.12.2.20	<code>spinCameraRelease(spinCamera hCamera)</code>	153
6.12.2.21	<code>spinCameraUnregisterDeviceEvent(spinCamera hCamera, spinDeviceEvent h↔ DeviceEvent)</code>	153
6.12.2.22	<code>spinCameraUnregisterImageEvent(spinCamera hCamera, spinImageEvent h↔ ImageEvent)</code>	154
6.12.2.23	<code>spinCameraWritePort(spinCamera hCamera, uint64_t iAddress, void *pBuffer, size_t iSize)</code>	154
6.13	Image Access	155
6.13.1	Detailed Description	157
6.13.2	Function Documentation	157
6.13.2.1	<code>spinImageCalculateStatistics(spinImage hImage, const spinImageStatistics h↔ Statistics)</code>	157
6.13.2.2	<code>spinImageCheckCRC(spinImage hImage, bool8_t *pbCheckCRC)</code>	158
6.13.2.3	<code>spinImageConvert(spinImage hSrcImage, spinPixelFormatEnums pixelFormat, spinImage hDestImage)</code>	158
6.13.2.4	<code>spinImageConvertEx(spinImage hSrcImage, spinPixelFormatEnums pixelFormat, spinColorProcessingAlgorithm algorithm, spinImage hDestImage)</code>	158
6.13.2.5	<code>spinImageCreate(spinImage hSrcImage, spinImage *phDestImage)</code>	159
6.13.2.6	<code>spinImageCreateEmpty(spinImage *phImage)</code>	159
6.13.2.7	<code>spinImageCreateEx(spinImage *phImage, size_t width, size_t height, size_t↔ offsetX, size_t offsetY, spinPixelFormatEnums pixelFormat, void *pData)</code>	160
6.13.2.8	<code>spinImageDeepCopy(spinImage hSrcImage, spinImage hDestImage)</code>	160
6.13.2.9	<code>spinImageDestroy(spinImage hImage)</code>	160
6.13.2.10	<code>spinImageGetBitsPerPixel(spinImage hImage, size_t *pBitsPerPixel)</code>	161
6.13.2.11	<code>spinImageGetBufferSize(spinImage hImage, size_t *pSize)</code>	161

6.13.2.12 spinImageGetChunkLayoutID(spinImage hImage, uint64_t *pId)	161
6.13.2.13 spinImageGetColorProcessing(spinImage hImage, spinColorProcessingAlgorithm *pAlgorithm)	162
6.13.2.14 spinImageGetData(spinImage hImage, void **ppData)	162
6.13.2.15 spinImageGetDefaultColorProcessing(spinColorProcessingAlgorithm *pAlgorithm)	163
6.13.2.16 spinImageGetFrameID(spinImage hImage, uint64_t *pFrameID)	163
6.13.2.17 spinImageGetHeight(spinImage hImage, size_t *pHeight)	163
6.13.2.18 spinImageGetID(spinImage hImage, uint64_t *pId)	164
6.13.2.19 spinImageGetOffsetX(spinImage hImage, size_t *pOffsetX)	164
6.13.2.20 spinImageGetOffsetY(spinImage hImage, size_t *pOffsetY)	164
6.13.2.21 spinImageGetPaddingX(spinImage hImage, size_t *pPaddingX)	165
6.13.2.22 spinImageGetPaddingY(spinImage hImage, size_t *pPaddingY)	165
6.13.2.23 spinImageGetPayloadType(spinImage hImage, size_t *pPayloadType)	166
6.13.2.24 spinImageGetPixelFormat(spinImage hImage, spinPixelFormatEnums *pPixelFormat)	166
6.13.2.25 spinImageGetPixelFormatName(spinImage hImage, char *pBuf, size_t *pBufLen)	166
6.13.2.26 spinImageGetPrivateData(spinImage hImage, void **ppData)	167
6.13.2.27 spinImageGetSize(spinImage hImage, size_t *pImageSize)	167
6.13.2.28 spinImageGetStatus(spinImage hImage, spinImageStatus *pStatus)	168
6.13.2.29 spinImageGetStatusDescription(spinImageStatus status, char *pBuf, size_t *pBufLen)	168
6.13.2.30 spinImageGetStride(spinImage hImage, size_t *pStride)	168
6.13.2.31 spinImageGetTimeStamp(spinImage hImage, uint64_t *pTimeStamp)	169
6.13.2.32 spinImageGetTLPayloadType(spinImage hImage, spinPayloadTypeInfoIds *pPayloadType)	169
6.13.2.33 spinImageGetTLPixelFormat(spinImage hImage, uint64_t *pPixelFormat)	170
6.13.2.34 spinImageGetTLPixelFormatNamespace(spinImage hImage, spinPixelFormatNamespaceID *pPixelFormatNamespace)	170
6.13.2.35 spinImageGetValidPayloadSize(spinImage hImage, size_t *pSize)	170
6.13.2.36 spinImageGetWidth(spinImage hImage, size_t *pWidth)	171
6.13.2.37 spinImageHasCRC(spinImage hImage, bool8_t *pbHasCRC)	171
6.13.2.38 spinImageIsIncomplete(spinImage hImage, bool8_t *pbIsIncomplete)	171
6.13.2.39 spinImageRelease(spinImage hImage)	172

6.13.2.40	<code>spinImageReset(spinImage hImage, size_t width, size_t height, size_t offsetX, size_t offsetY, spinPixelFormatEnums pixelFormat)</code>	172
6.13.2.41	<code>spinImageResetEx(spinImage hImage, size_t width, size_t height, size_t offsetX, size_t offsetY, spinPixelFormatEnums pixelFormat, void *pData)</code>	173
6.13.2.42	<code>spinImageSave(spinImage hImage, const char *pFilename, spinImageFileFormat format)</code>	173
6.13.2.43	<code>spinImageSaveBmp(spinImage hImage, const char *pFilename, const spinBMPOption *pOption)</code>	174
6.13.2.44	<code>spinImageSaveFromExt(spinImage hImage, const char *pFilename)</code>	174
6.13.2.45	<code>spinImageSaveJpeg(spinImage hImage, const char *pFilename, const spinJPEGOption *pOption)</code>	174
6.13.2.46	<code>spinImageSaveJpg2(spinImage hImage, const char *pFilename, const spinJPEG2Option *pOption)</code>	175
6.13.2.47	<code>spinImageSavePgm(spinImage hImage, const char *pFilename, const spinPGMOption *pOption)</code>	175
6.13.2.48	<code>spinImageSavePng(spinImage hImage, const char *pFilename, const spinPNGOption *pOption)</code>	176
6.13.2.49	<code>spinImageSavePpm(spinImage hImage, const char *pFilename, const spinPPMOption *pOption)</code>	176
6.13.2.50	<code>spinImageSaveTiff(spinImage hImage, const char *pFilename, const spinTIFFOption *pOption)</code>	176
6.13.2.51	<code>spinImageSetDefaultColorProcessing(spinColorProcessingAlgorithm algorithm)</code>	177
6.14	Event Access	178
6.14.1	Detailed Description	178
6.14.2	Function Documentation	178
6.14.2.1	<code>spinArrivalEventCreate(spinArrivalEvent *phArrivalEvent, spinArrivalEventFunction pFunction, void *pUserData)</code>	178
6.14.2.2	<code>spinArrivalEventDestroy(spinArrivalEvent hArrivalEvent)</code>	179
6.14.2.3	<code>spinDeviceEventCreate(spinDeviceEvent *phDeviceEvent, spinDeviceEventFunction pFunction, void *pUserData)</code>	179
6.14.2.4	<code>spinDeviceEventDestroy(spinDeviceEvent hDeviceEvent)</code>	180
6.14.2.5	<code>spinImageEventCreate(spinImageEvent *phImageEvent, spinImageEventFunction pFunction, void *pUserData)</code>	180
6.14.2.6	<code>spinImageEventDestroy(spinImageEvent hImageEvent)</code>	180
6.14.2.7	<code>spinInterfaceEventCreate(spinInterfaceEvent *phInterfaceEvent, spinArrivalEventFunction pArrivalFunction, spinRemovalEventFunction pRemovalFunction, void *pUserData)</code>	181

6.14.2.8	<code>spinInterfaceEventDestroy(spinInterfaceEvent hInterfaceEvent)</code>	181
6.14.2.9	<code>spinLogEventCreate(spinLogEvent *phLogEvent, spinLogEventFunction pFunction, void *pUserData)</code>	182
6.14.2.10	<code>spinLogEventDestroy(spinLogEvent hLogEvent)</code>	182
6.14.2.11	<code>spinRemovalEventCreate(spinRemovalEvent *phRemovalEvent, spinRemovalEventFunction pFunction, void *pUserData)</code>	182
6.14.2.12	<code>spinRemovalEventDestroy(spinRemovalEvent hRemovalEvent)</code>	183
6.15	ImageStatistics Access	184
6.15.1	Detailed Description	184
6.15.2	Function Documentation	185
6.15.2.1	<code>spinImageStatisticsCreate(spinImageStatistics *phStatistics)</code>	185
6.15.2.2	<code>spinImageStatisticsDestroy(spinImageStatistics hStatistics)</code>	185
6.15.2.3	<code>spinImageStatisticsDisableAll(spinImageStatistics hStatistics)</code>	185
6.15.2.4	<code>spinImageStatisticsEnableAll(spinImageStatistics hStatistics)</code>	186
6.15.2.5	<code>spinImageStatisticsEnableGreyOnly(spinImageStatistics hStatistics)</code>	186
6.15.2.6	<code>spinImageStatisticsEnableHslOnly(spinImageStatistics hStatistics)</code>	186
6.15.2.7	<code>spinImageStatisticsEnableRgbOnly(spinImageStatistics hStatistics)</code>	187
6.15.2.8	<code>spinImageStatisticsGetAll(spinImageStatistics hStatistics, spinStatisticsChannel channel, unsigned int *pRangeMin, unsigned int *pRangeMax, unsigned int *pPixelValueMin, unsigned int *pPixelValueMax, unsigned int *pNumPixelValues, float *pPixelValueMean, int **ppHistogram)</code>	187
6.15.2.9	<code>spinImageStatisticsGetChannelStatus(spinImageStatistics hStatistics, spinStatisticsChannel channel, bool8_t *pbEnabled)</code>	188
6.15.2.10	<code>spinImageStatisticsGetHistogram(spinImageStatistics hStatistics, spinStatisticsChannel channel, int **ppHistogram)</code>	188
6.15.2.11	<code>spinImageStatisticsGetMean(spinImageStatistics hStatistics, spinStatisticsChannel channel, float *pMean)</code>	188
6.15.2.12	<code>spinImageStatisticsGetNumPixelValues(spinImageStatistics hStatistics, spinStatisticsChannel channel, unsigned int *pNumValues)</code>	189
6.15.2.13	<code>spinImageStatisticsGetPixelValueRange(spinImageStatistics hStatistics, spinStatisticsChannel channel, unsigned int *pMin, unsigned int *pMax)</code>	189
6.15.2.14	<code>spinImageStatisticsGetRange(spinImageStatistics hStatistics, spinStatisticsChannel channel, unsigned int *pMin, unsigned int *pMax)</code>	190
6.15.2.15	<code>spinImageStatisticsSetChannelStatus(spinImageStatistics hStatistics, spinStatisticsChannel channel, bool8_t bEnable)</code>	190
6.16	Logging Event Data Access	191

6.16.1 Detailed Description	191
6.16.2 Function Documentation	191
6.16.2.1 spinLogDataGetCategoryName(spinLogEventData hLogEventData, char *pBuf, size_t *pBufLen)	191
6.16.2.2 spinLogDataGetLogMessage(spinLogEventData hLogEventData, char *pBuf, size_t *pBufLen)	192
6.16.2.3 spinLogDataGetNDC(spinLogEventData hLogEventData, char *pBuf, size_t *pBufLen)	192
6.16.2.4 spinLogDataGetPriority(spinLogEventData hLogEventData, int64_t *pValue)	192
6.16.2.5 spinLogDataGetPriorityName(spinLogEventData hLogEventData, char *pBuf, size_t *pBufLen)	193
6.16.2.6 spinLogDataGetThreadName(spinLogEventData hLogEventData, char *pBuf, size_t *pBufLen)	193
6.16.2.7 spinLogDataGetTimestamp(spinLogEventData hLogEventData, char *pBuf, size_t *pBufLen)	194
6.17 Device Event Data Access	195
6.17.1 Detailed Description	195
6.17.2 Function Documentation	195
6.17.2.1 spinDeviceEventGetId(spinDeviceEventData hDeviceEventData, uint64_t *pEventId)	195
6.17.2.2 spinDeviceEventGetName(spinDeviceEventData hDeviceEventData, char *pBuf, size_t *pBufLen)	195
6.17.2.3 spinDeviceEventGetPayloadData(spinDeviceEventData hDeviceEventData, const uint8_t *pBuf, size_t *pBufSize)	196
6.17.2.4 spinDeviceEventGetPayloadDataSize(spinDeviceEventData hDeviceEventData, size_t *pBufSize)	196
6.18 AVIRecorder Access	198
6.18.1 Detailed Description	198
6.18.2 Function Documentation	198
6.18.2.1 SPINNAKER_API_DEPRECATED("""spinAVIRecorderOpenUncompressed is deprecated, use spinVideoOpenUncompressed instead.""", spinAVIRecorderOpenUncompressed(spinAVIRecorder *phRecorder, const char *pName, spinAVIOption option))	198
6.18.2.2 SPINNAKER_API_DEPRECATED("""spinAVIRecorderOpenMJPEG is deprecated, use spinVideoOpenMJPEG instead.""", spinAVIRecorderOpenMJPEG(spinAVIRecorder *phRecorder, const char *pName, spinMJPEGOption option))	198

6.18.2.3	SPINNAKERC_API_DEPRECATED("spinAVIRecorderOpenH264 is deprecated, use spinVideoOpenH264 instead.", spinAVIRecorderOpenH264(spinAVIRecorder *phRecorder, const char *pName, spinH264Option option))	198
6.18.2.4	SPINNAKERC_API_DEPRECATED("spinAVIRecorderAppend is deprecated, use spinVideoAppend instead.", spinAVIRecorderAppend(spinAVIRecorder hRecorder, spinImage hImage))	198
6.18.2.5	SPINNAKERC_API_DEPRECATED("spinAVISetMaximumSize is deprecated, use spinVideoSetMaximumFileSize instead.", spinAVISetMaximumSize(spinAVIRecorder hRecorder, unsigned int size))	198
6.18.2.6	SPINNAKERC_API_DEPRECATED("spinAVIRecorderClose is deprecated, use spinVideoClose instead.", spinAVIRecorderClose(spinAVIRecorder hRecorder))	199
6.19	Chunk data access	200
6.19.1	Detailed Description	200
6.19.2	Function Documentation	200
6.19.2.1	spinImageChunkDataGetFloatValue(spinImage hImage, const char *pName, double *pValue)	200
6.19.2.2	spinImageChunkDataGetIntValue(spinImage hImage, const char *pName, int64_t *pValue)	200
6.20	Spinnaker C Handles	201
6.20.1	Detailed Description	202
6.20.2	Typedef Documentation	202
6.20.2.1	spinArrivalEvent	202
6.20.2.2	spinAVIRecorder	202
6.20.2.3	spinCamera	202
6.20.2.4	spinCameraList	202
6.20.2.5	spinDeviceEvent	202
6.20.2.6	spinDeviceEventData	202
6.20.2.7	spinImage	203
6.20.2.8	spinImageEvent	203
6.20.2.9	spinImageStatistics	203
6.20.2.10	spinInterface	203
6.20.2.11	spinInterfaceEvent	203
6.20.2.12	spinInterfaceList	203
6.20.2.13	spinLogEvent	203

6.20.2.14 spinLogEventData	204
6.20.2.15 spinRemovalEvent	204
6.20.2.16 spinSystem	204
6.20.2.17 spinVideo	204
6.21 Spinnaker C Function Signatures	205
6.21.1 Detailed Description	205
6.21.2 Typedef Documentation	205
6.21.2.1 spinArrivalEventFunction	205
6.21.2.2 spinDeviceEventFunction	205
6.21.2.3 spinImageEventFunction	205
6.21.2.4 spinLogEventFunction	205
6.21.2.5 spinRemovalEventFunction	205
6.22 Spinnaker C Enumerations	206
6.22.1 Detailed Description	209
6.22.2 Enumeration Type Documentation	209
6.22.2.1 spinColorProcessingAlgorithm	209
6.22.2.2 spinError	210
6.22.2.3 spinImageFileFormat	211
6.22.2.4 spinImageStatus	211
6.22.2.5 spinnakerLogLevel	212
6.22.2.6 spinPayloadTypeInfoIDs	212
6.22.2.7 spinPixelFormatNamespaceID	212
6.22.2.8 spinStatisticsChannel	213
6.23 Spinnaker C Structures	214
6.23.1 Detailed Description	215
6.23.2 Enumeration Type Documentation	215
6.23.2.1 actionCommandStatus	215
6.23.2.2 spinCompressionMethod	215
6.24 Spinnaker C GenICam API	216
6.24.1 Detailed Description	217

6.25 Node Map Access	218
6.25.1 Detailed Description	218
6.25.2 Function Documentation	218
6.25.2.1 spinNodeMapGetNode(spinNodeMapHandle hNodeMap, const char *pName, spinNodeHandle *phNode)	218
6.25.2.2 spinNodeMapGetNodeByIndex(spinNodeMapHandle hNodeMap, size_t index, spinNodeHandle *phNode)	219
6.25.2.3 spinNodeMapGetNumNodes(spinNodeMapHandle hNodeMap, size_t *pValue)	219
6.25.2.4 spinNodeMapPoll(spinNodeMapHandle hNodeMap, int64_t timestamp)	219
6.26 Node Access	221
6.26.1 Detailed Description	222
6.26.2 Function Documentation	222
6.26.2.1 spinNodeDeregisterCallback(spinNodeHandle hNode, spinNodeCallbackHandle hCb)	222
6.26.2.2 spinNodeGetAccessMode(spinNodeHandle hNode, spinAccessMode *p↔ AccessMode)	222
6.26.2.3 spinNodeGetCachingMode(spinNodeHandle hNode, spinCachingMode *p↔ CachingMode)	223
6.26.2.4 spinNodeGetDescription(spinNodeHandle hNode, char *pBuf, size_t *pBufLen)	223
6.26.2.5 spinNodeGetDisplayName(spinNodeHandle hNode, char *pBuf, size_t *pBufLen)	224
6.26.2.6 spinNodeGetImposedAccessMode(spinNodeHandle hNode, spinAccessMode imposedAccessMode)	224
6.26.2.7 spinNodeGetImposedVisibility(spinNodeHandle hNode, spinVisibility imposed↔ Visibility)	224
6.26.2.8 spinNodeGetName(spinNodeHandle hNode, char *pBuf, size_t *pBufLen)	225
6.26.2.9 spinNodeGetNameSpace(spinNodeHandle hNode, spinNameSpace *p↔ Namespace)	225
6.26.2.10 spinNodeGetPollingTime(spinNodeHandle hNode, int64_t *pPollingTime)	226
6.26.2.11 spinNodeGetToolTip(spinNodeHandle hNode, char *pBuf, size_t *pBufLen)	226
6.26.2.12 spinNodeGetType(spinNodeHandle hNode, spinNodeType *pType)	226
6.26.2.13 spinNodeGetVisibility(spinNodeHandle hNode, spinVisibility *pVisibility)	227
6.26.2.14 spinNodeInvalidateNode(spinNodeHandle hNode)	227
6.26.2.15 spinNodesAvailable(spinNodeHandle hNode, bool8_t *pbResult)	227

6.26.2.16	<code>spinNodesEqual(spinNodeHandle hNodeFirst, spinNodeHandle hNodeSecond, bool8_t *pbResult)</code>	228
6.26.2.17	<code>spinNodesImplemented(spinNodeHandle hNode, bool8_t *pbResult)</code>	228
6.26.2.18	<code>spinNodesReadable(spinNodeHandle hNode, bool8_t *pbResult)</code>	229
6.26.2.19	<code>spinNodesWritable(spinNodeHandle hNode, bool8_t *pbResult)</code>	229
6.26.2.20	<code>spinNodeRegisterCallback(spinNodeHandle hNode, spinNodeCallbackFunction pCbFunction, spinNodeCallbackHandle *phCb)</code>	229
6.27	IValue Access	231
6.27.1	Detailed Description	231
6.27.2	Function Documentation	231
6.27.2.1	<code>spinNodeFromString(spinNodeHandle hNode, const char *pBuf)</code>	231
6.27.2.2	<code>spinNodeFromStringEx(spinNodeHandle hNode, bool8_t bVerify, const char *pBuf)</code>	232
6.27.2.3	<code>spinNodeToString(spinNodeHandle hNode, char *pBuf, size_t *pBufLen)</code>	232
6.27.2.4	<code>spinNodeToStringEx(spinNodeHandle hNode, bool8_t bVerify, char *pBuf, size_t *pBufLen)</code>	233
6.28	String Access	234
6.28.1	Detailed Description	234
6.28.2	Function Documentation	234
6.28.2.1	<code>spinStringGetMaxLength(spinNodeHandle hNode, int64_t *pValue)</code>	234
6.28.2.2	<code>spinStringGetValue(spinNodeHandle hNode, char *pBuf, size_t *pBufLen)</code>	235
6.28.2.3	<code>spinStringGetValueEx(spinNodeHandle hNode, bool8_t bVerify, char *pBuf, size_t *pBufLen)</code>	235
6.28.2.4	<code>spinStringSetValue(spinNodeHandle hNode, const char *pBuf)</code>	236
6.28.2.5	<code>spinStringSetValueEx(spinNodeHandle hNode, bool8_t bVerify, const char *pBuf)</code>	236
6.29	Integer Access	237
6.29.1	Detailed Description	237
6.29.2	Function Documentation	237
6.29.2.1	<code>spinIntegerGetInc(spinNodeHandle hNode, int64_t *pValue)</code>	237
6.29.2.2	<code>spinIntegerGetMax(spinNodeHandle hNode, int64_t *pValue)</code>	238
6.29.2.3	<code>spinIntegerGetMin(spinNodeHandle hNode, int64_t *pValue)</code>	238
6.29.2.4	<code>spinIntegerGetRepresentation(spinNodeHandle hNode, spinRepresentation *pValue)</code>	239
6.29.2.5	<code>spinIntegerGetValue(spinNodeHandle hNode, int64_t *pValue)</code>	239

6.29.2.6	<code>spinIntegerGetValueEx(spinNodeHandle hNode, bool8_t bVerify, int64_t *pValue)</code>	239
6.29.2.7	<code>spinIntegerSetValue(spinNodeHandle hNode, int64_t value)</code>	240
6.29.2.8	<code>spinIntegerSetValueEx(spinNodeHandle hNode, bool8_t bVerify, int64_t value)</code>	240
6.30	IFloat Access	241
6.30.1	Detailed Description	241
6.30.2	Function Documentation	241
6.30.2.1	<code>spinFloatGetMax(spinNodeHandle hNode, double *pValue)</code>	241
6.30.2.2	<code>spinFloatGetMin(spinNodeHandle hNode, double *pValue)</code>	242
6.30.2.3	<code>spinFloatGetRepresentation(spinNodeHandle hNode, spinRepresentation *pValue)</code>	242
6.30.2.4	<code>spinFloatGetUnit(spinNodeHandle hNode, char *pBuf, size_t *pBufLen)</code>	243
6.30.2.5	<code>spinFloatGetValue(spinNodeHandle hNode, double *pValue)</code>	243
6.30.2.6	<code>spinFloatGetValueEx(spinNodeHandle hNode, bool8_t bVerify, double *pValue)</code>	243
6.30.2.7	<code>spinFloatSetValue(spinNodeHandle hNode, double value)</code>	244
6.30.2.8	<code>spinFloatSetValueEx(spinNodeHandle hNode, bool8_t bVerify, double value)</code>	244
6.31	IEnumeration Access	245
6.31.1	Detailed Description	245
6.31.2	Function Documentation	245
6.31.2.1	<code>spinEnumerationGetCurrentEntry(spinNodeHandle hNode, spinNodeHandle *phEntry)</code>	245
6.31.2.2	<code>spinEnumerationGetEntryByIndex(spinNodeHandle hNode, size_t index, spinNodeHandle *phEntry)</code>	246
6.31.2.3	<code>spinEnumerationGetEntryByName(spinNodeHandle hNode, const char *pName, spinNodeHandle *phEntry)</code>	246
6.31.2.4	<code>spinEnumerationGetNumEntries(spinNodeHandle hNode, size_t *pValue)</code>	247
6.31.2.5	<code>spinEnumerationSetEnumValue(spinNodeHandle hNode, size_t value)</code>	247
6.31.2.6	<code>spinEnumerationSetIntValue(spinNodeHandle hNode, int64_t value)</code>	247
6.32	IEnumEntry Access	249
6.32.1	Detailed Description	249
6.32.2	Function Documentation	249
6.32.2.1	<code>spinEnumerationEntryGetEnumValue(spinNodeHandle hNode, size_t *pValue)</code>	249
6.32.2.2	<code>spinEnumerationEntryGetIntValue(spinNodeHandle hNode, int64_t *pValue)</code>	250

6.32.2.3	spinEnumerationEntryGetSymbolic(spinNodeHandle hNode, char *pBuf, size_t *pBufLen)	250
6.33	IBoolean Access	251
6.33.1	Detailed Description	251
6.33.2	Function Documentation	251
6.33.2.1	spinBooleanGetValue(spinNodeHandle hNode, bool8_t *pbValue)	251
6.33.2.2	spinBooleanSetValue(spinNodeHandle hNode, bool8_t value)	252
6.34	ICommand Access	253
6.34.1	Detailed Description	253
6.34.2	Function Documentation	253
6.34.2.1	spinCommandExecute(spinNodeHandle hNode)	253
6.34.2.2	spinCommandIsDone(spinNodeHandle hNode, bool8_t *pbValue)	254
6.35	ICategory Access	255
6.35.1	Detailed Description	255
6.35.2	Function Documentation	255
6.35.2.1	spinCategoryGetFeatureByIndex(spinNodeHandle hNode, size_t index, spinNodeHandle *phFeature)	255
6.35.2.2	spinCategoryGetNumFeatures(spinNodeHandle hNode, size_t *pValue)	256
6.36	IRegister Access	257
6.36.1	Detailed Description	257
6.36.2	Function Documentation	257
6.36.2.1	spinRegisterGet(spinNodeHandle hNode, uint8_t *pBuf, int64_t length)	257
6.36.2.2	spinRegisterGetAddress(spinNodeHandle hNode, int64_t *pAddress)	258
6.36.2.3	spinRegisterGetEx(spinNodeHandle hNode, bool8_t bVerify, bool8_t bIgnoreCache, uint8_t *pBuf, int64_t length)	258
6.36.2.4	spinRegisterGetLength(spinNodeHandle hNode, int64_t *pLength)	259
6.36.2.5	spinRegisterSet(spinNodeHandle hNode, const uint8_t *pBuf, int64_t length)	259
6.36.2.6	spinRegisterSetEx(spinNodeHandle hNode, bool8_t bVerify, const uint8_t *pBuf, int64_t length)	259
6.36.2.7	spinRegisterSetReference(spinNodeHandle hNode, spinNodeHandle hRef)	260
6.37	Spinnaker C GenICam Handles	261
6.37.1	Detailed Description	261

6.37.2	Typedef Documentation	261
6.37.2.1	spinNodeCallbackFunction	261
6.37.2.2	spinNodeCallbackHandle	261
6.37.2.3	spinNodeHandle	261
6.37.2.4	spinNodeMapHandle	261
6.38	Spinnaker C GenICam Enumerations	262
6.38.1	Detailed Description	264
6.38.2	Enumeration Type Documentation	264
6.38.2.1	spinAccessMode	264
6.38.2.2	spinCachingMode	265
6.38.2.3	spinDisplayNotation	265
6.38.2.4	spinEndianess	265
6.38.2.5	spinIncMode	265
6.38.2.6	spinInputDirection	266
6.38.2.7	spinInterfaceType	266
6.38.2.8	spinLinkType	266
6.38.2.9	spinNameSpace	267
6.38.2.10	spinNodeType	267
6.38.2.11	spinRepresentation	267
6.38.2.12	spinSign	268
6.38.2.13	spinSlope	268
6.38.2.14	spinStandardNameSpace	268
6.38.2.15	spinVisibility	268
6.38.2.16	spinXMLValidation	269
6.38.2.17	spinYesNo	269
6.39	SpinVideo Recording Access	270
6.39.1	Detailed Description	270
6.39.2	Function Documentation	270
6.39.2.1	spinVideoAppend(spinVideo hSpinVideo, spinImage hImage)	270
6.39.2.2	spinVideoClose(spinVideo hSpinVideo)	270

6.39.2.3	<code>spinVideoOpenH264(spinVideo *phSpinVideo, const char *pName, spinH264↵ Option option)</code>	270
6.39.2.4	<code>spinVideoOpenMJPEG(spinVideo *phSpinVideo, const char *pName, spinMJP↵ GOption option)</code>	270
6.39.2.5	<code>spinVideoOpenUncompressed(spinVideo *phSpinVideo, const char *pName, spinAVIOption option)</code>	270
6.39.2.6	<code>spinVideoSetMaximumFileSize(spinVideo hSpinVideo, unsigned int size)</code>	270
6.40	Transport Layer Enumerations	272
6.40.1	Detailed Description	273
6.40.2	Enumeration Type Documentation	273
6.40.2.1	<code>spinTLDeviceAccessStatusEnums</code>	273
6.40.2.2	<code>spinTLDeviceCurrentSpeedEnums</code>	274
6.40.2.3	<code>spinTLDeviceEndiannessMechanismEnums</code>	274
6.40.2.4	<code>spinTLDeviceTypeEnums</code>	274
6.40.2.5	<code>spinTLGenICamXMLLocationEnums</code>	275
6.40.2.6	<code>spinTLGevCCPEnums</code>	275
6.40.2.7	<code>spinTLGUIXMLLocationEnums</code>	275
6.40.2.8	<code>spinTLPOEStatusEnums</code>	275
6.40.2.9	<code>spinTLStreamBufferCountModeEnums</code>	276
6.40.2.10	<code>spinTLStreamBufferHandlingModeEnums</code>	276
6.40.2.11	<code>spinTLStreamDefaultBufferCountModeEnums</code>	276
6.40.2.12	<code>spinTLStreamTypeEnums</code>	277
6.41	TLDevice Structures	278
6.41.1	Detailed Description	278
6.42	TLInterface Structures	279
6.42.1	Detailed Description	279
6.43	TLStream Structures	280
6.43.1	Detailed Description	280

7	Data Structure Documentation	281
7.1	actionCommandResult Struct Reference	281
7.1.1	Detailed Description	281
7.1.2	Field Documentation	281
7.1.2.1	DeviceAddress	281
7.1.2.2	Status	281
7.2	quickSpin Struct Reference	282
7.2.1	Field Documentation	294
7.2.1.1	AasRoiEnable	294
7.2.1.2	AasRoiHeight	294
7.2.1.3	AasRoiOffsetX	294
7.2.1.4	AasRoiOffsetY	294
7.2.1.5	AasRoiWidth	294
7.2.1.6	AcquisitionAbort	294
7.2.1.7	AcquisitionArm	294
7.2.1.8	AcquisitionBurstFrameCount	294
7.2.1.9	AcquisitionFrameCount	294
7.2.1.10	AcquisitionFrameRate	294
7.2.1.11	AcquisitionFrameRateEnable	294
7.2.1.12	AcquisitionLineRate	294
7.2.1.13	AcquisitionMode	294
7.2.1.14	AcquisitionResultingFrameRate	294
7.2.1.15	AcquisitionStart	294
7.2.1.16	AcquisitionStatus	295
7.2.1.17	AcquisitionStatusSelector	295
7.2.1.18	AcquisitionStop	295
7.2.1.19	ActionDeviceKey	295
7.2.1.20	ActionGroupKey	295
7.2.1.21	ActionGroupMask	295
7.2.1.22	ActionQueueSize	295

7.2.1.23	ActionSelector	295
7.2.1.24	ActionUnconditionalMode	295
7.2.1.25	AdaptiveCompressionEnable	295
7.2.1.26	AdcBitDepth	295
7.2.1.27	aPAUSEMACCtrlFramesReceived	295
7.2.1.28	aPAUSEMACCtrlFramesTransmitted	295
7.2.1.29	AutoAlgorithmSelector	295
7.2.1.30	AutoExposureControlLoopDamping	295
7.2.1.31	AutoExposureControlPriority	295
7.2.1.32	AutoExposureEVCompensation	295
7.2.1.33	AutoExposureExposureTimeLowerLimit	295
7.2.1.34	AutoExposureExposureTimeUpperLimit	295
7.2.1.35	AutoExposureGainLowerLimit	295
7.2.1.36	AutoExposureGainUpperLimit	295
7.2.1.37	AutoExposureGreyValueLowerLimit	295
7.2.1.38	AutoExposureGreyValueUpperLimit	295
7.2.1.39	AutoExposureLightingMode	296
7.2.1.40	AutoExposureMeteringMode	296
7.2.1.41	AutoExposureTargetGreyValue	296
7.2.1.42	AutoExposureTargetGreyValueAuto	296
7.2.1.43	BalanceRatio	296
7.2.1.44	BalanceRatioSelector	296
7.2.1.45	BalanceWhiteAuto	296
7.2.1.46	BalanceWhiteAutoDamping	296
7.2.1.47	BalanceWhiteAutoLowerLimit	296
7.2.1.48	BalanceWhiteAutoProfile	296
7.2.1.49	BalanceWhiteAutoUpperLimit	296
7.2.1.50	BinningHorizontal	296
7.2.1.51	BinningHorizontalMode	296
7.2.1.52	BinningSelector	296

7.2.1.53	BinningVertical	296
7.2.1.54	BinningVerticalMode	296
7.2.1.55	BlackLevel	296
7.2.1.56	BlackLevelAuto	296
7.2.1.57	BlackLevelAutoBalance	296
7.2.1.58	BlackLevelClampingEnable	296
7.2.1.59	BlackLevelRaw	296
7.2.1.60	BlackLevelSelector	296
7.2.1.61	ChunkBlackLevel	296
7.2.1.62	ChunkBlackLevelSelector	297
7.2.1.63	ChunkCounterSelector	297
7.2.1.64	ChunkCounterValue	297
7.2.1.65	ChunkCRC	297
7.2.1.66	ChunkEnable	297
7.2.1.67	ChunkEncoderSelector	297
7.2.1.68	ChunkEncoderStatus	297
7.2.1.69	ChunkEncoderValue	297
7.2.1.70	ChunkExposureEndLineStatusAll	297
7.2.1.71	ChunkExposureTime	297
7.2.1.72	ChunkExposureTimeSelector	297
7.2.1.73	ChunkFrameID	297
7.2.1.74	ChunkGain	297
7.2.1.75	ChunkGainSelector	297
7.2.1.76	ChunkHeight	297
7.2.1.77	ChunkImage	297
7.2.1.78	ChunkImageComponent	297
7.2.1.79	ChunkInferenceConfidence	297
7.2.1.80	ChunkInferenceResult	297
7.2.1.81	ChunkLinePitch	297
7.2.1.82	ChunkLineStatusAll	297

7.2.1.83	ChunkModeActive	297
7.2.1.84	ChunkOffsetX	297
7.2.1.85	ChunkOffsetY	298
7.2.1.86	ChunkPartSelector	298
7.2.1.87	ChunkPixelDynamicRangeMax	298
7.2.1.88	ChunkPixelDynamicRangeMin	298
7.2.1.89	ChunkPixelFormat	298
7.2.1.90	ChunkRegionID	298
7.2.1.91	ChunkScan3dAxisMax	298
7.2.1.92	ChunkScan3dAxisMin	298
7.2.1.93	ChunkScan3dCoordinateOffset	298
7.2.1.94	ChunkScan3dCoordinateReferenceSelector	298
7.2.1.95	ChunkScan3dCoordinateReferenceValue	298
7.2.1.96	ChunkScan3dCoordinateScale	298
7.2.1.97	ChunkScan3dCoordinateSelector	298
7.2.1.98	ChunkScan3dCoordinateSystem	298
7.2.1.99	ChunkScan3dCoordinateSystemReference	298
7.2.1.100	ChunkScan3dCoordinateTransformSelector	298
7.2.1.101	ChunkScan3dDistanceUnit	298
7.2.1.102	ChunkScan3dInvalidDataFlag	298
7.2.1.103	ChunkScan3dInvalidDataValue	298
7.2.1.104	ChunkScan3dOutputMode	298
7.2.1.105	ChunkScan3dTransformValue	298
7.2.1.106	ChunkScanLineSelector	298
7.2.1.107	ChunkSelector	298
7.2.1.108	ChunkSequencerSetActive	299
7.2.1.109	ChunkSerialData	299
7.2.1.110	ChunkSerialDataLength	299
7.2.1.111	ChunkSerialReceiveOverflow	299
7.2.1.112	ChunkSourceID	299

7.2.1.113 ChunkStreamChannelID	299
7.2.1.114 ChunkTimerSelector	299
7.2.1.115 ChunkTimerValue	299
7.2.1.116 ChunkTimestamp	299
7.2.1.117 ChunkTimestampLatchValue	299
7.2.1.118 ChunkTransferBlockID	299
7.2.1.119 ChunkTransferQueueCurrentBlockCount	299
7.2.1.120 ChunkTransferStreamID	299
7.2.1.121 ChunkWidth	299
7.2.1.122 CIConfiguration	299
7.2.1.123 CITimeSlotsCount	299
7.2.1.124 ColorTransformationEnable	299
7.2.1.125 ColorTransformationSelector	299
7.2.1.126 ColorTransformationValue	299
7.2.1.127 ColorTransformationValueSelector	299
7.2.1.128 CompressionRatio	299
7.2.1.129 CounterDelay	299
7.2.1.130 CounterDuration	299
7.2.1.131 CounterEventActivation	300
7.2.1.132 CounterEventSource	300
7.2.1.133 CounterReset	300
7.2.1.134 CounterResetActivation	300
7.2.1.135 CounterResetSource	300
7.2.1.136 CounterSelector	300
7.2.1.137 CounterStatus	300
7.2.1.138 CounterTriggerActivation	300
7.2.1.139 CounterTriggerSource	300
7.2.1.140 CounterValue	300
7.2.1.141 CounterValueAtReset	300
7.2.1.142 CxpConnectionSelector	300

7.2.1.143 CxpConnectionTestErrorCount	300
7.2.1.144 CxpConnectionTestMode	300
7.2.1.145 CxpConnectionTestPacketCount	300
7.2.1.146 CxpLinkConfiguration	300
7.2.1.147 CxpLinkConfigurationPreferred	300
7.2.1.148 CxpLinkConfigurationStatus	300
7.2.1.149 CxpPoCxpAuto	300
7.2.1.150 CxpPoCxpStatus	300
7.2.1.151 CxpPoCxpTripReset	300
7.2.1.152 CxpPoCxpTurnOff	300
7.2.1.153 DecimationHorizontal	300
7.2.1.154 DecimationHorizontalMode	301
7.2.1.155 DecimationSelector	301
7.2.1.156 DecimationVertical	301
7.2.1.157 DecimationVerticalMode	301
7.2.1.158 DefectCorrectionMode	301
7.2.1.159 DefectCorrectStaticEnable	301
7.2.1.160 DefectTableApply	301
7.2.1.161 DefectTableCoordinateX	301
7.2.1.162 DefectTableCoordinateY	301
7.2.1.163 DefectTableFactoryRestore	301
7.2.1.164 DefectTableIndex	301
7.2.1.165 DefectTablePixelCount	301
7.2.1.166 DefectTableSave	301
7.2.1.167 Deinterlacing	301
7.2.1.168 DeviceCharacterSet	301
7.2.1.169 DeviceClockFrequency	301
7.2.1.170 DeviceClockSelector	301
7.2.1.171 DeviceConnectionSelector	301
7.2.1.172 DeviceConnectionSpeed	301

7.2.1.173 DeviceConnectionStatus	301
7.2.1.174 DeviceEventChannelCount	301
7.2.1.175 DeviceFamilyName	301
7.2.1.176 DeviceFeaturePersistenceEnd	301
7.2.1.177 DeviceFeaturePersistenceStart	302
7.2.1.178 DeviceFirmwareVersion	302
7.2.1.179 DeviceGenCPVersionMajor	302
7.2.1.180 DeviceGenCPVersionMinor	302
7.2.1.181 DeviceID	302
7.2.1.182 DeviceIndicatorMode	302
7.2.1.183 DeviceLinkBandwidthReserve	302
7.2.1.184 DeviceLinkCommandTimeout	302
7.2.1.185 DeviceLinkConnectionCount	302
7.2.1.186 DeviceLinkCurrentThroughput	302
7.2.1.187 DeviceLinkHeartbeatMode	302
7.2.1.188 DeviceLinkHeartbeatTimeout	302
7.2.1.189 DeviceLinkSelector	302
7.2.1.190 DeviceLinkSpeed	302
7.2.1.191 DeviceLinkThroughputLimit	302
7.2.1.192 DeviceLinkThroughputLimitMode	302
7.2.1.193 DeviceManifestEntrySelector	302
7.2.1.194 DeviceManifestPrimaryURL	302
7.2.1.195 DeviceManifestSchemaMajorVersion	302
7.2.1.196 DeviceManifestSchemaMinorVersion	302
7.2.1.197 DeviceManifestSecondaryURL	302
7.2.1.198 DeviceManifestXMLMajorVersion	302
7.2.1.199 DeviceManifestXMLMinorVersion	302
7.2.1.200 DeviceManifestXMLSubMinorVersion	303
7.2.1.201 DeviceManufacturerInfo	303
7.2.1.202 DeviceMaxThroughput	303

7.2.1.203 DeviceModelName	303
7.2.1.204 DevicePowerSupplySelector	303
7.2.1.205 DeviceRegistersCheck	303
7.2.1.206 DeviceRegistersEndianness	303
7.2.1.207 DeviceRegistersStreamingEnd	303
7.2.1.208 DeviceRegistersStreamingStart	303
7.2.1.209 DeviceRegistersValid	303
7.2.1.210 DeviceReset	303
7.2.1.211 DeviceScanType	303
7.2.1.212 DeviceSerialNumber	303
7.2.1.213 DeviceSerialPortBaudRate	303
7.2.1.214 DeviceSerialPortSelector	303
7.2.1.215 DeviceSFNCVersionMajor	303
7.2.1.216 DeviceSFNCVersionMinor	303
7.2.1.217 DeviceSFNCVersionSubMinor	303
7.2.1.218 DeviceStreamChannelCount	303
7.2.1.219 DeviceStreamChannelEndianness	303
7.2.1.220 DeviceStreamChannelLink	303
7.2.1.221 DeviceStreamChannelPacketSize	303
7.2.1.222 DeviceStreamChannelSelector	303
7.2.1.223 DeviceStreamChannelType	304
7.2.1.224 DeviceTapGeometry	304
7.2.1.225 DeviceTemperature	304
7.2.1.226 DeviceTemperatureSelector	304
7.2.1.227 DeviceTLType	304
7.2.1.228 DeviceTLVersionMajor	304
7.2.1.229 DeviceTLVersionMinor	304
7.2.1.230 DeviceTLVersionSubMinor	304
7.2.1.231 DeviceType	304
7.2.1.232 DeviceUptime	304

7.2.1.233 DeviceUserID	304
7.2.1.234 DeviceVendorName	304
7.2.1.235 DeviceVersion	304
7.2.1.236 EncoderDivider	304
7.2.1.237 EncoderMode	304
7.2.1.238 EncoderOutputMode	304
7.2.1.239 EncoderReset	304
7.2.1.240 EncoderResetActivation	304
7.2.1.241 EncoderResetSource	304
7.2.1.242 EncoderSelector	304
7.2.1.243 EncoderSourceA	304
7.2.1.244 EncoderSourceB	304
7.2.1.245 EncoderStatus	304
7.2.1.246 EncoderTimeout	305
7.2.1.247 EncoderValue	305
7.2.1.248 EncoderValueAtReset	305
7.2.1.249 EnumerationCount	305
7.2.1.250 EventAcquisitionEnd	305
7.2.1.251 EventAcquisitionEndFrameID	305
7.2.1.252 EventAcquisitionEndTimestamp	305
7.2.1.253 EventAcquisitionError	305
7.2.1.254 EventAcquisitionErrorFrameID	305
7.2.1.255 EventAcquisitionErrorTimestamp	305
7.2.1.256 EventAcquisitionStart	305
7.2.1.257 EventAcquisitionStartFrameID	305
7.2.1.258 EventAcquisitionStartTimestamp	305
7.2.1.259 EventAcquisitionTransferEnd	305
7.2.1.260 EventAcquisitionTransferEndFrameID	305
7.2.1.261 EventAcquisitionTransferEndTimestamp	305
7.2.1.262 EventAcquisitionTransferStart	305

7.2.1.263 EventAcquisitionTransferStartFrameID	305
7.2.1.264 EventAcquisitionTransferStartTimestamp	305
7.2.1.265 EventAcquisitionTrigger	305
7.2.1.266 EventAcquisitionTriggerFrameID	305
7.2.1.267 EventAcquisitionTriggerTimestamp	305
7.2.1.268 EventActionLate	305
7.2.1.269 EventActionLateFrameID	306
7.2.1.270 EventActionLateTimestamp	306
7.2.1.271 EventCounter0End	306
7.2.1.272 EventCounter0EndFrameID	306
7.2.1.273 EventCounter0EndTimestamp	306
7.2.1.274 EventCounter0Start	306
7.2.1.275 EventCounter0StartFrameID	306
7.2.1.276 EventCounter0StartTimestamp	306
7.2.1.277 EventCounter1End	306
7.2.1.278 EventCounter1EndFrameID	306
7.2.1.279 EventCounter1EndTimestamp	306
7.2.1.280 EventCounter1Start	306
7.2.1.281 EventCounter1StartFrameID	306
7.2.1.282 EventCounter1StartTimestamp	306
7.2.1.283 EventEncoder0Restarted	306
7.2.1.284 EventEncoder0RestartedFrameID	306
7.2.1.285 EventEncoder0RestartedTimestamp	306
7.2.1.286 EventEncoder0Stopped	306
7.2.1.287 EventEncoder0StoppedFrameID	306
7.2.1.288 EventEncoder0StoppedTimestamp	306
7.2.1.289 EventEncoder1Restarted	306
7.2.1.290 EventEncoder1RestartedFrameID	306
7.2.1.291 EventEncoder1RestartedTimestamp	306
7.2.1.292 EventEncoder1Stopped	307

7.2.1.293 EventEncoder1StoppedFrameID	307
7.2.1.294 EventEncoder1StoppedTimestamp	307
7.2.1.295 EventError	307
7.2.1.296 EventErrorCode	307
7.2.1.297 EventErrorFrameID	307
7.2.1.298 EventErrorTimestamp	307
7.2.1.299 EventExposureEnd	307
7.2.1.300 EventExposureEndFrameID	307
7.2.1.301 EventExposureEndTimestamp	307
7.2.1.302 EventExposureStart	307
7.2.1.303 EventExposureStartFrameID	307
7.2.1.304 EventExposureStartTimestamp	307
7.2.1.305 EventFrameBurstEnd	307
7.2.1.306 EventFrameBurstEndFrameID	307
7.2.1.307 EventFrameBurstEndTimestamp	307
7.2.1.308 EventFrameBurstStart	307
7.2.1.309 EventFrameBurstStartFrameID	307
7.2.1.310 EventFrameBurstStartTimestamp	307
7.2.1.311 EventFrameEnd	307
7.2.1.312 EventFrameEndFrameID	307
7.2.1.313 EventFrameEndTimestamp	307
7.2.1.314 EventFrameStart	307
7.2.1.315 EventFrameStartFrameID	308
7.2.1.316 EventFrameStartTimestamp	308
7.2.1.317 EventFrameTransferEnd	308
7.2.1.318 EventFrameTransferEndFrameID	308
7.2.1.319 EventFrameTransferEndTimestamp	308
7.2.1.320 EventFrameTransferStart	308
7.2.1.321 EventFrameTransferStartFrameID	308
7.2.1.322 EventFrameTransferStartTimestamp	308

7.2.1.323 EventFrameTrigger	308
7.2.1.324 EventFrameTriggerFrameID	308
7.2.1.325 EventFrameTriggerTimestamp	308
7.2.1.326 EventLine0AnyEdge	308
7.2.1.327 EventLine0AnyEdgeFrameID	308
7.2.1.328 EventLine0AnyEdgeTimestamp	308
7.2.1.329 EventLine0FallingEdge	308
7.2.1.330 EventLine0FallingEdgeFrameID	308
7.2.1.331 EventLine0FallingEdgeTimestamp	308
7.2.1.332 EventLine0RisingEdge	308
7.2.1.333 EventLine0RisingEdgeFrameID	308
7.2.1.334 EventLine0RisingEdgeTimestamp	308
7.2.1.335 EventLine1AnyEdge	308
7.2.1.336 EventLine1AnyEdgeFrameID	308
7.2.1.337 EventLine1AnyEdgeTimestamp	308
7.2.1.338 EventLine1FallingEdge	309
7.2.1.339 EventLine1FallingEdgeFrameID	309
7.2.1.340 EventLine1FallingEdgeTimestamp	309
7.2.1.341 EventLine1RisingEdge	309
7.2.1.342 EventLine1RisingEdgeFrameID	309
7.2.1.343 EventLine1RisingEdgeTimestamp	309
7.2.1.344 EventLinkSpeedChange	309
7.2.1.345 EventLinkSpeedChangeFrameID	309
7.2.1.346 EventLinkSpeedChangeTimestamp	309
7.2.1.347 EventLinkTrigger0	309
7.2.1.348 EventLinkTrigger0FrameID	309
7.2.1.349 EventLinkTrigger0Timestamp	309
7.2.1.350 EventLinkTrigger1	309
7.2.1.351 EventLinkTrigger1FrameID	309
7.2.1.352 EventLinkTrigger1Timestamp	309

7.2.1.353 EventNotification	309
7.2.1.354 EventSelector	309
7.2.1.355 EventSequencerSetChange	309
7.2.1.356 EventSequencerSetChangeFrameID	309
7.2.1.357 EventSequencerSetChangeTimestamp	309
7.2.1.358 EventSerialData	309
7.2.1.359 EventSerialDataLength	309
7.2.1.360 EventSerialPortReceive	309
7.2.1.361 EventSerialPortReceiveTimestamp	310
7.2.1.362 EventSerialReceiveOverflow	310
7.2.1.363 EventStream0TransferBlockEnd	310
7.2.1.364 EventStream0TransferBlockEndFrameID	310
7.2.1.365 EventStream0TransferBlockEndTimestamp	310
7.2.1.366 EventStream0TransferBlockStart	310
7.2.1.367 EventStream0TransferBlockStartFrameID	310
7.2.1.368 EventStream0TransferBlockStartTimestamp	310
7.2.1.369 EventStream0TransferBlockTrigger	310
7.2.1.370 EventStream0TransferBlockTriggerFrameID	310
7.2.1.371 EventStream0TransferBlockTriggerTimestamp	310
7.2.1.372 EventStream0TransferBurstEnd	310
7.2.1.373 EventStream0TransferBurstEndFrameID	310
7.2.1.374 EventStream0TransferBurstEndTimestamp	310
7.2.1.375 EventStream0TransferBurstStart	310
7.2.1.376 EventStream0TransferBurstStartFrameID	310
7.2.1.377 EventStream0TransferBurstStartTimestamp	310
7.2.1.378 EventStream0TransferEnd	310
7.2.1.379 EventStream0TransferEndFrameID	310
7.2.1.380 EventStream0TransferEndTimestamp	310
7.2.1.381 EventStream0TransferOverflow	310
7.2.1.382 EventStream0TransferOverflowFrameID	310

7.2.1.383 EventStream0TransferOverflowTimestamp	310
7.2.1.384 EventStream0TransferPause	311
7.2.1.385 EventStream0TransferPauseFrameID	311
7.2.1.386 EventStream0TransferPauseTimestamp	311
7.2.1.387 EventStream0TransferResume	311
7.2.1.388 EventStream0TransferResumeFrameID	311
7.2.1.389 EventStream0TransferResumeTimestamp	311
7.2.1.390 EventStream0TransferStart	311
7.2.1.391 EventStream0TransferStartFrameID	311
7.2.1.392 EventStream0TransferStartTimestamp	311
7.2.1.393 EventTest	311
7.2.1.394 EventTestTimestamp	311
7.2.1.395 EventTimer0End	311
7.2.1.396 EventTimer0EndFrameID	311
7.2.1.397 EventTimer0EndTimestamp	311
7.2.1.398 EventTimer0Start	311
7.2.1.399 EventTimer0StartFrameID	311
7.2.1.400 EventTimer0StartTimestamp	311
7.2.1.401 EventTimer1End	311
7.2.1.402 EventTimer1EndFrameID	311
7.2.1.403 EventTimer1EndTimestamp	311
7.2.1.404 EventTimer1Start	311
7.2.1.405 EventTimer1StartFrameID	311
7.2.1.406 EventTimer1StartTimestamp	311
7.2.1.407 ExposureActiveMode	312
7.2.1.408 ExposureAuto	312
7.2.1.409 ExposureMode	312
7.2.1.410 ExposureTime	312
7.2.1.411 ExposureTimeMode	312
7.2.1.412 ExposureTimeSelector	312

7.2.1.413 FactoryReset	312
7.2.1.414 FileAccessBuffer	312
7.2.1.415 FileAccessLength	312
7.2.1.416 FileAccessOffset	312
7.2.1.417 FileOpenMode	312
7.2.1.418 FileOperationExecute	312
7.2.1.419 FileOperationResult	312
7.2.1.420 FileOperationSelector	312
7.2.1.421 FileOperationStatus	312
7.2.1.422 FileSelector	312
7.2.1.423 FileSize	312
7.2.1.424 Gain	312
7.2.1.425 GainAuto	312
7.2.1.426 GainAutoBalance	312
7.2.1.427 GainSelector	312
7.2.1.428 Gamma	312
7.2.1.429 GammaEnable	312
7.2.1.430 GevActiveLinkCount	313
7.2.1.431 GevCCP	313
7.2.1.432 GevCurrentDefaultGateway	313
7.2.1.433 GevCurrentIPAddress	313
7.2.1.434 GevCurrentIPConfigurationDHCP	313
7.2.1.435 GevCurrentIPConfigurationLLA	313
7.2.1.436 GevCurrentIPConfigurationPersistentIP	313
7.2.1.437 GevCurrentPhysicalLinkConfiguration	313
7.2.1.438 GevCurrentSubnetMask	313
7.2.1.439 GevDiscoveryAckDelay	313
7.2.1.440 GevFirstURL	313
7.2.1.441 GevGVCPEExtendedStatusCodes	313
7.2.1.442 GevGVCPEExtendedStatusCodesSelector	313

7.2.1.443	GevGVCPHeartbeatDisable	313
7.2.1.444	GevGVCPPendingAck	313
7.2.1.445	GevGVCPPendingTimeout	313
7.2.1.446	GevGVSPExtendedIDMode	313
7.2.1.447	GevHeartbeatTimeout	313
7.2.1.448	GevIEEE1588	313
7.2.1.449	GevIEEE1588ClockAccuracy	313
7.2.1.450	GevIEEE1588Mode	313
7.2.1.451	GevIEEE1588Status	313
7.2.1.452	GevInterfaceSelector	313
7.2.1.453	GevIPConfigurationStatus	314
7.2.1.454	GevMACAddress	314
7.2.1.455	GevMCDA	314
7.2.1.456	GevMCPHostPort	314
7.2.1.457	GevMCRC	314
7.2.1.458	GevMCSP	314
7.2.1.459	GevMCTT	314
7.2.1.460	GevNumberOfInterfaces	314
7.2.1.461	GevPAUSEFrameReception	314
7.2.1.462	GevPAUSEFrameTransmission	314
7.2.1.463	GevPersistentDefaultGateway	314
7.2.1.464	GevPersistentIPAddress	314
7.2.1.465	GevPersistentSubnetMask	314
7.2.1.466	GevPhysicalLinkConfiguration	314
7.2.1.467	GevPrimaryApplicationIPAddress	314
7.2.1.468	GevPrimaryApplicationSocket	314
7.2.1.469	GevPrimaryApplicationSwitchoverKey	314
7.2.1.470	GevSCCFGAllInTransmission	314
7.2.1.471	GevSCCFGExtendedChunkData	314
7.2.1.472	GevSCCFGPacketResendDestination	314

7.2.1.473	GevSCCFGUnconditionalStreaming	314
7.2.1.474	GevSCDA	314
7.2.1.475	GevSCPD	314
7.2.1.476	GevSCPDirection	315
7.2.1.477	GevSCPHostPort	315
7.2.1.478	GevSCPInterfaceIndex	315
7.2.1.479	GevSCPSBigEndian	315
7.2.1.480	GevSCPSDoNotFragment	315
7.2.1.481	GevSCPSFireTestPacket	315
7.2.1.482	GevSCPSPacketSize	315
7.2.1.483	GevSCSP	315
7.2.1.484	GevSCZoneConfigurationLock	315
7.2.1.485	GevSCZoneCount	315
7.2.1.486	GevSCZoneDirectionAll	315
7.2.1.487	GevSecondURL	315
7.2.1.488	GevStreamChannelSelector	315
7.2.1.489	GevSupportedOption	315
7.2.1.490	GevSupportedOptionSelector	315
7.2.1.491	GevTimestampTickFrequency	315
7.2.1.492	GuiXmlManifestAddress	315
7.2.1.493	Height	315
7.2.1.494	HeightMax	315
7.2.1.495	ImageComponentEnable	315
7.2.1.496	ImageComponentSelector	315
7.2.1.497	ImageCompressionBitrate	315
7.2.1.498	ImageCompressionJPEGFormatOption	315
7.2.1.499	ImageCompressionMode	316
7.2.1.500	ImageCompressionQuality	316
7.2.1.501	ImageCompressionRateOption	316
7.2.1.502	IspEnable	316

7.2.1.503 LineFilterWidth	316
7.2.1.504 LineFormat	316
7.2.1.505 LineInputFilterSelector	316
7.2.1.506 LineInverter	316
7.2.1.507 LineMode	316
7.2.1.508 LinePitch	316
7.2.1.509 LineSelector	316
7.2.1.510 LineSource	316
7.2.1.511 LineStatus	316
7.2.1.512 LineStatusAll	316
7.2.1.513 LinkErrorCount	316
7.2.1.514 LinkUptime	316
7.2.1.515 LogicBlockLUTInputActivation	316
7.2.1.516 LogicBlockLUTInputSelector	316
7.2.1.517 LogicBlockLUTInputSource	316
7.2.1.518 LogicBlockLUTOutputValue	316
7.2.1.519 LogicBlockLUTOutputValueAll	316
7.2.1.520 LogicBlockLUTRowIndex	316
7.2.1.521 LogicBlockLUTSelector	316
7.2.1.522 LogicBlockSelector	317
7.2.1.523 LUTEnable	317
7.2.1.524 LUTIndex	317
7.2.1.525 LUTSelector	317
7.2.1.526 LUTValue	317
7.2.1.527 LUTValueAll	317
7.2.1.528 MaxDeviceResetTime	317
7.2.1.529 OffsetX	317
7.2.1.530 OffsetY	317
7.2.1.531 PacketResendRequestCount	317
7.2.1.532 PayloadSize	317

7.2.1.533 PixelColorFilter	317
7.2.1.534 PixelDynamicRangeMax	317
7.2.1.535 PixelDynamicRangeMin	317
7.2.1.536 PixelFormat	317
7.2.1.537 PixelFormatInfoID	317
7.2.1.538 PixelFormatInfoSelector	317
7.2.1.539 PixelSize	317
7.2.1.540 PowerSupplyCurrent	317
7.2.1.541 PowerSupplyVoltage	317
7.2.1.542 RegionDestination	317
7.2.1.543 RegionMode	317
7.2.1.544 RegionSelector	317
7.2.1.545 ReverseX	318
7.2.1.546 ReverseY	318
7.2.1.547 RgbTransformLightSource	318
7.2.1.548 Saturation	318
7.2.1.549 SaturationEnable	318
7.2.1.550 Scan3dAxisMax	318
7.2.1.551 Scan3dAxisMin	318
7.2.1.552 Scan3dCoordinateOffset	318
7.2.1.553 Scan3dCoordinateReferenceSelector	318
7.2.1.554 Scan3dCoordinateReferenceValue	318
7.2.1.555 Scan3dCoordinateScale	318
7.2.1.556 Scan3dCoordinateSelector	318
7.2.1.557 Scan3dCoordinateSystem	318
7.2.1.558 Scan3dCoordinateSystemReference	318
7.2.1.559 Scan3dCoordinateTransformSelector	318
7.2.1.560 Scan3dDistanceUnit	318
7.2.1.561 Scan3dInvalidDataFlag	318
7.2.1.562 Scan3dInvalidDataValue	318

7.2.1.563 Scan3dOutputMode	318
7.2.1.564 Scan3dTransformValue	318
7.2.1.565 SensorDescription	318
7.2.1.566 SensorDigitizationTaps	318
7.2.1.567 SensorHeight	318
7.2.1.568 SensorShutterMode	319
7.2.1.569 SensorTaps	319
7.2.1.570 SensorWidth	319
7.2.1.571 SequencerConfigurationMode	319
7.2.1.572 SequencerConfigurationValid	319
7.2.1.573 SequencerFeatureEnable	319
7.2.1.574 SequencerMode	319
7.2.1.575 SequencerPathSelector	319
7.2.1.576 SequencerSetActive	319
7.2.1.577 SequencerSetLoad	319
7.2.1.578 SequencerSetNext	319
7.2.1.579 SequencerSetSave	319
7.2.1.580 SequencerSetSelector	319
7.2.1.581 SequencerSetStart	319
7.2.1.582 SequencerSetValid	319
7.2.1.583 SequencerTriggerActivation	319
7.2.1.584 SequencerTriggerSource	319
7.2.1.585 SerialPortBaudRate	319
7.2.1.586 SerialPortDataBits	319
7.2.1.587 SerialPortParity	319
7.2.1.588 SerialPortSelector	319
7.2.1.589 SerialPortSource	319
7.2.1.590 SerialPortStopBits	319
7.2.1.591 SerialReceiveFramingErrorCount	320
7.2.1.592 SerialReceiveParityErrorCount	320

7.2.1.593 SerialReceiveQueueClear	320
7.2.1.594 SerialReceiveQueueCurrentCharacterCount	320
7.2.1.595 SerialReceiveQueueMaxCharacterCount	320
7.2.1.596 SerialTransmitQueueCurrentCharacterCount	320
7.2.1.597 SerialTransmitQueueMaxCharacterCount	320
7.2.1.598 Sharpening	320
7.2.1.599 SharpeningAuto	320
7.2.1.600 SharpeningEnable	320
7.2.1.601 SharpeningThreshold	320
7.2.1.602 SoftwareSignalPulse	320
7.2.1.603 SoftwareSignalSelector	320
7.2.1.604 SourceCount	320
7.2.1.605 SourceSelector	320
7.2.1.606 Test0001	320
7.2.1.607 TestEventGenerate	320
7.2.1.608 TestPattern	320
7.2.1.609 TestPatternGeneratorSelector	320
7.2.1.610 TestPendingAck	320
7.2.1.611 TimerDelay	320
7.2.1.612 TimerDuration	320
7.2.1.613 TimerReset	320
7.2.1.614 TimerSelector	321
7.2.1.615 TimerStatus	321
7.2.1.616 TimerTriggerActivation	321
7.2.1.617 TimerTriggerSource	321
7.2.1.618 TimerValue	321
7.2.1.619 Timestamp	321
7.2.1.620 TimestampLatch	321
7.2.1.621 TimestampLatchValue	321
7.2.1.622 TimestampReset	321

7.2.1.623 TLPParamsLocked	321
7.2.1.624 TransferAbort	321
7.2.1.625 TransferBlockCount	321
7.2.1.626 TransferBurstCount	321
7.2.1.627 TransferComponentSelector	321
7.2.1.628 TransferControlMode	321
7.2.1.629 TransferOperationMode	321
7.2.1.630 TransferPause	321
7.2.1.631 TransferQueueCurrentBlockCount	321
7.2.1.632 TransferQueueMaxBlockCount	321
7.2.1.633 TransferQueueMode	321
7.2.1.634 TransferQueueOverflowCount	321
7.2.1.635 TransferResume	321
7.2.1.636 TransferSelector	321
7.2.1.637 TransferStart	322
7.2.1.638 TransferStatus	322
7.2.1.639 TransferStatusSelector	322
7.2.1.640 TransferStop	322
7.2.1.641 TransferStreamChannel	322
7.2.1.642 TransferTriggerActivation	322
7.2.1.643 TransferTriggerMode	322
7.2.1.644 TransferTriggerSelector	322
7.2.1.645 TransferTriggerSource	322
7.2.1.646 TriggerActivation	322
7.2.1.647 TriggerDelay	322
7.2.1.648 TriggerDivider	322
7.2.1.649 TriggerEventTest	322
7.2.1.650 TriggerMode	322
7.2.1.651 TriggerMultiplier	322
7.2.1.652 TriggerOverlap	322

7.2.1.653	TriggerSelector	322
7.2.1.654	TriggerSoftware	322
7.2.1.655	TriggerSource	322
7.2.1.656	UserOutputSelector	322
7.2.1.657	UserOutputValue	322
7.2.1.658	UserOutputValueAll	322
7.2.1.659	UserOutputValueAllMask	322
7.2.1.660	UserSetDefault	323
7.2.1.661	UserSetFeatureEnable	323
7.2.1.662	UserSetLoad	323
7.2.1.663	UserSetSave	323
7.2.1.664	UserSetSelector	323
7.2.1.665	V3_3Enable	323
7.2.1.666	WhiteClip	323
7.2.1.667	WhiteClipSelector	323
7.2.1.668	Width	323
7.2.1.669	WidthMax	323
7.3	quickSpinTLDevice Struct Reference	323
7.3.1	Field Documentation	324
7.3.1.1	DeviceAccessStatus	324
7.3.1.2	DeviceCurrentSpeed	324
7.3.1.3	DeviceDisplayName	324
7.3.1.4	DeviceDriverVersion	324
7.3.1.5	DeviceEndianessMechanism	324
7.3.1.6	DeviceID	324
7.3.1.7	DeviceInstanceId	324
7.3.1.8	DeviceIsUpdater	324
7.3.1.9	DeviceLinkSpeed	324
7.3.1.10	DeviceModelName	324
7.3.1.11	DeviceMulticastMonitorMode	324

7.3.1.12	DeviceSerialNumber	324
7.3.1.13	DeviceType	324
7.3.1.14	DeviceU3VProtocol	324
7.3.1.15	DeviceUserID	324
7.3.1.16	DeviceVendorName	325
7.3.1.17	DeviceVersion	325
7.3.1.18	GenICamXMLLocation	325
7.3.1.19	GenICamXMLPath	325
7.3.1.20	GevCCP	325
7.3.1.21	GevDeviceDiscoverMaximumPacketSize	325
7.3.1.22	GevDeviceGateway	325
7.3.1.23	GevDeviceIPAddress	325
7.3.1.24	GevDeviceIsWrongSubnet	325
7.3.1.25	GevDeviceMACAddress	325
7.3.1.26	GevDeviceMaximumPacketSize	325
7.3.1.27	GevDeviceMaximumRetryCount	325
7.3.1.28	GevDeviceModelsBigEndian	325
7.3.1.29	GevDevicePort	325
7.3.1.30	GevDeviceReadAndWriteTimeout	325
7.3.1.31	GevDeviceSubnetMask	325
7.3.1.32	GevVersionMajor	325
7.3.1.33	GevVersionMinor	325
7.3.1.34	GUIXMLLocation	325
7.3.1.35	GUIXMLPath	325
7.4	quickSpinTLInterface Struct Reference	326
7.4.1	Field Documentation	326
7.4.1.1	ActionCommand	326
7.4.1.2	AutoForceIP	326
7.4.1.3	DeviceAccessStatus	326
7.4.1.4	DeviceCount	326

7.4.1.5	DeviceID	327
7.4.1.6	DeviceModelName	327
7.4.1.7	DeviceSelector	327
7.4.1.8	DeviceUnlock	327
7.4.1.9	DeviceUpdateList	327
7.4.1.10	DeviceVendorName	327
7.4.1.11	GevActionDeviceKey	327
7.4.1.12	GevActionGroupKey	327
7.4.1.13	GevActionGroupMask	327
7.4.1.14	GevActionTime	327
7.4.1.15	GevDeviceIPAddress	327
7.4.1.16	GevDeviceMACAddress	327
7.4.1.17	GevDeviceSubnetMask	327
7.4.1.18	GevInterfaceGateway	327
7.4.1.19	GevInterfaceIPAddress	327
7.4.1.20	GevInterfaceMACAddress	327
7.4.1.21	GevInterfaceSubnetMask	327
7.4.1.22	HostAdapterDriverVersion	327
7.4.1.23	HostAdapterName	327
7.4.1.24	HostAdapterVendor	327
7.4.1.25	IncompatibleDeviceCount	327
7.4.1.26	IncompatibleDeviceID	327
7.4.1.27	IncompatibleDeviceModelName	327
7.4.1.28	IncompatibleDeviceSelector	328
7.4.1.29	IncompatibleDeviceVendorName	328
7.4.1.30	IncompatibleGevDeviceIPAddress	328
7.4.1.31	IncompatibleGevDeviceMACAddress	328
7.4.1.32	IncompatibleGevDeviceSubnetMask	328
7.4.1.33	InterfaceDisplayName	328
7.4.1.34	InterfaceID	328

7.4.1.35	InterfaceType	328
7.4.1.36	POEStatus	328
7.5	quickSpinTLStream Struct Reference	328
7.5.1	Field Documentation	329
7.5.1.1	GevFailedPacketCount	329
7.5.1.2	GevMaximumNumberResendBuffers	329
7.5.1.3	GevMaximumNumberResendRequests	329
7.5.1.4	GevPacketResendMode	329
7.5.1.5	GevPacketResendTimeout	329
7.5.1.6	GevResendPacketCount	329
7.5.1.7	GevResendRequestCount	329
7.5.1.8	GevTotalPacketCount	329
7.5.1.9	StreamBlockTransferSize	329
7.5.1.10	StreamBufferCountManual	329
7.5.1.11	StreamBufferCountMax	329
7.5.1.12	StreamBufferCountMode	329
7.5.1.13	StreamBufferCountResult	329
7.5.1.14	StreamBufferHandlingMode	329
7.5.1.15	StreamBufferUnderrunCount	329
7.5.1.16	StreamCRCCheckEnable	329
7.5.1.17	StreamDefaultBufferCount	329
7.5.1.18	StreamDefaultBufferCountMax	329
7.5.1.19	StreamDefaultBufferCountMode	329
7.5.1.20	StreamFailedBufferCount	329
7.5.1.21	StreamID	329
7.5.1.22	StreamTotalBufferCount	329
7.5.1.23	StreamType	329
7.6	spinAVIOption Struct Reference	330
7.6.1	Detailed Description	330
7.6.2	Field Documentation	330

7.6.2.1	frameRate	330
7.6.2.2	reserved	330
7.7	spinBMPOption Struct Reference	330
7.7.1	Detailed Description	331
7.7.2	Field Documentation	331
7.7.2.1	indexedColor_8bit	331
7.7.2.2	reserved	331
7.8	spinChunkData Struct Reference	331
7.8.1	Detailed Description	332
7.8.2	Field Documentation	332
7.8.2.1	m_blackLevel	332
7.8.2.2	m_counterValue	332
7.8.2.3	m_cRC	332
7.8.2.4	m_encoderValue	332
7.8.2.5	m_exposureEndLineStatusAll	332
7.8.2.6	m_exposureTime	332
7.8.2.7	m_frameID	332
7.8.2.8	m_gain	332
7.8.2.9	m_height	332
7.8.2.10	m_image	332
7.8.2.11	m_inferenceConfidence	332
7.8.2.12	m_inferenceResult	332
7.8.2.13	m_linePitch	332
7.8.2.14	m_lineStatusAll	332
7.8.2.15	m_offsetX	333
7.8.2.16	m_offsetY	333
7.8.2.17	m_partSelector	333
7.8.2.18	m_pixelDynamicRangeMax	333
7.8.2.19	m_pixelDynamicRangeMin	333
7.8.2.20	m_scan3dAxisMax	333

7.8.2.21	m_scan3dAxisMin	333
7.8.2.22	m_scan3dCoordinateOffset	333
7.8.2.23	m_scan3dCoordinateReferenceValue	333
7.8.2.24	m_scan3dCoordinateScale	333
7.8.2.25	m_scan3dInvalidDataValue	333
7.8.2.26	m_scan3dTransformValue	333
7.8.2.27	m_scanLineSelector	333
7.8.2.28	m_sequencerSetActive	333
7.8.2.29	m_serialDataLength	333
7.8.2.30	m_streamChannelID	333
7.8.2.31	m_timerValue	333
7.8.2.32	m_timestamp	333
7.8.2.33	m_timestampLatchValue	333
7.8.2.34	m_transferBlockID	333
7.8.2.35	m_transferQueueCurrentBlockCount	333
7.8.2.36	m_width	333
7.9	spinH264Option Struct Reference	334
7.9.1	Detailed Description	334
7.9.2	Field Documentation	334
7.9.2.1	bitrate	334
7.9.2.2	frameRate	334
7.9.2.3	height	334
7.9.2.4	reserved	334
7.9.2.5	width	335
7.10	spinJPEGOption Struct Reference	335
7.10.1	Detailed Description	335
7.10.2	Field Documentation	335
7.10.2.1	progressive	335
7.10.2.2	quality	335
7.10.2.3	reserved	336

7.11 spinJPG2Option Struct Reference	336
7.11.1 Detailed Description	336
7.11.2 Field Documentation	336
7.11.2.1 quality	336
7.11.2.2 reserved	336
7.12 spinLibraryVersion Struct Reference	336
7.12.1 Detailed Description	337
7.12.2 Field Documentation	337
7.12.2.1 build	337
7.12.2.2 major	337
7.12.2.3 minor	337
7.12.2.4 type	337
7.13 spinMJPGOption Struct Reference	337
7.13.1 Detailed Description	338
7.13.2 Field Documentation	338
7.13.2.1 frameRate	338
7.13.2.2 quality	338
7.13.2.3 reserved	338
7.14 spinPGMOption Struct Reference	338
7.14.1 Detailed Description	338
7.14.2 Field Documentation	339
7.14.2.1 binaryFile	339
7.14.2.2 reserved	339
7.15 spinPNGOption Struct Reference	339
7.15.1 Detailed Description	339
7.15.2 Field Documentation	339
7.15.2.1 compressionLevel	339
7.15.2.2 interlaced	339
7.15.2.3 reserved	340
7.16 spinPPMOption Struct Reference	340
7.16.1 Detailed Description	340
7.16.2 Field Documentation	340
7.16.2.1 binaryFile	340
7.16.2.2 reserved	340
7.17 spinTIFFOption Struct Reference	340
7.17.1 Detailed Description	341
7.17.2 Field Documentation	341
7.17.2.1 compression	341
7.17.2.2 reserved	341

8 File Documentation	343
8.1 doc/Doxygen/spindocs/C/Licensing.dox File Reference	343
8.2 doc/Doxygen/spindocs/C/MainPage.dox File Reference	343
8.3 include/spinc/CameraDefsC.h File Reference	343
8.4 include/spinc/ChunkDataDefC.h File Reference	374
8.5 include/spinc/QuickSpinC.h File Reference	375
8.6 include/spinc/QuickSpinDefsC.h File Reference	375
8.6.1 Typedef Documentation	376
8.6.1.1 quickSpinBooleanNode	376
8.6.1.2 quickSpinCommandNode	376
8.6.1.3 quickSpinEnumerationNode	376
8.6.1.4 quickSpinFloatNode	376
8.6.1.5 quickSpinIntegerNode	376
8.6.1.6 quickSpinRegisterNode	376
8.6.1.7 quickSpinStringNode	376
8.7 include/spinc/SpinnakerC.h File Reference	376
8.8 include/spinc/SpinnakerDefsC.h File Reference	385
8.9 include/spinc/SpinnakerGenApiC.h File Reference	391
8.10 include/spinc/SpinnakerGenApiDefsC.h File Reference	395
8.11 include/spinc/SpinnakerPlatformC.h File Reference	398
8.11.1 Macro Definition Documentation	398
8.11.1.1 SPINNAKERC_API	398
8.12 include/spinc/SpinVideoC.h File Reference	399
8.13 include/spinc/TransportLayerDefsC.h File Reference	399
8.14 include/spinc/TransportLayerDeviceC.h File Reference	401
8.15 include/spinc/TransportLayerInterfaceC.h File Reference	402
8.16 include/spinc/TransportLayerStreamC.h File Reference	403
Index	405

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	112
QuickSpin Access	113
Transport Layer Enumerations	272
TLDevice Structures	278
TLInterface Structures	279
TLStream Structures	280
Spinnaker C API	114
Spinnaker C Definitions	11
Camera Enumerations	13
Chunk Data Structures	111
Spinnaker C Handles	201
Spinnaker C Function Signatures	205
Spinnaker C Enumerations	206
Spinnaker C Structures	214
Error Handling	116
System Access	120
InterfaceList Access	130
CameraList Access	133
Interface Access	138
Camera Access	145
SpinVideo Recording Access	270
Image Access	155
Event Access	178
ImageStatistics Access	184
Logging Event Data Access	191
Device Event Data Access	195
AVIRecorder Access	198
Chunk data access	200
Spinnaker C GenICam API	216
Node Map Access	218
Node Access	221
IValue Access	231
String Access	234
Integer Access	237

IFloat Access	241
IEnumeration Access	245
IEnumEntry Access	249
IBoolean Access	251
ICommand Access	253
ICategory Access	255
IRegister Access	257
Spinnaker C GenICam Handles	261
Spinnaker C GenICam Enumerations	262

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

actionCommandResult	
Action Command Result	281
quickSpin	282
quickSpinTLDevice	323
quickSpinTLInterface	326
quickSpinTLStream	328
spinAVIOption	
Options for saving uncompressed videos	330
spinBMPOption	
Options for saving BMP images	330
spinChunkData	
The type of information that can be obtained from image chunk data	331
spinH264Option	
Options for saving H264 videos	334
spinJPEGOption	
Options for saving JPEG images	335
spinJPG2Option	
Options for saving JPEG 2000 images	336
spinLibraryVersion	
Provides easier access to the current version of Spinnaker	336
spinMJPGOption	
Options for saving MJPG videos	337
spinPGMOption	
Options for saving PGM images	338
spinPNGOption	
Options for saving PNG images	339
spinPPMOption	
Options for saving PPM images	340
spinTIFFOption	
Options for saving TIFF images	340

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

include/spinc/ CameraDefsC.h	343
include/spinc/ ChunkDataDefC.h	374
include/spinc/ QuickSpinC.h	375
include/spinc/ QuickSpinDefsC.h	375
include/spinc/ SpinnakerC.h	376
include/spinc/ SpinnakerDefsC.h	385
include/spinc/ SpinnakerGenApiC.h	391
include/spinc/ SpinnakerGenApiDefsC.h	395
include/spinc/ SpinnakerPlatformC.h	398
include/spinc/ SpinVideoC.h	399
include/spinc/ TransportLayerDefsC.h	399
include/spinc/ TransportLayerDeviceC.h	401
include/spinc/ TransportLayerInterfaceC.h	402
include/spinc/ TransportLayerStreamC.h	403

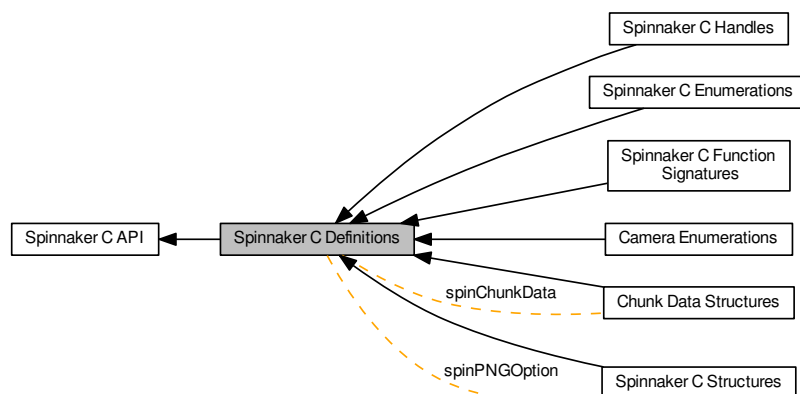
Chapter 6

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 typedef uint8_t bool8_t

6.1.3 Variable Documentation

6.1.3.1 const bool8_t False = 0 [static]

6.1.3.2 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_DEVICECTLTYPE }
- enum spinDevicePowerSupplySelectorEnums {
DevicePowerSupplySelector_External,
NUM_DEVICEPOWERSUPPLYSELECTOR }
- enum spinDeviceTemperatureSelectorEnums {
DeviceTemperatureSelector_Sensor,
NUM_DEVICETEMPERATURESELECTOR }
- enum spinDeviceIndicatorModeEnums {
DeviceIndicatorMode_Inactive,
DeviceIndicatorMode_Active,
DeviceIndicatorMode_ErrorStatus,
NUM_DEVICEINDICATORMODE }
- enum spinAutoExposureControlPriorityEnums {
AutoExposureControlPriority_Gain,
AutoExposureControlPriority_ExposureTime,
NUM_AUTOEXPOSURECONTROLPRIORITY }
- enum spinAutoExposureMeteringModeEnums {
AutoExposureMeteringMode_Average,
AutoExposureMeteringMode_Spot,
AutoExposureMeteringMode_Partial,
AutoExposureMeteringMode_CenterWeighted,
AutoExposureMeteringMode_HistogramPeak,
NUM_AUTOEXPOSUREMETERINGMODE }
- enum spinBalanceWhiteAutoProfileEnums {
BalanceWhiteAutoProfile_Indoor,
BalanceWhiteAutoProfile_Outdoor,
NUM_BALANCEWHITEAUTOPROFILE }
- enum spinAutoAlgorithmSelectorEnums {
AutoAlgorithmSelector_Awb,
AutoAlgorithmSelector_Ae,
NUM_AUTOALGORITHMSELECTOR }
- enum spinAutoExposureTargetGreyValueAutoEnums {
AutoExposureTargetGreyValueAuto_Off,
AutoExposureTargetGreyValueAuto_Continuous,
NUM_AUTOEXPOSURETARGETGREYVALUEAUTO }
- enum spinAutoExposureLightingModeEnums {
AutoExposureLightingMode_AutoDetect,
AutoExposureLightingMode_Backlight,
AutoExposureLightingMode_Frontlight,
AutoExposureLightingMode_Normal,
NUM_AUTOEXPOSURELIGHTINGMODE }
- enum spinGevIEEE1588StatusEnums {


```

    GevIEEE1588Status_Initializing,
    GevIEEE1588Status_Faulty,
    GevIEEE1588Status_Disabled,
    GevIEEE1588Status_Listening,
    GevIEEE1588Status_PreMaster,
    GevIEEE1588Status_Master,
    GevIEEE1588Status_Passive,
    GevIEEE1588Status_Uncalibrated,
    GevIEEE1588Status_Slave,
    NUM_GEVIEEE1588STATUS }

• enum spinGevIEEE1588ModeEnums {
    GevIEEE1588Mode_Auto,
    GevIEEE1588Mode_SlaveOnly,
    NUM_GEVIEEE1588MODE }

• enum spinGevIEEE1588ClockAccuracyEnums {
    GevIEEE1588ClockAccuracy_Unknown,
    NUM_GEVIEEE1588CLOCKACCURACY }

• enum spinGevCCPEnums {
    GevCCP_OpenAccess,
    GevCCP_ExclusiveAccess,
    GevCCP_ControlAccess,
    NUM_GEVCCP }

• enum spinGevSupportedOptionSelectorEnums {
    GevSupportedOptionSelector_UserDefinedName,
    GevSupportedOptionSelector_SerialNumber,
    GevSupportedOptionSelector_HeartbeatDisable,
    GevSupportedOptionSelector_LinkSpeed,
    GevSupportedOptionSelector_CCPApplicationSocket,
    GevSupportedOptionSelector_ManifestTable,
    GevSupportedOptionSelector_TestData,
    GevSupportedOptionSelector_DiscoveryAckDelay,
    GevSupportedOptionSelector_DiscoveryAckDelayWritable,
    GevSupportedOptionSelector_ExtendedStatusCodes,
    GevSupportedOptionSelector_Action,
    GevSupportedOptionSelector_PendingAck,
    GevSupportedOptionSelector_EventData,
    GevSupportedOptionSelector_Event,
    GevSupportedOptionSelector_PacketResend,
    GevSupportedOptionSelector_WriteMem,
    GevSupportedOptionSelector_CommandsConcatenation,
    GevSupportedOptionSelector_IPConfigurationLLA,
    GevSupportedOptionSelector_IPConfigurationDHCP,
    GevSupportedOptionSelector_IPConfigurationPersistentIP,
    GevSupportedOptionSelector_StreamChannelSourceSocket,
    GevSupportedOptionSelector_MessageChannelSourceSocket,
    NUM_GEVSUPPORTEDOPTIONSELECTOR }

• enum spinBlackLevelSelectorEnums {
    BlackLevelSelector_All,
    BlackLevelSelector_Analog,
    BlackLevelSelector_Digital,
    NUM_BLACKLEVELSELECTOR }

• enum spinBalanceWhiteAutoEnums {
    BalanceWhiteAuto_Off,
    BalanceWhiteAuto_Once,
    BalanceWhiteAuto_Continuous,
    NUM_BALANCEWHITEAUTO }

• enum spinGainAutoEnums {

```

```
GainAuto_Off,  
GainAuto_Once,  
GainAuto_Continuous,  
NUM_GAINAUTO }  
  
• enum spinBalanceRatioSelectorEnums {  
    BalanceRatioSelector_Red,  
    BalanceRatioSelector_Blue,  
    NUM_BALANCERATIOSELECTOR }  
  
• enum spinGainSelectorEnums {  
    GainSelector_All,  
    NUM_GAINSELECTOR }  
  
• enum spinDefectCorrectionModeEnums {  
    DefectCorrectionMode_Average,  
    DefectCorrectionMode_Highlight,  
    DefectCorrectionMode_Zero,  
    NUM_DEFECTCORRECTIONMODE }  
  
• enum spinUserSetSelectorEnums {  
    UserSetSelector_Default,  
    UserSetSelector_UserSet0,  
    UserSetSelector_UserSet1,  
    NUM_USERSETSELECTOR }  
  
• enum spinUserSetDefaultEnums {  
    UserSetDefault_Default,  
    UserSetDefault_UserSet0,  
    UserSetDefault_UserSet1,  
    NUM_USERSETDEFAULT }  
  
• enum spinSerialPortBaudRateEnums {  
    SerialPortBaudRate_Baud300,  
    SerialPortBaudRate_Baud600,  
    SerialPortBaudRate_Baud1200,  
    SerialPortBaudRate_Baud2400,  
    SerialPortBaudRate_Baud4800,  
    SerialPortBaudRate_Baud9600,  
    SerialPortBaudRate_Baud14400,  
    SerialPortBaudRate_Baud19200,  
    SerialPortBaudRate_Baud38400,  
    SerialPortBaudRate_Baud57600,  
    SerialPortBaudRate_Baud115200,  
    SerialPortBaudRate_Baud230400,  
    SerialPortBaudRate_Baud460800,  
    SerialPortBaudRate_Baud921600,  
    NUM_SERIALPORTBAUDRATE }  
  
• enum spinSerialPortParityEnums {  
    SerialPortParity_None,  
    SerialPortParity_Odd,  
    SerialPortParity_Even,  
    SerialPortParity_Mark,  
    SerialPortParity_Space,  
    NUM_SERIALPORTPARITY }  
  
• enum spinSerialPortSelectorEnums {  
    SerialPortSelector_SerialPort0,  
    NUM_SERIALPORTSELECTOR }  
  
• enum spinSerialPortStopBitsEnums {  
    SerialPortStopBits_Bits1,  
    SerialPortStopBits_Bits1AndAHalf,  
    SerialPortStopBits_Bits2,  
    NUM_SERIALPORTSTOPBITS }
```

- enum `spinSerialPortSourceEnums` {
 `SerialPortSource_Line0`,
 `SerialPortSource_Line1`,
 `SerialPortSource_Line2`,
 `SerialPortSource_Line3`,
 `SerialPortSource_Off`,
 `NUM_SERIALPORTSOURCE` }
- enum `spinSequencerModeEnums` {
 `SequencerMode_Off`,
 `SequencerMode_On`,
 `NUM_SEQUENCERMODE` }
- enum `spinSequencerConfigurationValidEnums` {
 `SequencerConfigurationValid_No`,
 `SequencerConfigurationValid_Yes`,
 `NUM_SEQUENCERCONFIGURATIONVALID` }
- enum `spinSequencerSetValidEnums` {
 `SequencerSetValid_No`,
 `SequencerSetValid_Yes`,
 `NUM_SEQUENCERSETVALID` }
- enum `spinSequencerTriggerActivationEnums` {
 `SequencerTriggerActivation_RisingEdge`,
 `SequencerTriggerActivation_FallingEdge`,
 `SequencerTriggerActivation_AnyEdge`,
 `SequencerTriggerActivation_LevelHigh`,
 `SequencerTriggerActivation_LevelLow`,
 `NUM_SEQUENCERTRIGGERACTIVATION` }
- enum `spinSequencerConfigurationModeEnums` {
 `SequencerConfigurationMode_Off`,
 `SequencerConfigurationMode_On`,
 `NUM_SEQUENCERCONFIGURATIONMODE` }
- enum `spinSequencerTriggerSourceEnums` {
 `SequencerTriggerSource_Off`,
 `SequencerTriggerSource_FrameStart`,
 `NUM_SEQUENCERTRIGGERSOURCE` }
- enum `spinTransferQueueModeEnums` {
 `TransferQueueMode_FirstInFirstOut`,
 `NUM_TRANSFERQUEUEMODE` }
- enum `spinTransferOperationModeEnums` {
 `TransferOperationMode_Continuous`,
 `TransferOperationMode_MultiBlock`,
 `NUM_TRANSFEROPERATIONMODE` }
- enum `spinTransferControlModeEnums` {
 `TransferControlMode_Basic`,
 `TransferControlMode_Automatic`,
 `TransferControlMode_UserControlled`,
 `NUM_TRANSFERCONTROLMODE` }
- enum `spinChunkGainSelectorEnums` {
 `ChunkGainSelector_All`,
 `ChunkGainSelector_Red`,
 `ChunkGainSelector_Green`,
 `ChunkGainSelector_Blue`,
 `NUM_CHUNKGAINSELECTOR` }
- enum `spinChunkSelectorEnums` {

```

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

• enum spinChunkBlackLevelSelectorEnums {
    ChunkBlackLevelSelector_All,
    NUM_CHUNKBLACKLEVELSELECTOR }

• enum spinChunkPixelFormatEnums {
    ChunkPixelFormat_Mono8,
    ChunkPixelFormat_Mono12Packed,
    ChunkPixelFormat_Mono16,
    ChunkPixelFormat_RGB8Packed,
    ChunkPixelFormat_YUV422Packed,
    ChunkPixelFormat_BayerGR8,
    ChunkPixelFormat_BayerRG8,
    ChunkPixelFormat_BayerGB8,
    ChunkPixelFormat_BayerBG8,
    ChunkPixelFormat_YCbCr601_422_8_CbYCrY,
    NUM_CHUNKPIXELFORMAT }

• enum spinFileOperationStatusEnums {
    FileOperationStatus_Success,
    FileOperationStatus_Failure,
    FileOperationStatus_Overflow,
    NUM_FILEOPERATIONSTATUS }

• enum spinFileOpenModeEnums {
    FileOpenMode_Read,
    FileOpenMode_Write,
    FileOpenMode_ReadWrite,
    NUM_FILEOPENMODE }

• enum spinFileOperationSelectorEnums {
    FileOperationSelector_Open,
    FileOperationSelector_Close,
    FileOperationSelector_Read,
    FileOperationSelector_Write,
    FileOperationSelector_Delete,
    NUM_FILEOPERATIONSELECTOR }

• enum spinFileSelectorEnums {
    FileSelector_UserSetDefault,
    FileSelector_UserSet0,
    FileSelector_UserSet1,
    FileSelector_UserFile1,
    FileSelector_SerialPort0,
    NUM_FILESELECTOR }

• enum spinBinningSelectorEnums {

```

```

    BinningSelector_All,
    BinningSelector_Sensor,
    BinningSelector_ISP,
    NUM_BINNINGSELECTOR }
• enum spinTestPatternGeneratorSelectorEnums {
    TestPatternGeneratorSelector_Sensor,
    TestPatternGeneratorSelector_PipelineStart,
    NUM_TESTPATTERNGENERATORSELECTOR }
• enum 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_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,

```

    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_DEVICESTRIALPORTSELECTOR` }
- enum `spinDeviceSerialPortBaudRateEnums` {
 `DeviceSerialPortBaudRate_Baud9600`,
 `DeviceSerialPortBaudRate_Baud19200`,
 `DeviceSerialPortBaudRate_Baud38400`,
 `DeviceSerialPortBaudRate_Baud57600`,
 `DeviceSerialPortBaudRate_Baud115200`,
 `DeviceSerialPortBaudRate_Baud230400`,
 `DeviceSerialPortBaudRate_Baud460800`,
 `DeviceSerialPortBaudRate_Baud921600`,
 `NUM_DEVICESTRIALPORTBAUDRATE` }
- 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` {

Generated by Doxygen

- ```
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_GEVPHYSCALLINKCONFIGURATION }

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

```

- 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 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 enum spinAcquisitionStatusSelectorEnums

< Selects the internal acquisition signal to read using AcquisitionStatus.

Enumerator

**AcquisitionStatusSelector\_AcquisitionTriggerWait** Device is currently waiting for a trigger for the capture of one or many frames.

**AcquisitionStatusSelector\_AcquisitionActive** Device is currently doing an acquisition of one or many frames.

**AcquisitionStatusSelector\_AcquisitionTransfer** Device is currently transferring an acquisition of one or many frames.

**AcquisitionStatusSelector\_FrameTriggerWait** Device is currently waiting for a frame start trigger.

**AcquisitionStatusSelector\_FrameActive** Device is currently doing the capture of a frame.

**AcquisitionStatusSelector\_ExposureActive** Device is doing the exposure of a frame.

**NUM\_ACQUISITIONSTATUSSELECTOR**

### 6.2.2.3 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 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 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 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 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 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 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 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 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 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 enum spinBinningHorizontalModeEnums

&lt;

## 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 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 enum spinBinningVerticalModeEnums

&lt;

## Enumerator

**BinningVerticalMode\_Sum** The response from the combined vertical cells is added, resulting in increased sensitivity (a brighter image).

**BinningVerticalMode\_Average** The response from the combined vertical cells is averaged, resulting in increased signal/noise ratio. Not all sensors support average binning.

**NUM\_BINNINGVERTICALMODE**

## 6.2.2.16 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 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 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 enum spinChunkBlackLevelSelectorEnums

< Selects which black level to retrieve

#### Enumerator

***ChunkBlackLevelSelector\_All***

***NUM\_CHUNKBLACKLEVELSELECTOR***

### 6.2.2.20 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 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 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 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 enum spinChunkGainSelectorEnums

< Selects which gain to retrieve

Enumerator

***ChunkGainSelector\_All***  
***ChunkGainSelector\_Red***  
***ChunkGainSelector\_Green***  
***ChunkGainSelector\_Blue***  
***NUM\_CHUNKGAINSELECTOR***

## 6.2.2.25 enum spinChunkImageComponentEnums

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

Enumerator

***ChunkImageComponent\_Intensity*** The image data is the intensity component.  
***ChunkImageComponent\_Color*** The image data is color component.  
***ChunkImageComponent\_Infrared*** The image data is infrared component.  
***ChunkImageComponent\_Ultraviolet*** The image data is the ultraviolet component.  
***ChunkImageComponent\_Range*** The image data is the range (distance) component.  
***ChunkImageComponent\_Disparity*** The image data is the disparity component.  
***ChunkImageComponent\_Confidence*** The image data is the confidence map component.  
***ChunkImageComponent\_Scatter*** The image data is the scatter component.  
***NUM\_CHUNKIMAGECOMPONENT***

## 6.2.2.26 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 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 enum spinChunkScan3dCoordinateReferenceSelectorEnums

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

Enumerator

**ChunkScan3dCoordinateReferenceSelector\_RotationX** Rotation around X axis.

**ChunkScan3dCoordinateReferenceSelector\_RotationY** Rotation around Y axis.

**ChunkScan3dCoordinateReferenceSelector\_RotationZ** Rotation around Z axis.

**ChunkScan3dCoordinateReferenceSelector\_TranslationX** X axis translation.

**ChunkScan3dCoordinateReferenceSelector\_TranslationY** Y axis translation.

**ChunkScan3dCoordinateReferenceSelector\_TranslationZ** Z axis translation.

**NUM\_CHUNKSCAN3DCOORDINATEREFERENCESELECTOR**

## 6.2.2.29 enum spinChunkScan3dCoordinateSelectorEnums

< Selects which Coordinate to retrieve data from.

Enumerator

**ChunkScan3dCoordinateSelector\_CoordinateA** The first (X or Theta) coordinate

**ChunkScan3dCoordinateSelector\_CoordinateB** The second (Y or Phi) coordinate

**ChunkScan3dCoordinateSelector\_CoordinateC** The third (Z or Rho) coordinate.

**NUM\_CHUNKSCAN3DCOORDINATESELECTOR**

## 6.2.2.30 enum spinChunkScan3dCoordinateSystemEnums

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

Enumerator

**ChunkScan3dCoordinateSystem\_Cartesian** Default value. 3-axis orthogonal, right-hand X-Y-Z.

**ChunkScan3dCoordinateSystem\_Spherical** A Theta-Phi-Rho coordinate system.

**ChunkScan3dCoordinateSystem\_Cylindrical** A Theta-Y-Rho coordinate system.

**NUM\_CHUNKSCAN3DCOORDINATESYSTEM**

### 6.2.2.31 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\_CHUNKSCAN3DCOORDINATESYSTEMREFERENCE***

### 6.2.2.32 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 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 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\_PointCloud** 3 Coordinates without organization. The full 3 coordinate data without any organization of data points. Typically only valid points transmitted giving varying image size.

**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 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 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 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 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 enum spinCIConfigurationEnums

< 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

**CIConfiguration\_Base** Standard base configuration described by the Camera Link standard.

**CIConfiguration\_Medium** Standard medium configuration described by the Camera Link standard.

**CIConfiguration\_Full** Standard full configuration described by the Camera Link standard.

**CIConfiguration\_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.

**CIConfiguration\_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 enum spinCITimeSlotsCountEnums

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

## Enumerator

**CITimeSlotsCount\_One** One

**CITimeSlotsCount\_Two** Two

**CITimeSlotsCount\_Three** Three

**NUM\_CLTIMESLOTSCOUNT**

## 6.2.2.41 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 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 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 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 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 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 enum spinCounterSelectorEnums

< Selects which counter to configure

Enumerator

***CounterSelector\_Counter0***

***CounterSelector\_Counter1***

***NUM\_COUNTERSELECTOR***

#### 6.2.2.48 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 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 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 enum spinCxpConnectionTestModeEnums

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

## Enumerator

**CxpConnectionTestMode\_Off** Off  
**CxpConnectionTestMode\_Mode1** Mode 1  
**NUM\_CXP\_CONNECTIONTESTMODE**

## 6.2.2.52 enum spinCxpLinkConfigurationEnums

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

## Enumerator

**CxpLinkConfiguration\_Auto** Sets Automatic discovery for the Link Configuration.  
**CxpLinkConfiguration\_CXP1\_X1** Force the Link to 1 Connection operating at CXP-1 speed (1.25 Gbps).

|                                            |                                                                        |
|--------------------------------------------|------------------------------------------------------------------------|
| <b><i>CxpLinkConfiguration_CXP2_X1</i></b> | Force the Link to 1 Connection operating at CXP-2 speed (2.50 Gbps).   |
| <b><i>CxpLinkConfiguration_CXP3_X1</i></b> | Force the Link to 1 Connection operating at CXP-3 speed (3.125 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP5_X1</i></b> | Force the Link to 1 Connection operating at CXP-5 speed (5.00 Gbps).   |
| <b><i>CxpLinkConfiguration_CXP6_X1</i></b> | Force the Link to 1 Connection operating at CXP-6 speed (6.25 Gbps).   |
| <b><i>CxpLinkConfiguration_CXP1_X2</i></b> | Force the Link to 2 Connections operating at CXP-1 speed (1.25 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP2_X2</i></b> | Force the Link to 2 Connections operating at CXP-2 speed (2.50 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP3_X2</i></b> | Force the Link to 2 Connections operating at CXP-3 speed (3.125 Gbps). |
| <b><i>CxpLinkConfiguration_CXP5_X2</i></b> | Force the Link to 2 Connections operating at CXP-5 speed (5.00 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP6_X2</i></b> | Force the Link to 3 Connections operating at CXP-6 speed (6.25 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP1_X3</i></b> | Force the Link to 3 Connections operating at CXP-1 speed (1.25 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP2_X3</i></b> | Force the Link to 3 Connections operating at CXP-2 speed (2.50 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP3_X3</i></b> | Force the Link to 3 Connections operating at CXP-3 speed (3.125 Gbps). |
| <b><i>CxpLinkConfiguration_CXP5_X3</i></b> | Force the Link to 3 Connections operating at CXP-5 speed (5.00 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP6_X3</i></b> | Force the Link to 3 Connections operating at CXP-6 speed (6.25 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP1_X4</i></b> | Force the Link to 4 Connections operating at CXP-1 speed (1.25 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP2_X4</i></b> | Force the Link to 4 Connections operating at CXP-2 speed (2.50 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP3_X4</i></b> | Force the Link to 4 Connections operating at CXP-3 speed (3.125 Gbps). |
| <b><i>CxpLinkConfiguration_CXP5_X4</i></b> | Force the Link to 4 Connections operating at CXP-5 speed (5.00 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP6_X4</i></b> | Force the Link to 4 Connections operating at CXP-6 speed (6.25 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP1_X5</i></b> | Force the Link to 5 Connections operating at CXP-1 speed (1.25 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP2_X5</i></b> | Force the Link to 5 Connections operating at CXP-2 speed (2.50 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP3_X5</i></b> | Force the Link to 5 Connections operating at CXP-3 speed (3.125 Gbps). |
| <b><i>CxpLinkConfiguration_CXP5_X5</i></b> | Force the Link to 5 Connections operating at CXP-5 speed (5.00 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP6_X5</i></b> | Force the Link to 5 Connections operating at CXP-6 speed (6.25 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP1_X6</i></b> | Force the Link to 6 Connections operating at CXP-1 speed (1.25 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP2_X6</i></b> | Force the Link to 6 Connections operating at CXP-2 speed (2.50 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP3_X6</i></b> | Force the Link to 6 Connections operating at CXP-3 speed (3.125 Gbps). |
| <b><i>CxpLinkConfiguration_CXP5_X6</i></b> | Force the Link to 6 Connections operating at CXP-5 speed (5.00 Gbps).  |
| <b><i>CxpLinkConfiguration_CXP6_X6</i></b> | Force the Link to 6 Connections operating at CXP-6 speed (6.25 Gbps).  |
| <b><i>NUM_CXPLINKCONFIGURATION</i></b>     |                                                                        |

### 6.2.2.53 enum spinCxpLinkConfigurationPreferredEnums

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

Enumerator

|                                                     |                                                     |
|-----------------------------------------------------|-----------------------------------------------------|
| <b><i>CxpLinkConfigurationPreferred_CXP1_X1</i></b> | 1 Connection operating at CXP-1 speed (1.25 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP2_X1</i></b> | 1 Connection operating at CXP-2 speed (2.50 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP3_X1</i></b> | 1 Connection operating at CXP-3 speed (3.125 Gbps). |

|                                                     |                                                      |
|-----------------------------------------------------|------------------------------------------------------|
| <b><i>CxpLinkConfigurationPreferred_CXP5_X1</i></b> | 1 Connection operating at CXP-5 speed (5.00 Gbps).   |
| <b><i>CxpLinkConfigurationPreferred_CXP6_X1</i></b> | 1 Connection operating at CXP-6 speed (6.25 Gbps).   |
| <b><i>CxpLinkConfigurationPreferred_CXP1_X2</i></b> | 2 Connections operating at CXP-1 speed (1.25 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP2_X2</i></b> | 2 Connections operating at CXP-2 speed (2.50 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP3_X2</i></b> | 2 Connections operating at CXP-3 speed (3.125 Gbps). |
| <b><i>CxpLinkConfigurationPreferred_CXP5_X2</i></b> | 2 Connections operating at CXP-4 speed (5.00 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP6_X2</i></b> | 3 Connections operating at CXP-5 speed (6.25 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP1_X3</i></b> | 3 Connections operating at CXP-1 speed (1.25 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP2_X3</i></b> | 3 Connections operating at CXP-2 speed (2.50 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP3_X3</i></b> | 3 Connections operating at CXP-3 speed (3.125 Gbps). |
| <b><i>CxpLinkConfigurationPreferred_CXP5_X3</i></b> | 3 Connections operating at CXP-5 speed (5.00 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP6_X3</i></b> | 3 Connections operating at CXP-6 speed (6.25 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP1_X4</i></b> | 4 Connections operating at CXP-1 speed (1.25 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP2_X4</i></b> | 4 Connections operating at CXP-2 speed (2.50 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP3_X4</i></b> | 4 Connections operating at CXP-3 speed (3.125 Gbps). |
| <b><i>CxpLinkConfigurationPreferred_CXP5_X4</i></b> | 4 Connections operating at CXP-5 speed (5.00 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP6_X4</i></b> | 4 Connections operating at CXP-6 speed (6.25 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP1_X5</i></b> | 5 Connections operating at CXP-1 speed (1.25 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP2_X5</i></b> | 5 Connections operating at CXP-2 speed (2.50 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP3_X5</i></b> | 5 Connections operating at CXP-3 speed (3.125 Gbps). |
| <b><i>CxpLinkConfigurationPreferred_CXP5_X5</i></b> | 5 Connections operating at CXP-5 speed (5.00 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP6_X5</i></b> | 5 Connections operating at CXP-6 speed (6.25 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP1_X6</i></b> | 6 Connections operating at CXP-1 speed (1.25 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP2_X6</i></b> | 6 Connections operating at CXP-2 speed (2.50 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP3_X6</i></b> | 6 Connections operating at CXP-3 speed (3.125 Gbps). |
| <b><i>CxpLinkConfigurationPreferred_CXP5_X6</i></b> | 6 Connections operating at CXP-5 speed (5.00 Gbps).  |
| <b><i>CxpLinkConfigurationPreferred_CXP6_X6</i></b> | 6 Connections operating at CXP-6 speed (6.25 Gbps).  |
| <b><i>NUM_CXPLINKCONFIGURATIONPREFERRED</i></b>     |                                                      |

#### 6.2.2.54 enum spinCxpLinkConfigurationStatusEnums

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

##### Enumerator

|                                                  |                                                                                                                               |
|--------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| <b><i>CxpLinkConfigurationStatus_None</i></b>    | The Link configuration of the Device is unknown. Either the configuration operation has failed or there is nothing connected. |
| <b><i>CxpLinkConfigurationStatus_Pending</i></b> | The Device is in the process of configuring the Link. The Link cannot be used yet.                                            |
| <b><i>CxpLinkConfigurationStatus_CXP1_X1</i></b> | 1 Connection operating at CXP-1 speed (1.25 Gbps).                                                                            |
| <b><i>CxpLinkConfigurationStatus_CXP2_X1</i></b> | 1 Connection operating at CXP-2 speed (2.50 Gbps).                                                                            |
| <b><i>CxpLinkConfigurationStatus_CXP3_X1</i></b> | 1 Connection operating at CXP-3 speed (3.125 Gbps).                                                                           |
| <b><i>CxpLinkConfigurationStatus_CXP5_X1</i></b> | 1 Connection operating at CXP-5 speed (5.00 Gbps).                                                                            |
| <b><i>CxpLinkConfigurationStatus_CXP6_X1</i></b> | 1 Connection operating at CXP-6 speed (6.25 Gbps).                                                                            |
| <b><i>CxpLinkConfigurationStatus_CXP1_X2</i></b> | 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 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 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 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 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 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 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 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 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 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 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 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 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 enum spinDevicePowerSupplySelectorEnums**

< Selects the power supply source to control or read.

Enumerator

***DevicePowerSupplySelector\_External***

***NUM\_DEVICEPOWERSUPPLYSELECTOR***

**6.2.2.68 enum spinDeviceRegistersEndiannessEnums**

< Endianness of the registers of the device.

Enumerator

***DeviceRegistersEndianness\_Little***

***DeviceRegistersEndianness\_Big***

***NUM\_DEVICEREGISTERSENDIANNES***

**6.2.2.69 enum spinDeviceScanTypeEnums**

< Scan type of the sensor of the device.

Enumerator

***DeviceScanType\_Areascan***

***NUM\_DEVICESCANTYPE***

#### 6.2.2.70 enum spinDeviceSerialPortBaudRateEnums

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

Enumerator

***DeviceSerialPortBaudRate\_Baud9600*** Serial port speed of 9600 baud.  
***DeviceSerialPortBaudRate\_Baud19200*** Serial port speed of 19200 baud.  
***DeviceSerialPortBaudRate\_Baud38400*** Serial port speed of 38400 baud.  
***DeviceSerialPortBaudRate\_Baud57600*** Serial port speed of 57600 baud.  
***DeviceSerialPortBaudRate\_Baud115200*** Serial port speed of 115200 baud.  
***DeviceSerialPortBaudRate\_Baud230400*** Serial port speed of 230400 baud.  
***DeviceSerialPortBaudRate\_Baud460800*** Serial port speed of 460800 baud.  
***DeviceSerialPortBaudRate\_Baud921600*** Serial port speed of 921600 baud.  
***NUM\_DEVICESERIALPORTBAUDRATE***

#### 6.2.2.71 enum spinDeviceSerialPortSelectorEnums

< Selects which serial port of the device to control.

Enumerator

***DeviceSerialPortSelector\_CameraLink*** Serial port associated to the Camera link connection.  
***NUM\_DEVICESERIALPORTSELECTOR***

#### 6.2.2.72 enum spinDeviceStreamChannelEndiannessEnums

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

Enumerator

***DeviceStreamChannelEndianness\_Big*** Stream channel data is big Endian.  
***DeviceStreamChannelEndianness\_Little*** Stream channel data is little Endian.  
***NUM\_DEVICESTREAMCHANNELENDIANNESS***

#### 6.2.2.73 enum spinDeviceStreamChannelTypeEnums

< Reports the type of the stream channel.

Enumerator

***DeviceStreamChannelType\_Transmitter*** Data stream transmitter channel.  
***DeviceStreamChannelType\_Receiver*** Data stream receiver channel.  
***NUM\_DEVICESTREAMCHANNELTYPE***

## 6.2.2.74 enum spinDeviceTapGeometryEnums

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

## Enumerator

**DeviceTapGeometry\_Geometry\_1X\_1Y** Geometry\_1X\_1Y  
**DeviceTapGeometry\_Geometry\_1X2\_1Y** Geometry\_1X2\_1Y  
**DeviceTapGeometry\_Geometry\_1X2\_1Y2** Geometry\_1X2\_1Y2  
**DeviceTapGeometry\_Geometry\_2X\_1Y** Geometry\_2X\_1Y  
**DeviceTapGeometry\_Geometry\_2X\_1Y2Geometry\_2XE\_1Y** Geometry\_2X\_1Y2Geometry\_2XE\_1Y  
**DeviceTapGeometry\_Geometry\_2XE\_1Y2** Geometry\_2XE\_1Y2  
**DeviceTapGeometry\_Geometry\_2XM\_1Y** Geometry\_2XM\_1Y  
**DeviceTapGeometry\_Geometry\_2XM\_1Y2** Geometry\_2XM\_1Y2  
**DeviceTapGeometry\_Geometry\_1X\_1Y2** Geometry\_1X\_1Y2  
**DeviceTapGeometry\_Geometry\_1X\_2YE** Geometry\_1X\_2YE  
**DeviceTapGeometry\_Geometry\_1X3\_1Y** Geometry\_1X3\_1Y  
**DeviceTapGeometry\_Geometry\_3X\_1Y** Geometry\_3X\_1Y  
**DeviceTapGeometry\_Geometry\_1X** Geometry\_1X  
**DeviceTapGeometry\_Geometry\_1X2** Geometry\_1X2  
**DeviceTapGeometry\_Geometry\_2X** Geometry\_2X  
**DeviceTapGeometry\_Geometry\_2XE** Geometry\_2XE  
**DeviceTapGeometry\_Geometry\_2XM** Geometry\_2XM  
**DeviceTapGeometry\_Geometry\_1X3** Geometry\_1X3  
**DeviceTapGeometry\_Geometry\_3X** Geometry\_3X  
**DeviceTapGeometry\_Geometry\_1X4\_1Y** Geometry\_1X4\_1Y  
**DeviceTapGeometry\_Geometry\_4X\_1Y** Geometry\_4X\_1Y  
**DeviceTapGeometry\_Geometry\_2X2\_1Y** Geometry\_2X2\_1Y  
**DeviceTapGeometry\_Geometry\_2X2E\_1YGeometry\_2X2M\_1Y** Geometry\_2X2E\_1YGeometry\_2X2M↵  
 \_1Y  
**DeviceTapGeometry\_Geometry\_1X2\_2YE** Geometry\_1X2\_2YE  
**DeviceTapGeometry\_Geometry\_2X\_2YE** Geometry\_2X\_2YE  
**DeviceTapGeometry\_Geometry\_2XE\_2YE** Geometry\_2XE\_2YE  
**DeviceTapGeometry\_Geometry\_2XM\_2YE** Geometry\_2XM\_2YE  
**DeviceTapGeometry\_Geometry\_1X4** Geometry\_1X4  
**DeviceTapGeometry\_Geometry\_4X** Geometry\_4X  
**DeviceTapGeometry\_Geometry\_2X2** Geometry\_2X2  
**DeviceTapGeometry\_Geometry\_2X2E** Geometry\_2X2E  
**DeviceTapGeometry\_Geometry\_2X2M** Geometry\_2X2M  
**DeviceTapGeometry\_Geometry\_1X8\_1Y** Geometry\_1X8\_1Y  
**DeviceTapGeometry\_Geometry\_8X\_1Y** Geometry\_8X\_1Y  
**DeviceTapGeometry\_Geometry\_4X2\_1Y** Geometry\_4X2\_1Y  
**DeviceTapGeometry\_Geometry\_2X2E\_2YE** Geometry\_2X2E\_2YE  
**DeviceTapGeometry\_Geometry\_1X8** Geometry\_1X8  
**DeviceTapGeometry\_Geometry\_8X** Geometry\_8X

***DeviceTapGeometry\_Geometry\_4X2*** Geometry\_4X2  
***DeviceTapGeometry\_Geometry\_4X2E*** Geometry\_4X2E  
***DeviceTapGeometry\_Geometry\_4X2E\_1Y*** Geometry\_4X2E\_1Y  
***DeviceTapGeometry\_Geometry\_1X10\_1Y*** Geometry\_1X10\_1Y  
***DeviceTapGeometry\_Geometry\_10X\_1Y*** Geometry\_10X\_1Y  
***DeviceTapGeometry\_Geometry\_1X10*** Geometry\_1X10  
***DeviceTapGeometry\_Geometry\_10X*** Geometry\_10X  
***NUM\_DEVICETAPGEOMETRY***

#### 6.2.2.75 enum spinDeviceTemperatureSelectorEnums

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

Enumerator

***DeviceTemperatureSelector\_Sensor***  
***NUM\_DEVICETEMPERATURESELECTOR***

#### 6.2.2.76 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 enum spinDeviceTypeEnums

< Returns the device type.

Enumerator

***DeviceType\_Transmitter*** Data stream transmitter device.  
***DeviceType\_Receiver*** Data stream receiver device.  
***DeviceType\_Transceiver*** Data stream receiver and transmitter device.  
***DeviceType\_Peripheral*** Controllable device (with no data stream handling).  
***NUM\_DEVICETYPE***

## 6.2.2.78 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 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 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 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).

**EncoderResetSource\_LinkTrigger2** Resets on the reception of the chosen Link Trigger (received from the transport layer).

**NUM\_ENCODERRESETSOURCE**

## 6.2.2.82 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 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 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 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 enum spinEventNotificationEnums

< Enables/Disables the selected event.

Enumerator

***EventNotification\_On***  
***EventNotification\_Off***  
***NUM\_EVENTNOTIFICATION***

## 6.2.2.87 enum spinEventSelectorEnums

< Selects which Event to enable or disable.

Enumerator

***EventSelector\_Error***  
***EventSelector\_ExposureEnd***  
***EventSelector\_SerialPortReceive***  
***NUM\_EVENTSELECTOR***

## 6.2.2.88 enum spinExposureActiveModeEnums

< Control sensor active exposure mode.

Enumerator

***ExposureActiveMode\_Line1***  
***ExposureActiveMode\_AnyPixels***  
***ExposureActiveMode\_AllPixels***  
***NUM\_EXPOSUREACTIVEMODE***

## 6.2.2.89 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 enum spinExposureModeEnums

< Sets the operation mode of the Exposure.

## Enumerator

**ExposureMode\_Timed** Timed exposure. The exposure time is set using the ExposureTime or ExposureTimeAuto 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 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 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 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 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 enum spinFileOperationStatusEnums

< Represents the file operation execution status.

Enumerator

***FileOperationStatus\_Success*** File Operation was sucessful.  
***FileOperationStatus\_Failure*** File Operation failed.  
***FileOperationStatus\_Overflow*** An overflow occurred while executing the File Operation.  
***NUM\_FILEOPERATIONSTATUS***

#### 6.2.2.96 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 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 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 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 enum spinGevCCPEnums

< Controls the device access privilege of an application.

Enumerator

***GevCCP\_OpenAccess***

***GevCCP\_ExclusiveAccess***

***GevCCP\_ControlAccess***

***NUM\_GEVCCP***

### 6.2.2.101 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 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 enum spinGevGVSPExtendedIDModeEnums

< Enables the extended IDs mode.

#### Enumerator

***GevGVSPExtendedIDMode\_Off*** Off  
***GevGVSPExtendedIDMode\_On*** On  
***NUM\_GEVGVSPEXTENDEDIDMODE***

### 6.2.2.104 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\_GEVIIEEE1588CLOCKACCURACY***

## 6.2.2.105 enum spinGevIEEE1588ModeEnums

< Provides the mode of the IEEE 1588 clock.

Enumerator

**GevIEEE1588Mode\_Auto** Automatic  
**GevIEEE1588Mode\_SlaveOnly** Slave Only  
**NUM\_GEVIEEE1588MODE**

## 6.2.2.106 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 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 enum spinGevPhysicalLinkConfigurationEnums

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

Enumerator

**GevPhysicalLinkConfiguration\_SingleLink** Single Link  
**GevPhysicalLinkConfiguration\_MultiLink** Multi Link  
**GevPhysicalLinkConfiguration\_StaticLAG** Static LAG  
**GevPhysicalLinkConfiguration\_DynamicLAG** Dynamic LAG  
**NUM\_GEVPHYSICALLINKCONFIGURATION**

## 6.2.2.109 enum spinGevSupportedOptionSelectorEnums

< Selects the GEV option to interrogate for existing support.

Enumerator

*GevSupportedOptionSelector\_UserDefinedName*  
*GevSupportedOptionSelector\_SerialNumber*  
*GevSupportedOptionSelector\_HeartbeatDisable*  
*GevSupportedOptionSelector\_LinkSpeed*  
*GevSupportedOptionSelector\_CCPApplicationSocket*  
*GevSupportedOptionSelector\_ManifestTable*  
*GevSupportedOptionSelector\_TestData*  
*GevSupportedOptionSelector\_DiscoveryAckDelay*  
*GevSupportedOptionSelector\_DiscoveryAckDelayWritable*  
*GevSupportedOptionSelector\_ExtendedStatusCodes*  
*GevSupportedOptionSelector\_Action*  
*GevSupportedOptionSelector\_PendingAck*  
*GevSupportedOptionSelector\_EventData*  
*GevSupportedOptionSelector\_Event*  
*GevSupportedOptionSelector\_PacketResend*  
*GevSupportedOptionSelector\_WriteMem*  
*GevSupportedOptionSelector\_CommandsConcatenation*  
*GevSupportedOptionSelector\_IPConfigurationLLA*  
*GevSupportedOptionSelector\_IPConfigurationDHCP*  
*GevSupportedOptionSelector\_IPConfigurationPersistentIP*  
*GevSupportedOptionSelector\_StreamChannelSourceSocket*  
*GevSupportedOptionSelector\_MessageChannelSourceSocket*  
*NUM\_GEVSUPPORTEDOPTIONSELECTOR*

## 6.2.2.110 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 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\_BaselineStandard*** Indicates this is a baseline sequential (single-scan) DCT-based JPEG.

***ImageCompressionJPEGFormatOption\_BaselineOptimized*** Provides optimized color and slightly better compression than baseline standard by using custom Huffman tables optimized after statistical analysis of the image content.

***ImageCompressionJPEGFormatOption\_Progressive*** Indicates this is a progressive (multi-scan) DCT-based JPEG.

**NUM\_IMAGECOMPRESSIONJPEGFORMATOPTION**

#### 6.2.2.112 enum spinImageCompressionModeEnums

<

Enumerator

***ImageCompressionMode\_Off***

***ImageCompressionMode\_Lossless***

**NUM\_IMAGECOMPRESSIONMODE**

#### 6.2.2.113 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 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 enum spinLineInputFilterSelectorEnums

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

Enumerator

***LineInputFilterSelector\_Deglitch***  
***LineInputFilterSelector\_Debounce***  
***NUM\_LINEINPUTFILTERSELECTOR***

#### 6.2.2.116 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 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 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 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 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 enum spinLogicBlockLUTInputSourceEnums

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

Enumerator

***LogicBlockLUTInputSource\_Zero*** Zero  
***LogicBlockLUTInputSource\_Line0*** Line0  
***LogicBlockLUTInputSource\_Line1*** Line1  
***LogicBlockLUTInputSource\_Line2*** Line2  
***LogicBlockLUTInputSource\_Line3*** Line3  
***LogicBlockLUTInputSource\_UserOutput0*** UserOutput0  
***LogicBlockLUTInputSource\_UserOutput1*** UserOutput1  
***LogicBlockLUTInputSource\_UserOutput2*** UserOutput2  
***LogicBlockLUTInputSource\_UserOutput3*** UserOutput3  
***LogicBlockLUTInputSource\_Counter0Start*** Counter0Start  
***LogicBlockLUTInputSource\_Counter1Start*** Counter1Start  
***LogicBlockLUTInputSource\_Counter0End*** Counter0End  
***LogicBlockLUTInputSource\_Counter1End*** Counter1End  
***LogicBlockLUTInputSource\_LogicBlock0*** LogicBlock0  
***LogicBlockLUTInputSource\_LogicBlock1*** LogicBlock1  
***LogicBlockLUTInputSource\_ExposureStart*** ExposureStart  
***LogicBlockLUTInputSource\_ExposureEnd*** ExposureEnd  
***LogicBlockLUTInputSource\_FrameTriggerWait*** FrameTriggerWait  
***LogicBlockLUTInputSource\_AcquisitionActive*** AcquisitionActive  
***NUM\_LOGICBLOCKLUTINPUTSOURCE***

## 6.2.2.122 enum spinLogicBlockLUTSelectorEnums

< Selects which LogicBlock LUT to configure

Enumerator

***LogicBlockLUTSelector\_Value***  
***LogicBlockLUTSelector\_Enable***  
***NUM\_LOGICBLOCKLUTSELECTOR***

## 6.2.2.123 enum spinLogicBlockSelectorEnums

< Selects which LogicBlock to configure

Enumerator

***LogicBlockSelector\_LogicBlock0***  
***LogicBlockSelector\_LogicBlock1***  
***NUM\_LOGICBLOCKSELECTOR***

## 6.2.2.124 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 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 enum spinPixelFormatEnums

< Format of the pixel provided by the camera.

Enumerator

***PixelFormat\_Mono8***  
***PixelFormat\_Mono16***  
***PixelFormat\_RGB8Packed***  
***PixelFormat\_BayerGR8***  
***PixelFormat\_BayerRG8***  
***PixelFormat\_BayerGB8***  
***PixelFormat\_BayerBG8***  
***PixelFormat\_BayerGR16***  
***PixelFormat\_BayerRG16***  
***PixelFormat\_BayerGB16***  
***PixelFormat\_BayerBG16***  
***PixelFormat\_Mono12Packed***  
***PixelFormat\_BayerGR12Packed***  
***PixelFormat\_BayerRG12Packed***  
***PixelFormat\_BayerGB12Packed***  
***PixelFormat\_BayerBG12Packed***  
***PixelFormat\_YUV411Packed***

***PixelFormat\_YUV422Packed***  
***PixelFormat\_YUV444Packed***  
***PixelFormat\_Mono12p***  
***PixelFormat\_BayerGR12p***  
***PixelFormat\_BayerRG12p***  
***PixelFormat\_BayerGB12p***  
***PixelFormat\_BayerBG12p***  
***PixelFormat\_YCbCr8***  
***PixelFormat\_YCbCr422\_8***  
***PixelFormat\_YCbCr411\_8***  
***PixelFormat\_BGR8***  
***PixelFormat\_BGRa8***  
***PixelFormat\_Mono10Packed***  
***PixelFormat\_BayerGR10Packed***  
***PixelFormat\_BayerRG10Packed***  
***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\_BayerBG10*** Bayer Blue-Green 10-bit unpacked  
***PixelFormat\_BayerBG12*** Bayer Blue-Green 12-bit unpacked  
***PixelFormat\_BayerGB10*** Bayer Green-Blue 10-bit unpacked  
***PixelFormat\_BayerGB12*** Bayer Green-Blue 12-bit unpacked  
***PixelFormat\_BayerGR10*** Bayer Green-Red 10-bit unpacked  
***PixelFormat\_BayerGR12*** Bayer Green-Red 12-bit unpacked  
***PixelFormat\_BayerRG10*** Bayer Red-Green 10-bit unpacked  
***PixelFormat\_BayerRG12*** Bayer Red-Green 12-bit unpacked  
***PixelFormat\_RGBa8*** Red-Green-Blue-alpha 8-bit  
***PixelFormat\_RGBa10*** Red-Green-Blue-alpha 10-bit unpacked  
***PixelFormat\_RGBa10p*** Red-Green-Blue-alpha 10-bit packed  
***PixelFormat\_RGBa12*** Red-Green-Blue-alpha 12-bit unpacked  
***PixelFormat\_RGBa12p*** Red-Green-Blue-alpha 12-bit packed  
***PixelFormat\_RGBa14*** Red-Green-Blue-alpha 14-bit unpacked  
***PixelFormat\_RGBa16*** Red-Green-Blue-alpha 16-bit  
***PixelFormat\_RGB8*** Red-Green-Blue 8-bit

***PixelFormat\_RGB8\_Planar*** Red-Green-Blue 8-bit planar  
***PixelFormat\_RGB10*** Red-Green-Blue 10-bit unpacked  
***PixelFormat\_RGB10\_Planar*** Red-Green-Blue 10-bit unpacked planar  
***PixelFormat\_RGB10p*** Red-Green-Blue 10-bit packed  
***PixelFormat\_RGB10p32*** Red-Green-Blue 10-bit packed into 32-bit  
***PixelFormat\_RGB12*** Red-Green-Blue 12-bit unpacked  
***PixelFormat\_RGB12\_Planar*** Red-Green-Blue 12-bit unpacked planar  
***PixelFormat\_RGB12p*** Red-Green-Blue 12-bit packed  
***PixelFormat\_RGB14*** Red-Green-Blue 14-bit unpacked  
***PixelFormat\_RGB16*** Red-Green-Blue 16-bit  
***PixelFormat\_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\_BGR10*** Blue-Green-Red 10-bit unpacked  
***PixelFormat\_BGR10p*** Blue-Green-Red 10-bit packed  
***PixelFormat\_BGR12*** Blue-Green-Red 12-bit unpacked  
***PixelFormat\_BGR12p*** Blue-Green-Red 12-bit packed  
***PixelFormat\_BGR14*** Blue-Green-Red 14-bit unpacked  
***PixelFormat\_BGR16*** Blue-Green-Red 16-bit  
***PixelFormat\_BGR565p*** Blue-Green-Red 5/6/5-bit packed  
***PixelFormat\_R8*** Red 8-bit  
***PixelFormat\_R10*** Red 10-bit  
***PixelFormat\_R12*** Red 12-bit  
***PixelFormat\_R16*** Red 16-bit  
***PixelFormat\_G8*** Green 8-bit  
***PixelFormat\_G10*** Green 10-bit  
***PixelFormat\_G12*** Green 12-bit  
***PixelFormat\_G16*** Green 16-bit  
***PixelFormat\_B8*** Blue 8-bit  
***PixelFormat\_B10*** Blue 10-bit  
***PixelFormat\_B12*** Blue 12-bit  
***PixelFormat\_B16*** Blue 16-bit  
***PixelFormat\_Coord3D\_ABC8*** 3D coordinate A-B-C 8-bit  
***PixelFormat\_Coord3D\_ABC8\_Planar*** 3D coordinate A-B-C 8-bit planar  
***PixelFormat\_Coord3D\_ABC10p*** 3D coordinate A-B-C 10-bit packed  
***PixelFormat\_Coord3D\_ABC10p\_Planar*** 3D coordinate A-B-C 10-bit packed planar  
***PixelFormat\_Coord3D\_ABC12p*** 3D coordinate A-B-C 12-bit packed  
***PixelFormat\_Coord3D\_ABC12p\_Planar*** 3D coordinate A-B-C 12-bit packed planar  
***PixelFormat\_Coord3D\_ABC16*** 3D coordinate A-B-C 16-bit  
***PixelFormat\_Coord3D\_ABC16\_Planar*** 3D coordinate A-B-C 16-bit planar

***PixelFormat\_Coord3D\_ABC32f*** 3D coordinate A-B-C 32-bit floating point

***PixelFormat\_Coord3D\_ABC32f\_Planar*** 3D coordinate A-B-C 32-bit floating point planar

***PixelFormat\_Coord3D\_AC8*** 3D coordinate A-C 8-bit

***PixelFormat\_Coord3D\_AC8\_Planar*** 3D coordinate A-C 8-bit planar

***PixelFormat\_Coord3D\_AC10p*** 3D coordinate A-C 10-bit packed

***PixelFormat\_Coord3D\_AC10p\_Planar*** 3D coordinate A-C 10-bit packed planar

***PixelFormat\_Coord3D\_AC12p*** 3D coordinate A-C 12-bit packed

***PixelFormat\_Coord3D\_AC12p\_Planar*** 3D coordinate A-C 12-bit packed planar

***PixelFormat\_Coord3D\_AC16*** 3D coordinate A-C 16-bit

***PixelFormat\_Coord3D\_AC16\_Planar*** 3D coordinate A-C 16-bit planar

***PixelFormat\_Coord3D\_AC32f*** 3D coordinate A-C 32-bit floating point

***PixelFormat\_Coord3D\_AC32f\_Planar*** 3D coordinate A-C 32-bit floating point planar

***PixelFormat\_Coord3D\_A8*** 3D coordinate A 8-bit

***PixelFormat\_Coord3D\_A10p*** 3D coordinate A 10-bit packed

***PixelFormat\_Coord3D\_A12p*** 3D coordinate A 12-bit packed

***PixelFormat\_Coord3D\_A16*** 3D coordinate A 16-bit

***PixelFormat\_Coord3D\_A32f*** 3D coordinate A 32-bit floating point

***PixelFormat\_Coord3D\_B8*** 3D coordinate B 8-bit

***PixelFormat\_Coord3D\_B10p*** 3D coordinate B 10-bit packed

***PixelFormat\_Coord3D\_B12p*** 3D coordinate B 12-bit packed

***PixelFormat\_Coord3D\_B16*** 3D coordinate B 16-bit

***PixelFormat\_Coord3D\_B32f*** 3D coordinate B 32-bit floating point

***PixelFormat\_Coord3D\_C8*** 3D coordinate C 8-bit

***PixelFormat\_Coord3D\_C10p*** 3D coordinate C 10-bit packed

***PixelFormat\_Coord3D\_C12p*** 3D coordinate C 12-bit packed

***PixelFormat\_Coord3D\_C16*** 3D coordinate C 16-bit

***PixelFormat\_Coord3D\_C32f*** 3D coordinate C 32-bit floating point

***PixelFormat\_Confidence1*** Confidence 1-bit unpacked

***PixelFormat\_Confidence1p*** Confidence 1-bit packed

***PixelFormat\_Confidence8*** Confidence 8-bit

***PixelFormat\_Confidence16*** Confidence 16-bit

***PixelFormat\_Confidence32f*** Confidence 32-bit floating point

***PixelFormat\_BiColorBGRG8*** Bi-color Blue/Green - Red/Green 8-bit

***PixelFormat\_BiColorBGRG10*** Bi-color Blue/Green - Red/Green 10-bit unpacked

***PixelFormat\_BiColorBGRG10p*** Bi-color Blue/Green - Red/Green 10-bit packed

***PixelFormat\_BiColorBGRG12*** Bi-color Blue/Green - Red/Green 12-bit unpacked

***PixelFormat\_BiColorBGRG12p*** Bi-color Blue/Green - Red/Green 12-bit packed

***PixelFormat\_BiColorRGBG8*** Bi-color Red/Green - Blue/Green 8-bit

***PixelFormat\_BiColorRGBG10*** Bi-color Red/Green - Blue/Green 10-bit unpacked

***PixelFormat\_BiColorRGBG10p*** Bi-color Red/Green - Blue/Green 10-bit packed

***PixelFormat\_BiColorRGBG12*** Bi-color Red/Green - Blue/Green 12-bit unpacked

***PixelFormat\_BiColorRGBG12p*** Bi-color Red/Green - Blue/Green 12-bit packed

***PixelFormat\_SCF1WBWG8*** Sparse Color Filter #1 White-Blue-White-Green 8-bit

***PixelFormat\_SCF1WBWG10*** Sparse Color Filter #1 White-Blue-White-Green 10-bit unpacked

***PixelFormat\_SCF1WBWG10p*** Sparse Color Filter #1 White-Blue-White-Green 10-bit packed

**PixelFormat\_SCF1WBWG12** Sparse Color Filter #1 White-Blue-White-Green 12-bit unpacked  
**PixelFormat\_SCF1WBWG12p** Sparse Color Filter #1 White-Blue-White-Green 12-bit packed  
**PixelFormat\_SCF1WBWG14** Sparse Color Filter #1 White-Blue-White-Green 14-bit unpacked  
**PixelFormat\_SCF1WBWG16** Sparse Color Filter #1 White-Blue-White-Green 16-bit unpacked  
**PixelFormat\_SCF1WGW8** Sparse Color Filter #1 White-Green-White-Blue 8-bit  
**PixelFormat\_SCF1WGW10** Sparse Color Filter #1 White-Green-White-Blue 10-bit unpacked  
**PixelFormat\_SCF1WGW10p** Sparse Color Filter #1 White-Green-White-Blue 10-bit packed  
**PixelFormat\_SCF1WGW12** Sparse Color Filter #1 White-Green-White-Blue 12-bit unpacked  
**PixelFormat\_SCF1WGW12p** Sparse Color Filter #1 White-Green-White-Blue 12-bit packed  
**PixelFormat\_SCF1WGW14** Sparse Color Filter #1 White-Green-White-Blue 14-bit unpacked  
**PixelFormat\_SCF1WGW16** Sparse Color Filter #1 White-Green-White-Blue 16-bit  
**PixelFormat\_SCF1WGW8** Sparse Color Filter #1 White-Green-White-Red 8-bit  
**PixelFormat\_SCF1WGW10** Sparse Color Filter #1 White-Green-White-Red 10-bit unpacked  
**PixelFormat\_SCF1WGW10p** Sparse Color Filter #1 White-Green-White-Red 10-bit packed  
**PixelFormat\_SCF1WGW12** Sparse Color Filter #1 White-Green-White-Red 12-bit unpacked  
**PixelFormat\_SCF1WGW12p** Sparse Color Filter #1 White-Green-White-Red 12-bit packed  
**PixelFormat\_SCF1WGW14** Sparse Color Filter #1 White-Green-White-Red 14-bit unpacked  
**PixelFormat\_SCF1WGW16** Sparse Color Filter #1 White-Green-White-Red 16-bit  
**PixelFormat\_SCF1WRWG8** Sparse Color Filter #1 White-Red-White-Green 8-bit  
**PixelFormat\_SCF1WRWG10** Sparse Color Filter #1 White-Red-White-Green 10-bit unpacked  
**PixelFormat\_SCF1WRWG10p** Sparse Color Filter #1 White-Red-White-Green 10-bit packed  
**PixelFormat\_SCF1WRWG12** Sparse Color Filter #1 White-Red-White-Green 12-bit unpacked  
**PixelFormat\_SCF1WRWG12p** Sparse Color Filter #1 White-Red-White-Green 12-bit packed  
**PixelFormat\_SCF1WRWG14** Sparse Color Filter #1 White-Red-White-Green 14-bit unpacked  
**PixelFormat\_SCF1WRWG16** Sparse Color Filter #1 White-Red-White-Green 16-bit  
**PixelFormat\_YCbCr8\_CbYCr** YCbCr 4:4:4 8-bit  
**PixelFormat\_YCbCr10\_CbYCr** YCbCr 4:4:4 10-bit unpacked  
**PixelFormat\_YCbCr10p\_CbYCr** YCbCr 4:4:4 10-bit packed  
**PixelFormat\_YCbCr12\_CbYCr** YCbCr 4:4:4 12-bit unpacked  
**PixelFormat\_YCbCr12p\_CbYCr** YCbCr 4:4:4 12-bit packed  
**PixelFormat\_YCbCr411\_8\_CbYYCrYY** YCbCr 4:1:1 8-bit  
**PixelFormat\_YCbCr422\_8\_CbYCrY** YCbCr 4:2:2 8-bit  
**PixelFormat\_YCbCr422\_10** YCbCr 4:2:2 10-bit unpacked  
**PixelFormat\_YCbCr422\_10\_CbYCrY** YCbCr 4:2:2 10-bit unpacked  
**PixelFormat\_YCbCr422\_10p** YCbCr 4:2:2 10-bit packed  
**PixelFormat\_YCbCr422\_10p\_CbYCrY** YCbCr 4:2:2 10-bit packed  
**PixelFormat\_YCbCr422\_12** YCbCr 4:2:2 12-bit unpacked  
**PixelFormat\_YCbCr422\_12\_CbYCrY** YCbCr 4:2:2 12-bit unpacked  
**PixelFormat\_YCbCr422\_12p** YCbCr 4:2:2 12-bit packed  
**PixelFormat\_YCbCr422\_12p\_CbYCrY** YCbCr 4:2:2 12-bit packed  
**PixelFormat\_YCbCr601\_8\_CbYCr** YCbCr 4:4:4 8-bit BT.601  
**PixelFormat\_YCbCr601\_10\_CbYCr** YCbCr 4:4:4 10-bit unpacked BT.601  
**PixelFormat\_YCbCr601\_10p\_CbYCr** YCbCr 4:4:4 10-bit packed BT.601  
**PixelFormat\_YCbCr601\_12\_CbYCr** YCbCr 4:4:4 12-bit unpacked BT.601  
**PixelFormat\_YCbCr601\_12p\_CbYCr** YCbCr 4:4:4 12-bit packed BT.601

**PixelFormat\_YCbCr601\_411\_8\_CbYYCrYY** YCbCr 4:1:1 8-bit BT.601  
**PixelFormat\_YCbCr601\_422\_8** YCbCr 4:2:2 8-bit BT.601  
**PixelFormat\_YCbCr601\_422\_8\_CbYCrY** YCbCr 4:2:2 8-bit BT.601  
**PixelFormat\_YCbCr601\_422\_10** YCbCr 4:2:2 10-bit unpacked BT.601  
**PixelFormat\_YCbCr601\_422\_10\_CbYCrY** YCbCr 4:2:2 10-bit unpacked BT.601  
**PixelFormat\_YCbCr601\_422\_10p** YCbCr 4:2:2 10-bit packed BT.601  
**PixelFormat\_YCbCr601\_422\_10p\_CbYCrY** YCbCr 4:2:2 10-bit packed BT.601  
**PixelFormat\_YCbCr601\_422\_12** YCbCr 4:2:2 12-bit unpacked BT.601  
**PixelFormat\_YCbCr601\_422\_12\_CbYCrY** YCbCr 4:2:2 12-bit unpacked BT.601  
**PixelFormat\_YCbCr601\_422\_12p** YCbCr 4:2:2 12-bit packed BT.601  
**PixelFormat\_YCbCr601\_422\_12p\_CbYCrY** YCbCr 4:2:2 12-bit packed BT.601  
**PixelFormat\_YCbCr709\_8\_CbYCr** YCbCr 4:4:4 8-bit BT.709  
**PixelFormat\_YCbCr709\_10\_CbYCr** YCbCr 4:4:4 10-bit unpacked BT.709  
**PixelFormat\_YCbCr709\_10p\_CbYCr** YCbCr 4:4:4 10-bit packed BT.709  
**PixelFormat\_YCbCr709\_12\_CbYCr** YCbCr 4:4:4 12-bit unpacked BT.709  
**PixelFormat\_YCbCr709\_12p\_CbYCr** YCbCr 4:4:4 12-bit packed BT.709  
**PixelFormat\_YCbCr709\_411\_8\_CbYYCrYY** YCbCr 4:1:1 8-bit BT.709  
**PixelFormat\_YCbCr709\_422\_8** YCbCr 4:2:2 8-bit BT.709  
**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\_YYYVYY** 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\_Raw16** Raw 16 bit.  
**PixelFormat\_Raw8** Raw bit.  
**PixelFormat\_R12\_Jpeg** Red 12-bit JPEG.  
**PixelFormat\_GR12\_Jpeg** Green Red 12-bit JPEG.  
**PixelFormat\_GB12\_Jpeg** Green Blue 12-bit JPEG.  
**PixelFormat\_B12\_Jpeg** Blue 12-bit packed JPEG.  
**UNKNOWN\_PIXELFORMAT**  
**NUM\_PIXELFORMAT**



## 6.2.2.127 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\_BayerBG8*** Bayer Blue-Green 8-bit  
***PixelFormatInfoSelector\_BayerBG10*** Bayer Blue-Green 10-bit unpacked  
***PixelFormatInfoSelector\_BayerBG10p*** Bayer Blue-Green 10-bit packed  
***PixelFormatInfoSelector\_BayerBG12*** Bayer Blue-Green 12-bit unpacked  
***PixelFormatInfoSelector\_BayerBG12p*** Bayer Blue-Green 12-bit packed  
***PixelFormatInfoSelector\_BayerBG16*** Bayer Blue-Green 16-bit  
***PixelFormatInfoSelector\_BayerGB8*** Bayer Green-Blue 8-bit  
***PixelFormatInfoSelector\_BayerGB10*** Bayer Green-Blue 10-bit unpacked  
***PixelFormatInfoSelector\_BayerGB10p*** Bayer Green-Blue 10-bit packed  
***PixelFormatInfoSelector\_BayerGB12*** Bayer Green-Blue 12-bit unpacked  
***PixelFormatInfoSelector\_BayerGB12p*** Bayer Green-Blue 12-bit packed  
***PixelFormatInfoSelector\_BayerGB16*** Bayer Green-Blue 16-bit  
***PixelFormatInfoSelector\_BayerGR8*** Bayer Green-Red 8-bit  
***PixelFormatInfoSelector\_BayerGR10*** Bayer Green-Red 10-bit unpacked  
***PixelFormatInfoSelector\_BayerGR10p*** Bayer Green-Red 10-bit packed  
***PixelFormatInfoSelector\_BayerGR12*** Bayer Green-Red 12-bit unpacked  
***PixelFormatInfoSelector\_BayerGR12p*** Bayer Green-Red 12-bit packed  
***PixelFormatInfoSelector\_BayerGR16*** Bayer Green-Red 16-bit  
***PixelFormatInfoSelector\_BayerRG8*** Bayer Red-Green 8-bit  
***PixelFormatInfoSelector\_BayerRG10*** Bayer Red-Green 10-bit unpacked  
***PixelFormatInfoSelector\_BayerRG10p*** Bayer Red-Green 10-bit packed  
***PixelFormatInfoSelector\_BayerRG12*** Bayer Red-Green 12-bit unpacked  
***PixelFormatInfoSelector\_BayerRG12p*** Bayer Red-Green 12-bit packed  
***PixelFormatInfoSelector\_BayerRG16*** Bayer Red-Green 16-bit  
***PixelFormatInfoSelector\_RGBa8*** Red-Green-Blue-alpha 8-bit  
***PixelFormatInfoSelector\_RGBa10*** Red-Green-Blue-alpha 10-bit unpacked  
***PixelFormatInfoSelector\_RGBa10p*** Red-Green-Blue-alpha 10-bit packed  
***PixelFormatInfoSelector\_RGBa12*** Red-Green-Blue-alpha 12-bit unpacked  
***PixelFormatInfoSelector\_RGBa12p*** Red-Green-Blue-alpha 12-bit packed

***PixelFormatInfoSelector\_RGBA14*** Red-Green-Blue-alpha 14-bit unpacked  
***PixelFormatInfoSelector\_RGBA16*** Red-Green-Blue-alpha 16-bit  
***PixelFormatInfoSelector\_RGB8*** Red-Green-Blue 8-bit  
***PixelFormatInfoSelector\_RGB8\_Planar*** Red-Green-Blue 8-bit planar  
***PixelFormatInfoSelector\_RGB10*** Red-Green-Blue 10-bit unpacked  
***PixelFormatInfoSelector\_RGB10\_Planar*** Red-Green-Blue 10-bit unpacked planar  
***PixelFormatInfoSelector\_RGB10p*** Red-Green-Blue 10-bit packed  
***PixelFormatInfoSelector\_RGB10p32*** Red-Green-Blue 10-bit packed into 32-bit  
***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\_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\_BGR8*** Blue-Green-Red 8-bit  
***PixelFormatInfoSelector\_BGR10*** Blue-Green-Red 10-bit unpacked  
***PixelFormatInfoSelector\_BGR10p*** Blue-Green-Red 10-bit packed  
***PixelFormatInfoSelector\_BGR12*** Blue-Green-Red 12-bit unpacked  
***PixelFormatInfoSelector\_BGR12p*** Blue-Green-Red 12-bit packed  
***PixelFormatInfoSelector\_BGR14*** Blue-Green-Red 14-bit unpacked  
***PixelFormatInfoSelector\_BGR16*** Blue-Green-Red 16-bit  
***PixelFormatInfoSelector\_BGR565p*** Blue-Green-Red 5/6/5-bit packed  
***PixelFormatInfoSelector\_R8*** Red 8-bit  
***PixelFormatInfoSelector\_R10*** Red 10-bit  
***PixelFormatInfoSelector\_R12*** Red 12-bit  
***PixelFormatInfoSelector\_R16*** Red 16-bit  
***PixelFormatInfoSelector\_G8*** Green 8-bit  
***PixelFormatInfoSelector\_G10*** Green 10-bit  
***PixelFormatInfoSelector\_G12*** Green 12-bit  
***PixelFormatInfoSelector\_G16*** Green 16-bit  
***PixelFormatInfoSelector\_B8*** Blue 8-bit  
***PixelFormatInfoSelector\_B10*** Blue 10-bit  
***PixelFormatInfoSelector\_B12*** Blue 12-bit  
***PixelFormatInfoSelector\_B16*** Blue 16-bit  
***PixelFormatInfoSelector\_Coord3D\_ABC8*** 3D coordinate A-B-C 8-bit  
***PixelFormatInfoSelector\_Coord3D\_ABC8\_Planar*** 3D coordinate A-B-C 8-bit planar  
***PixelFormatInfoSelector\_Coord3D\_ABC10p*** 3D coordinate A-B-C 10-bit packed

***PixelFormatInfoSelector\_Coord3D\_ABC10p\_Planar*** 3D coordinate A-B-C 10-bit packed planar  
***PixelFormatInfoSelector\_Coord3D\_ABC12p*** 3D coordinate A-B-C 12-bit packed  
***PixelFormatInfoSelector\_Coord3D\_ABC12p\_Planar*** 3D coordinate A-B-C 12-bit packed planar  
***PixelFormatInfoSelector\_Coord3D\_ABC16*** 3D coordinate A-B-C 16-bit  
***PixelFormatInfoSelector\_Coord3D\_ABC16\_Planar*** 3D coordinate A-B-C 16-bit planar  
***PixelFormatInfoSelector\_Coord3D\_ABC32f*** 3D coordinate A-B-C 32-bit floating point  
***PixelFormatInfoSelector\_Coord3D\_ABC32f\_Planar*** 3D coordinate A-B-C 32-bit floating point planar  
***PixelFormatInfoSelector\_Coord3D\_AC8*** 3D coordinate A-C 8-bit  
***PixelFormatInfoSelector\_Coord3D\_AC8\_Planar*** 3D coordinate A-C 8-bit planar  
***PixelFormatInfoSelector\_Coord3D\_AC10p*** 3D coordinate A-C 10-bit packed  
***PixelFormatInfoSelector\_Coord3D\_AC10p\_Planar*** 3D coordinate A-C 10-bit packed planar  
***PixelFormatInfoSelector\_Coord3D\_AC12p*** 3D coordinate A-C 12-bit packed  
***PixelFormatInfoSelector\_Coord3D\_AC12p\_Planar*** 3D coordinate A-C 12-bit packed planar  
***PixelFormatInfoSelector\_Coord3D\_AC16*** 3D coordinate A-C 16-bit  
***PixelFormatInfoSelector\_Coord3D\_AC16\_Planar*** 3D coordinate A-C 16-bit planar  
***PixelFormatInfoSelector\_Coord3D\_AC32f*** 3D coordinate A-C 32-bit floating point  
***PixelFormatInfoSelector\_Coord3D\_AC32f\_Planar*** 3D coordinate A-C 32-bit floating point planar  
***PixelFormatInfoSelector\_Coord3D\_A8*** 3D coordinate A 8-bit  
***PixelFormatInfoSelector\_Coord3D\_A10p*** 3D coordinate A 10-bit packed  
***PixelFormatInfoSelector\_Coord3D\_A12p*** 3D coordinate A 12-bit packed  
***PixelFormatInfoSelector\_Coord3D\_A16*** 3D coordinate A 16-bit  
***PixelFormatInfoSelector\_Coord3D\_A32f*** 3D coordinate A 32-bit floating point  
***PixelFormatInfoSelector\_Coord3D\_B8*** 3D coordinate B 8-bit  
***PixelFormatInfoSelector\_Coord3D\_B10p*** 3D coordinate B 10-bit packed  
***PixelFormatInfoSelector\_Coord3D\_B12p*** 3D coordinate B 12-bit packed  
***PixelFormatInfoSelector\_Coord3D\_B16*** 3D coordinate B 16-bit  
***PixelFormatInfoSelector\_Coord3D\_B32f*** 3D coordinate B 32-bit floating point  
***PixelFormatInfoSelector\_Coord3D\_C8*** 3D coordinate C 8-bit  
***PixelFormatInfoSelector\_Coord3D\_C10p*** 3D coordinate C 10-bit packed  
***PixelFormatInfoSelector\_Coord3D\_C12p*** 3D coordinate C 12-bit packed  
***PixelFormatInfoSelector\_Coord3D\_C16*** 3D coordinate C 16-bit  
***PixelFormatInfoSelector\_Coord3D\_C32f*** 3D coordinate C 32-bit floating point  
***PixelFormatInfoSelector\_Confidence1*** Confidence 1-bit unpacked  
***PixelFormatInfoSelector\_Confidence1p*** Confidence 1-bit packed  
***PixelFormatInfoSelector\_Confidence8*** Confidence 8-bit  
***PixelFormatInfoSelector\_Confidence16*** Confidence 16-bit  
***PixelFormatInfoSelector\_Confidence32f*** Confidence 32-bit floating point  
***PixelFormatInfoSelector\_BiColorBGRG8*** Bi-color Blue/Green - Red/Green 8-bit  
***PixelFormatInfoSelector\_BiColorBGRG10*** Bi-color Blue/Green - Red/Green 10-bit unpacked  
***PixelFormatInfoSelector\_BiColorBGRG10p*** Bi-color Blue/Green - Red/Green 10-bit packed  
***PixelFormatInfoSelector\_BiColorBGRG12*** Bi-color Blue/Green - Red/Green 12-bit unpacked  
***PixelFormatInfoSelector\_BiColorBGRG12p*** Bi-color Blue/Green - Red/Green 12-bit packed  
***PixelFormatInfoSelector\_BiColorRGBG8*** Bi-color Red/Green - Blue/Green 8-bit  
***PixelFormatInfoSelector\_BiColorRGBG10*** Bi-color Red/Green - Blue/Green 10-bit unpacked  
***PixelFormatInfoSelector\_BiColorRGBG10p*** Bi-color Red/Green - Blue/Green 10-bit packed

***PixelFormatInfoSelector\_BiColorRGBG12*** Bi-color Red/Green - Blue/Green 12-bit unpacked  
***PixelFormatInfoSelector\_BiColorRGBG12p*** Bi-color Red/Green - Blue/Green 12-bit packed  
***PixelFormatInfoSelector\_SCF1WBWG8*** Sparse Color Filter #1 White-Blue-White-Green 8-bit  
***PixelFormatInfoSelector\_SCF1WBWG10*** Sparse Color Filter #1 White-Blue-White-Green 10-bit unpacked  
  
***PixelFormatInfoSelector\_SCF1WBWG10p*** Sparse Color Filter #1 White-Blue-White-Green 10-bit packed  
***PixelFormatInfoSelector\_SCF1WBWG12*** Sparse Color Filter #1 White-Blue-White-Green 12-bit unpacked  
  
***PixelFormatInfoSelector\_SCF1WBWG12p*** Sparse Color Filter #1 White-Blue-White-Green 12-bit packed  
***PixelFormatInfoSelector\_SCF1WBWG14*** Sparse Color Filter #1 White-Blue-White-Green 14-bit unpacked  
  
***PixelFormatInfoSelector\_SCF1WBWG16*** Sparse Color Filter #1 White-Blue-White-Green 16-bit unpacked  
  
***PixelFormatInfoSelector\_SCF1WGWB8*** Sparse Color Filter #1 White-Green-White-Blue 8-bit  
***PixelFormatInfoSelector\_SCF1WGWB10*** Sparse Color Filter #1 White-Green-White-Blue 10-bit unpacked  
  
***PixelFormatInfoSelector\_SCF1WGWB10p*** Sparse Color Filter #1 White-Green-White-Blue 10-bit packed  
***PixelFormatInfoSelector\_SCF1WGWB12*** Sparse Color Filter #1 White-Green-White-Blue 12-bit unpacked  
  
***PixelFormatInfoSelector\_SCF1WGWB12p*** Sparse Color Filter #1 White-Green-White-Blue 12-bit packed  
***PixelFormatInfoSelector\_SCF1WGWB14*** Sparse Color Filter #1 White-Green-White-Blue 14-bit unpacked  
  
***PixelFormatInfoSelector\_SCF1WGWB16*** Sparse Color Filter #1 White-Green-White-Blue 16-bit  
***PixelFormatInfoSelector\_SCF1WGWR8*** Sparse Color Filter #1 White-Green-White-Red 8-bit  
***PixelFormatInfoSelector\_SCF1WGWR10*** Sparse Color Filter #1 White-Green-White-Red 10-bit unpacked  
  
***PixelFormatInfoSelector\_SCF1WGWR10p*** Sparse Color Filter #1 White-Green-White-Red 10-bit packed  
***PixelFormatInfoSelector\_SCF1WGWR12*** Sparse Color Filter #1 White-Green-White-Red 12-bit unpacked  
  
***PixelFormatInfoSelector\_SCF1WGWR12p*** Sparse Color Filter #1 White-Green-White-Red 12-bit packed  
***PixelFormatInfoSelector\_SCF1WGWR14*** Sparse Color Filter #1 White-Green-White-Red 14-bit unpacked  
  
***PixelFormatInfoSelector\_SCF1WGWR16*** Sparse Color Filter #1 White-Green-White-Red 16-bit  
***PixelFormatInfoSelector\_SCF1WRWG8*** Sparse Color Filter #1 White-Red-White-Green 8-bit  
***PixelFormatInfoSelector\_SCF1WRWG10*** Sparse Color Filter #1 White-Red-White-Green 10-bit unpacked  
  
***PixelFormatInfoSelector\_SCF1WRWG10p*** Sparse Color Filter #1 White-Red-White-Green 10-bit packed  
***PixelFormatInfoSelector\_SCF1WRWG12*** Sparse Color Filter #1 White-Red-White-Green 12-bit unpacked  
  
***PixelFormatInfoSelector\_SCF1WRWG12p*** Sparse Color Filter #1 White-Red-White-Green 12-bit packed  
***PixelFormatInfoSelector\_SCF1WRWG14*** Sparse Color Filter #1 White-Red-White-Green 14-bit unpacked  
  
***PixelFormatInfoSelector\_SCF1WRWG16*** Sparse Color Filter #1 White-Red-White-Green 16-bit  
***PixelFormatInfoSelector\_YCbCr8*** YCbCr 4:4:4 8-bit  
***PixelFormatInfoSelector\_YCbCr8\_CbYCr*** YCbCr 4:4:4 8-bit  
***PixelFormatInfoSelector\_YCbCr10\_CbYCr*** YCbCr 4:4:4 10-bit unpacked  
***PixelFormatInfoSelector\_YCbCr10p\_CbYCr*** YCbCr 4:4:4 10-bit packed  
***PixelFormatInfoSelector\_YCbCr12\_CbYCr*** YCbCr 4:4:4 12-bit unpacked

*PixelFormatInfoSelector\_YCbCr12p\_CbYCr* YCbCr 4:4:4 12-bit packed

*PixelFormatInfoSelector\_YCbCr411\_8* YCbCr 4:1:1 8-bit

*PixelFormatInfoSelector\_YCbCr411\_8\_CbYYCrYY* YCbCr 4:1:1 8-bit

*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\_CbYCrY* 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\_CbYCrY* 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\_CbYCrY* 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\_CbYCrY* YCbCr 4:2:2 12-bit packed BT.601

*PixelFormatInfoSelector\_YCbCr709\_8\_CbYCr* YCbCr 4:4:4 8-bit BT.709

*PixelFormatInfoSelector\_YCbCr709\_10\_CbYCr* YCbCr 4:4:4 10-bit unpacked BT.709

*PixelFormatInfoSelector\_YCbCr709\_10p\_CbYCr* YCbCr 4:4:4 10-bit packed BT.709

*PixelFormatInfoSelector\_YCbCr709\_12\_CbYCr* YCbCr 4:4:4 12-bit unpacked BT.709

*PixelFormatInfoSelector\_YCbCr709\_12p\_CbYCr* YCbCr 4:4:4 12-bit packed BT.709

*PixelFormatInfoSelector\_YCbCr709\_411\_8\_CbYYCrYY* 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\_CbYCrY* 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\_CbYCrY* 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\_CbYCrY* 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\_CbYCrY* YCbCr 4:2:2 12-bit packed BT.709

***PixelFormatInfoSelector\_YUV8\_UYV*** YUV 4:4:4 8-bit  
***PixelFormatInfoSelector\_YUV411\_8\_UYYVYY*** YUV 4:1:1 8-bit  
***PixelFormatInfoSelector\_YUV422\_8*** YUV 4:2:2 8-bit  
***PixelFormatInfoSelector\_YUV422\_8\_UYVY*** YUV 4:2:2 8-bit  
***PixelFormatInfoSelector\_Polarized8*** Monochrome Polarized 8-bit  
***PixelFormatInfoSelector\_Polarized10p*** Monochrome Polarized 10-bit packed  
***PixelFormatInfoSelector\_Polarized12p*** Monochrome Polarized 12-bit packed  
***PixelFormatInfoSelector\_Polarized16*** Monochrome Polarized 16-bit  
***PixelFormatInfoSelector\_BayerRGPolarized8*** Polarized Bayer Red Green filter 8-bit  
***PixelFormatInfoSelector\_BayerRGPolarized10p*** Polarized Bayer Red Green filter 10-bit packed  
***PixelFormatInfoSelector\_BayerRGPolarized12p*** Polarized Bayer Red Green filter 12-bit packed  
***PixelFormatInfoSelector\_BayerRGPolarized16*** Polarized Bayer Red Green filter 16-bit  
***NUM\_PIXELFORMATINFOSELECTOR***

#### 6.2.2.128 enum spinPixelSizeEnums

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

##### Enumerator

***PixelSize\_Bpp1*** 1 bit per pixel.  
***PixelSize\_Bpp2*** 2 bits per pixel.  
***PixelSize\_Bpp4*** 4 bits per pixel.  
***PixelSize\_Bpp8*** 8 bits per pixel.  
***PixelSize\_Bpp10*** 10 bits per pixel.  
***PixelSize\_Bpp12*** 12 bits per pixel.  
***PixelSize\_Bpp14*** 14 bits per pixel.  
***PixelSize\_Bpp16*** 16 bits per pixel.  
***PixelSize\_Bpp20*** 20 bits per pixel.  
***PixelSize\_Bpp24*** 24 bits per pixel.  
***PixelSize\_Bpp30*** 30 bits per pixel.  
***PixelSize\_Bpp32*** 32 bits per pixel.  
***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 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 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 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 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 Color↔TransformationValueSelector and ColorTransformationValue controls.

**NUM\_RGBTRANSFORMLIGHTSOURCE**

### 6.2.2.133 enum spinScan3dCoordinateReferenceSelectorEnums

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

Enumerator

**Scan3dCoordinateReferenceSelector\_RotationX** Rotation around X axis.  
**Scan3dCoordinateReferenceSelector\_RotationY** Rotation around Y axis.  
**Scan3dCoordinateReferenceSelector\_RotationZ** Rotation around Z axis.  
**Scan3dCoordinateReferenceSelector\_TranslationX** X axis translation.  
**Scan3dCoordinateReferenceSelector\_TranslationY** Y axis translation.  
**Scan3dCoordinateReferenceSelector\_TranslationZ** Z axis translation.  
**NUM\_SCAN3DCOORDINATEREFERENCESELECTOR**

### 6.2.2.134 enum spinScan3dCoordinateSelectorEnums

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

Enumerator

**Scan3dCoordinateSelector\_CoordinateA** The first (X or Theta) coordinate  
**Scan3dCoordinateSelector\_CoordinateB** The second (Y or Phi) coordinate  
**Scan3dCoordinateSelector\_CoordinateC** The third (Z or Rho) coordinate.  
**NUM\_SCAN3DCOORDINATESELECTOR**

### 6.2.2.135 enum spinScan3dCoordinateSystemEnums

< Specifies the Coordinate system to use for the device.

Enumerator

**Scan3dCoordinateSystem\_Cartesian** Default value. 3-axis orthogonal, right-hand X-Y-Z.  
**Scan3dCoordinateSystem\_Spherical** A Theta-Phi-Rho coordinate system.  
**Scan3dCoordinateSystem\_Cylindrical** A Theta-Y-Rho coordinate system.  
**NUM\_SCAN3DCOORDINATESYSTEM**

### 6.2.2.136 enum spinScan3dCoordinateSystemReferenceEnums

< Defines coordinate system reference location.

Enumerator

**Scan3dCoordinateSystemReference\_Anchor** Default value. Original fixed reference. The coordinate system fixed relative the camera reference point marker is used.  
**Scan3dCoordinateSystemReference\_Transformed** Transformed reference system. The transformed coordinate system is used according to the definition in the rotation and translation matrices.  
**NUM\_SCAN3DCOORDINATESYSTEMREFERENCE**



## 6.2.2.137 enum spinScan3dCoordinateTransformSelectorEnums

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

## Enumerator

**Scan3dCoordinateTransformSelector\_RotationX** Rotation around X axis.  
**Scan3dCoordinateTransformSelector\_RotationY** Rotation around Y axis.  
**Scan3dCoordinateTransformSelector\_RotationZ** Rotation around Z axis.  
**Scan3dCoordinateTransformSelector\_TranslationX** Translation along X axis.  
**Scan3dCoordinateTransformSelector\_TranslationY** Translation along Y axis.  
**Scan3dCoordinateTransformSelector\_TranslationZ** Translation along Z axis.  
**NUM\_SCAN3DCOORDINATETRANSFORMSELECTOR**

## 6.2.2.138 enum spinScan3dDistanceUnitEnums

< Specifies the unit used when delivering calibrated distance data.

## Enumerator

**Scan3dDistanceUnit\_Millimeter** Distance values are in millimeter units (default).  
**Scan3dDistanceUnit\_Inch** Distance values are in inch units.  
**NUM\_SCAN3DDISTANCEUNIT**

## 6.2.2.139 enum spinScan3dOutputModeEnums

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

## Enumerator

**Scan3dOutputMode\_UncalibratedC** Uncalibrated 2.5D Depth map. The distance data does not represent a physical unit and may be non-linear. The data is a 2.5D range map only.  
**Scan3dOutputMode\_CalibratedABC\_Grid** 3 Coordinates in grid organization. The full 3 coordinate data with the grid array organization from the sensor kept.  
**Scan3dOutputMode\_CalibratedABC\_PointCloud** 3 Coordinates without organization. The full 3 coordinate data without any organization of data points. Typically only valid points transmitted giving varying image size.  
**Scan3dOutputMode\_CalibratedAC** 2 Coordinates with fixed B sampling. The data is sent as a A and C coordinates (X,Z or Theta,Rho). The B (Y) axis uses the scale and offset parameters for the B axis.  
**Scan3dOutputMode\_CalibratedAC\_Linescan** 2 Coordinates with varying sampling. The data is sent as a A and C coordinates (X,Z or Theta,Rho). The B (Y) axis comes from the encoder chunk value.  
**Scan3dOutputMode\_CalibratedC** Calibrated 2.5D Depth map. The distance data is expressed in the chosen distance unit. The data is a 2.5D range map. No information on X-Y axes available.  
**Scan3dOutputMode\_CalibratedC\_Linescan** Depth Map with varying B sampling. The distance data is expressed in the chosen distance unit. The data is a 2.5D range map. The B (Y) axis comes from the encoder chunk value.  
**Scan3dOutputMode\_RectifiedC** Rectified 2.5D Depth map. The distance data has been rectified to a uniform sampling pattern in the X and Y direction. The data is a 2.5D range map only. If a complete 3D point cloud is rectified but transmitted as explicit coordinates it should be transmitted as one of the "CalibratedABC" formats.

**Scan3dOutputMode\_RectifiedC\_Linescan** Rectified 2.5D Depth map with varying B sampling. The data is sent as rectified 1D profiles using Coord3D\_C pixels. The B (Y) axis comes from the encoder chunk value.

**Scan3dOutputMode\_DisparityC** Disparity 2.5D Depth map. The distance is inversely proportional to the pixel (disparity) value.

**Scan3dOutputMode\_DisparityC\_Linescan** Disparity 2.5D Depth map with varying B sampling. The distance is inversely proportional to the pixel (disparity) value. The B (Y) axis comes from the encoder chunk value.

**NUM\_SCAN3DOUTPUTMODE**

#### 6.2.2.140 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 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 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 enum spinSequencerConfigurationModeEnums

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

Enumerator

***SequencerConfigurationMode\_Off***  
***SequencerConfigurationMode\_On***  
***NUM\_SEQUENCERCONFIGURATIONMODE***

## 6.2.2.144 enum spinSequencerConfigurationValidEnums

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

Enumerator

***SequencerConfigurationValid\_No***  
***SequencerConfigurationValid\_Yes***  
***NUM\_SEQUENCERCONFIGURATIONVALID***

## 6.2.2.145 enum spinSequencerModeEnums

< Controls whether or not a sequencer is active.

Enumerator

***SequencerMode\_Off***  
***SequencerMode\_On***  
***NUM\_SEQUENCERMODE***

## 6.2.2.146 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 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 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 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 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 enum spinSerialPortSelectorEnums

< Selects which serial port of the device to control.

Enumerator

***SerialPortSelector\_SerialPort0***  
***NUM\_SERIALPORTSELECTOR***

## 6.2.2.152 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 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 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 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 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 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 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 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 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 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.

**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 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 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 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 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 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 enum spinTransferStatusSelectorEnums

< Selects which status of the transfer module to read.

Enumerator

***TransferStatusSelector\_Streaming*** Data blocks are transmitted when enough data is available.  
***TransferStatusSelector\_Paused*** Data blocks transmission is suspended immediately.  
***TransferStatusSelector\_Stopping*** Data blocks transmission is stopping. The current block transmission will be completed and the transfer state will stop.  
***TransferStatusSelector\_Stopped*** Data blocks transmission is stopped.  
***TransferStatusSelector\_QueueOverflow*** Data blocks queue is in overflow state.  
***NUM\_TRANSFERSTATUSSELECTOR***

### 6.2.2.168 enum spinTransferTriggerActivationEnums

< Specifies the activation mode of the transfer control trigger.

Enumerator

***TransferTriggerActivation\_RisingEdge*** Specifies that the trigger is considered valid on the rising edge of the source signal.  
***TransferTriggerActivation\_FallingEdge*** Specifies that the trigger is considered valid on the falling edge of the source signal.  
***TransferTriggerActivation\_AnyEdge*** Specifies that the trigger is considered valid on the falling or rising edge of the source signal.  
***TransferTriggerActivation\_LevelHigh*** Specifies that the trigger is considered valid as long as the level of the source signal is high. This can apply to TransferActive and TransferPause trigger.  
***TransferTriggerActivation\_LevelLow*** Specifies that the trigger is considered valid as long as the level of the source signal is low. This can apply to TransferActive and TransferPause trigger.  
***NUM\_TRANSFERTRIGGERACTIVATION***

## 6.2.2.169 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 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 enum spinTransferTriggerSourceEnums

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

## Enumerator

**TransferTriggerSource\_Line0** Specifies which physical line (or pin) and associated I/O control block to use as external source for the transfer control trigger signal.

**TransferTriggerSource\_Line1** Specifies which physical line (or pin) and associated I/O control block to use as external source for the transfer control trigger signal.

**TransferTriggerSource\_Line2** Specifies which physical line (or pin) and associated I/O control block to use as external source for the transfer control trigger signal.

**TransferTriggerSource\_Counter0Start** Specifies which of the Counter signal to use as internal source for the transfer control trigger signal.

**TransferTriggerSource\_Counter1Start** Specifies which of the Counter signal to use as internal source for the transfer control trigger signal.

**TransferTriggerSource\_Counter2Start** Specifies which of the Counter signal to use as internal source for the transfer control trigger signal.

**TransferTriggerSource\_Counter0End** Specifies which of the Counter signal to use as internal source for the transfer control trigger signal.

- TransferTriggerSource\_Counter1End*** Specifies which of the Counter signal to use as internal source for the transfer control trigger signal.
- TransferTriggerSource\_Counter2End*** Specifies which of the Counter signal to use as internal source for the transfer control trigger signal.
- TransferTriggerSource\_Timer0Start*** Specifies which Timer signal to use as internal source for the transfer control trigger signal.
- 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 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 enum spinTriggerModeEnums

< Controls whether or not trigger is active.

Enumerator

***TriggerMode\_Off***  
***TriggerMode\_On***  
***NUM\_TRIGGERMODE***

## 6.2.2.174 enum spinTriggerOverlapEnums

< Specifies the overlap mode of the trigger.

Enumerator

***TriggerOverlap\_Off***  
***TriggerOverlap\_ReadOut***  
***TriggerOverlap\_PreviousFrame***  
***NUM\_TRIGGEROVERLAP***

## 6.2.2.175 enum spinTriggerSelectorEnums

< Selects the type of trigger to configure.

Enumerator

***TriggerSelector\_AcquisitionStart***  
***TriggerSelector\_FrameStart***  
***TriggerSelector\_FrameBurstStart***  
***NUM\_TRIGGERSELECTOR***

## 6.2.2.176 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 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 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 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 enum spinWhiteClipSelectorEnums

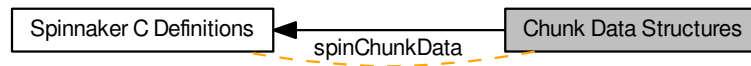
< Selects which White Clip to control.

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

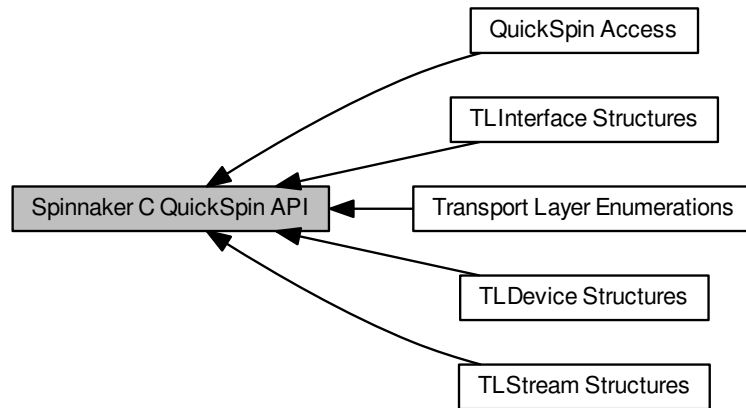
- struct [spinChunkData](#)

*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](#)

### 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)

### 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** [SPINNAKERC\\_API quickSpinInit](#) ( [spinCamera](#) hCamera, [quickSpin](#) \* pQuickSpin )

**6.5.2.2** [SPINNAKERC\\_API quickSpinInitEx](#) ( [spinCamera](#) hCamera, [quickSpin](#) \* pQuickSpin, [quickSpinTLDevice](#) \* pQuickSpinTLDevice, [quickSpinTLStream](#) \* pQuickSpinTLStream )

**6.5.2.3** [SPINNAKERC\\_API quickSpinTLDeviceInit](#) ( [spinCamera](#) hCamera, [quickSpinTLDevice](#) \* pQuickSpinTLDevice )

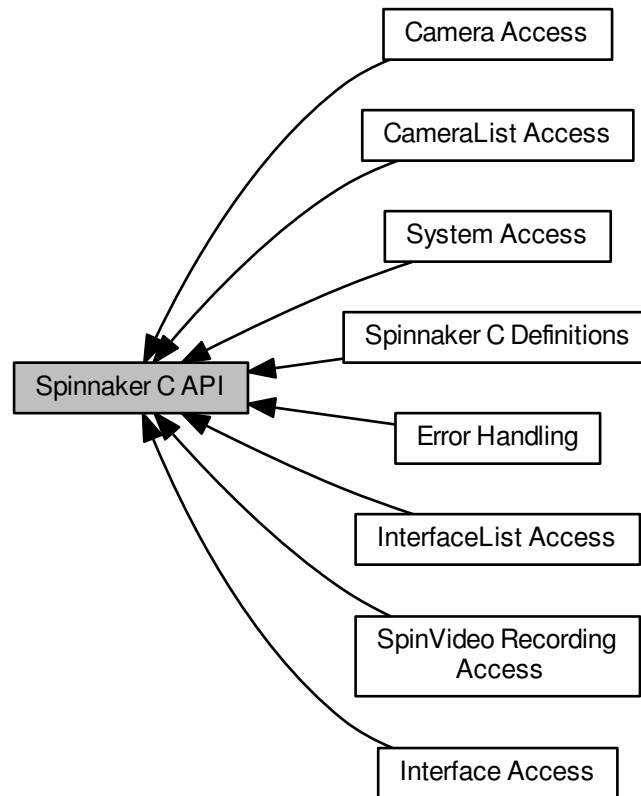
**6.5.2.4** [SPINNAKERC\\_API quickSpinTLInterfaceInit](#) ( [spinInterface](#) hInterface, [quickSpinTLInterface](#) \* pQuickSpinTLInterface )

**6.5.2.5** [SPINNAKERC\\_API quickSpinTLStreamInit](#) ( [spinCamera](#) hCamera, [quickSpinTLStream](#) \* pQuickSpinTLStream )

## 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 [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 |

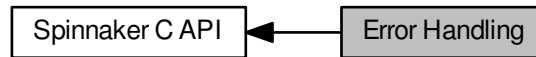
#### Returns

[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 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 SPINNAKER\_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 SPINNAKER\_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 SPINNAKER\_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 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 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 SPINNAKER\_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 SPINNAKER\_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.*

### 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 [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 **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 **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 **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 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 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 SPINNAKERC\_API spinSystemIsInUse ( spinSystem *hSystem*, bool8\_t \* *pIsInUse* )**

Checks whether a system is currently in use.

See also

[spinError](#)

## Parameters

|                  |                                                                      |
|------------------|----------------------------------------------------------------------|
| <i>hSystem</i>   | The system to check                                                  |
| <i>pblsInUse</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

**6.8.2.8 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

**6.8.2.9 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

**6.8.2.10 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

**6.8.2.11 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

**6.8.2.12 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

**6.8.2.13 SPINNAKER\_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>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

**6.8.2.14 SPINNAKER\_API spinSystemSetLogLevel ( 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

**6.8.2.15 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

**6.8.2.16 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

**6.8.2.17 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

**6.8.2.18 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

**6.8.2.19 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



**6.8.2.20 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

**6.8.2.21 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 |

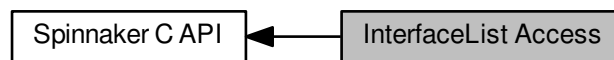
**Returns**

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](#) \*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.*

### 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 SPINNAKERC\_API spinInterfaceListClear ( [spinInterfaceList](#) hInterfaceList )

Clears an interface list.

See also

[spinError](#)

#### Parameters

|                                |                             |
|--------------------------------|-----------------------------|
| <a href="#">hInterfaceList</a> | The interface list to clear |
|--------------------------------|-----------------------------|

**Returns**

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

**6.9.2.2 SPINNAKER\_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 SPINNAKER\_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 SPINNAKER\_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 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

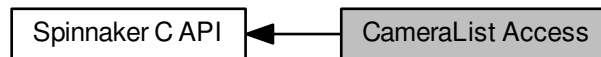
## See also

[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 [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 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 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 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 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 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 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 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 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 |

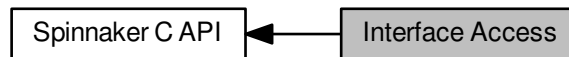
## Returns

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 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 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 **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 **SPINNAKERC\_API** `spinInterfaceIsInUse ( spinInterface hInterface, bool8_t * pblsInUse )`

Checks whether an interface is in use.

**See also**

[spinError](#)

**Parameters**

|                   |                                                                      |
|-------------------|----------------------------------------------------------------------|
| <i>hInterface</i> | The interface to check                                               |
| <i>pblsInUse</i>  | The boolean pointer to return whether or not the interface is in use |

**Returns**

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

#### 6.11.2.5 **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 **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 **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 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 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 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 **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 **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 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**

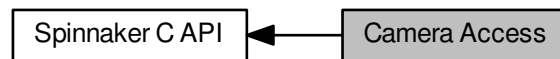
`spinError` The error code; returns `SPINNAKER_ERR_SUCCESS` (or 0) for no error



## 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 spinCamerasIsInitialized](#) ([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 [SPINNAKERC\\_API spinCameraBeginAcquisition](#) ( [spinCamera](#) hCamera )

Has a camera start acquiring images.

See also

[spinError](#)

Parameters

|                         |                                      |
|-------------------------|--------------------------------------|
| <a href="#">hCamera</a> | The camera to begin acquiring images |
|-------------------------|--------------------------------------|

Returns

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

#### 6.12.2.2 [SPINNAKERC\\_API spinCameraDelInit](#) ( [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 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 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 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 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 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 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 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 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 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 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 SPINNAKERC\_API spinCamerasInitialized ( 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 SPINNAKERC\_API spinCamerasStreaming ( spinCamera *hCamera*, bool8\_t \* *pblsStreaming* )**

Checks whether a camera is currently acquiring images.

## See also

[spinError](#)

## Parameters

|                      |                                                                                |
|----------------------|--------------------------------------------------------------------------------|
| <i>hCamera</i>       | The camera to check                                                            |
| <i>pblsStreaming</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 SPINNAKERC\_API spinCamerasValid ( 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 **SPINNAKERC\_API** *spinCameraReadPort* ( *spinCamera hCamera*, *uint64\_t iAddress*, *void \* pBuffer*, *size\_t iSize* )

6.12.2.17 **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 **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 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 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 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 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 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 [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 **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 **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 **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 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 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 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 SPINNAKERC\_API spinImageDeepCopy ( spinImage *hSrcImage*, spinImage *hDestImage* )

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

See also

[spinError](#)

##### Parameters

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

##### Returns

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

#### 6.13.2.9 SPINNAKERC\_API spinImageDestroy ( spinImage *hImage* )

Destroys an image.

See also

[spinError](#)



## Parameters

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

## Returns

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

#### 6.13.2.10 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 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 is returned |

## Returns

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

#### 6.13.2.12 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 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 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 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 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 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 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 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 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 SPINNAKER\_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 SPINNAKER\_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 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 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 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 SPINNAKER\_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 SPINNAKER\_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 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 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 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 SPINNAKER\_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 SPINNAKER\_API spinImageGetTLPayloadType ( spinImage hImage, spinPayloadTypeInfolDs \* 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 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 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 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 SPINNAKER\_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 SPINNAKER\_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 SPINNAKER\_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 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 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 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 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 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 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 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 SPINNAKER\_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 SPINNAKER\_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 SPINNAKER\_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 SPINNAKER\_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 SPINNAKER\_API spinImageSaveTiff ( spinImage *hImage*, const char \* *pFilename*, const spinTIFFOption \* *pOption* )**

Saves an image as a TIFF image.



See also

[spinError](#)

#### Parameters

|                  |                                                                                        |
|------------------|----------------------------------------------------------------------------------------|
| <i>hImage</i>    | The image to be saved                                                                  |
| <i>pFilename</i> | The filename to use to save the image (with or without the appropriate file extension) |
| <i>pOption</i>   | The image options related to saving as TIFF; includes compression method               |

#### Returns

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

#### 6.13.2.51 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 [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:<br>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 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 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:<br>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 SPINNAKER\_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 SPINNAKER\_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:<br>void(<em>spinImageEventFunction)(const spinImage hImage, void pUserData) |
| <i>pUserData</i>    | Properties that can be passed into the event function                                                                                                 |

**Returns**

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

**6.14.2.6 SPINNAKER\_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 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 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 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:<br>void(<em>spinLogEventFunction)(const spinLogEventData hEventData, void pUserData) |
| <i>pUserData</i>  | Properties that can be passed into the event function                                                                                                           |

##### Returns

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

#### 6.14.2.10 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 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:<br>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 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 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 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 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 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 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 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 SPINNAKER\_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 SPINNAKER\_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>ppHistogram</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 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 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>ppHistogram</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 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 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 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 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 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 [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 **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 **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 **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 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 SPINNAKERC\_API spinLogDataGetThreadName ( spinLogEventData *hLogEventData*, char \* *pBuf*, size\_t \* *pBufLen* )

Retrieves the thread name of a log event.

See also

[spinError](#)

#### Parameters

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

**Returns**

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

**6.16.2.7 SPINNAKERC\_API spinLogDataGetTimestamp ( spinLogEventData hLogEventData, char \* pBuf, size\_t \* pBufLen )**

Retrieves the timestamp of a log event.

**See also**

[spinError](#)

**Parameters**

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

**Returns**

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

## 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 [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 [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 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 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](#) ( "spinAVIRecorderOpenUncompressed is *deprecated*, use [spinVideoOpenUncompressed](#) instead." , spinAVIRecorderOpenUncompressed([spinAVIRecorder](#) \*phRecorder, const char \*pName, [spinAVIOption](#) option) )
- 6.18.2.2 [SPINNAKERC\\_API\\_DEPRECATED](#) ( "spinAVIRecorderOpenMJPEG is *deprecated*, use [spinVideoOpenMJPEG](#) instead." , spinAVIRecorderOpenMJPEG([spinAVIRecorder](#) \*phRecorder, const char \*pName, [spinMJPEGOption](#) option) )
- 6.18.2.3 [SPINNAKERC\\_API\\_DEPRECATED](#) ( "spinAVIRecorderOpenH264 is *deprecated*, use [spinVideoOpenH264](#) instead." , spinAVIRecorderOpenH264([spinAVIRecorder](#) \*phRecorder, const char \*pName, [spinH264Option](#) option) )
- 6.18.2.4 [SPINNAKERC\\_API\\_DEPRECATED](#) ( "spinAVIRecorderAppend is *deprecated*, use [spinVideoAppend](#) instead." , spinAVIRecorderAppend([spinAVIRecorder](#) hRecorder, [spinImage](#) hImage) )
- 6.18.2.5 [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 ( "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 SPINNAKERC\_API spinImageChunkDataGetFloatValue** ( [spinImage](#) *hImage*, const char \* *pName*, double \* *pValue* )

**6.19.2.2 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 `typedef void* spinArrivalEvent`

Handle for arrival event functionality.

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

#### 6.20.2.2 `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 `typedef void* spinCamera`

Handle for camera functionality.

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

#### 6.20.2.4 `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 `typedef void* spinDeviceEvent`

Handle for device event functionality.

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

#### 6.20.2.6 `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 `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 `typedef void* spinImageEvent`

Handle for image event functionality.

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

#### 6.20.2.9 `typedef void* spinImageStatistics`

Handle for image statistics functionality.

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

#### 6.20.2.10 `typedef void* spinInterface`

Handle for interface functionality.

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

#### 6.20.2.11 `typedef void* spinInterfaceEvent`

Handle for interface event functionality.

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

#### 6.20.2.12 `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 `typedef void* spinLogEvent`

Handle for logging event functionality.

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

#### 6.20.2.14 `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 `typedef void* spinRemovalEvent`

Handle for removal event functionality.

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

#### 6.20.2.16 `typedef void* spinSystem`

Handle for system functionality.

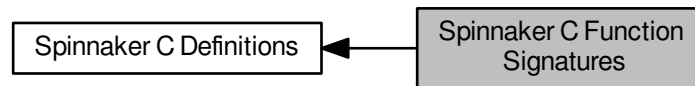
Created by calling [spinSystemGetInstance\(\)](#), which requires a call to [spinSystemReleaseInstance\(\)](#) to release.

#### 6.20.2.17 `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 typedef void(\* [spinArrivalEventFunction](#)) (uint64\_t deviceSerialNumber, void \*pUserData)

6.21.2.2 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 typedef void(\* [spinImageEventFunction](#)) (const [spinImage](#) hImage, void \*pUserData)

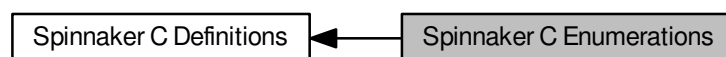
6.21.2.4 typedef void(\* [spinLogEventFunction](#)) (const [spinLogEventData](#) hEventData, void \*pUserData)

6.21.2.5 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`,
  - `EDGE_SENSING`,
  - `HQ_LINEAR`,
  - `RIGOROUS`,
  - `IPP`,
  - `DIRECTIONAL_FILTER`,
  - `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 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.

**EDGE\_SENSING** Weights surrounding pixels based on localized edge orientation.

**HQ\_LINEAR** Well-balanced speed and quality.

**RIGOROUS** Slowest but produces good results.

**IPP** Multithreaded with similar results to edge sensing.

**DIRECTIONAL\_FILTER** Best quality but much faster than rigorous.

**WEIGHTED\_DIRECTIONAL\_FILTER** Weighted pixel average from different directions.

### 6.22.2.2 enum spinError

The error codes used in Spinnaker C.

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

#### Enumerator

**SPINNAKER\_ERR\_SUCCESS** An error code of 0 means that the function has run without error.

**SPINNAKER\_ERR\_ERROR** The error codes in the range of -1000 to -1999 are reserved for Spinnaker exceptions.

**SPINNAKER\_ERR\_NOT\_INITIALIZED**

**SPINNAKER\_ERR\_NOT\_IMPLEMENTED**

**SPINNAKER\_ERR\_RESOURCE\_IN\_USE**

**SPINNAKER\_ERR\_ACCESS\_DENIED**

**SPINNAKER\_ERR\_INVALID\_HANDLE**

**SPINNAKER\_ERR\_INVALID\_ID**

**SPINNAKER\_ERR\_NO\_DATA**

**SPINNAKER\_ERR\_INVALID\_PARAMETER**

**SPINNAKER\_ERR\_IO**

**SPINNAKER\_ERR\_TIMEOUT**

**SPINNAKER\_ERR\_ABORT**

**SPINNAKER\_ERR\_INVALID\_BUFFER**

**SPINNAKER\_ERR\_NOT\_AVAILABLE**

**SPINNAKER\_ERR\_INVALID\_ADDRESS**

**SPINNAKER\_ERR\_BUFFER\_TOO\_SMALL**

**SPINNAKER\_ERR\_INVALID\_INDEX**

**SPINNAKER\_ERR\_PARSING\_CHUNK\_DATA**

**SPINNAKER\_ERR\_INVALID\_VALUE**

**SPINNAKER\_ERR\_RESOURCE\_EXHAUSTED**

**SPINNAKER\_ERR\_OUT\_OF\_MEMORY**

**SPINNAKER\_ERR\_BUSY**

**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 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 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 [GetNextImage\(\)](#) 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 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 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 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 enum spinStatisticsChannel

Channels that allow statistics to be calculated.

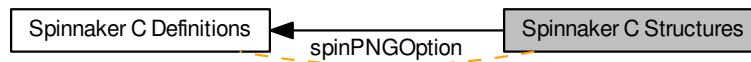
Enumerator

***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 [spinMJPEGOption](#)  
*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 enum [actionCommandStatus](#)

Possible Status Codes Returned from Action Command.

Enumerator

***ACTION\_COMMAND\_STATUS\_OK*** The device acknowledged the command.

***ACTION\_COMMAND\_STATUS\_NO\_REF\_TIME***

***ACTION\_COMMAND\_STATUS\_OVERFLOW***

***ACTION\_COMMAND\_STATUS\_ACTION\_LATE***

***ACTION\_COMMAND\_STATUS\_ERROR***

#### 6.23.2.2 enum [spinCompressionMethod](#)

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

Enumerator

***NONE***

***PACKBITS***

***DEFLATE***

***ADOBE\_DEFLATE***

***CCITTFAX3***

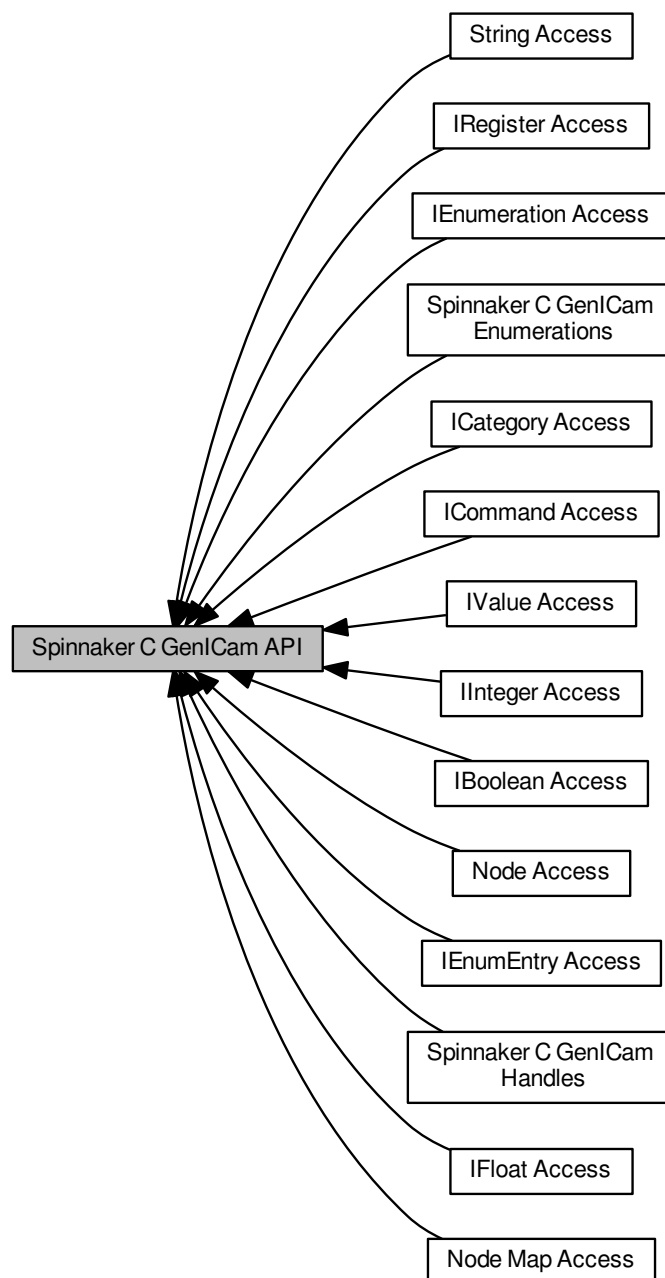
***CCITTFAX4***

***LZW***

***JPG***

## 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](#) \*phNode)  
*Retrieves a node from the nodemap by name.*
- [SPINNAKERC\\_API spinNodeMapGetNumNodes](#) ([spinNodeMapHandle](#) hNodeMap, size\_t \*pValue)  
*Gets the number of nodes in the map.*
- [SPINNAKERC\\_API spinNodeMapGetNodeByIndex](#) ([spinNodeMapHandle](#) hNodeMap, size\_t index, [spinNodeHandle](#) \*phNode)  
*Retrieves a node from the nodemap by index.*
- [SPINNAKERC\\_API 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 SPINNAKERC\_API spinNodeMapGetNode ( [spinNodeMapHandle](#) hNodeMap, const char \* pName, [spinNodeHandle](#) \* phNode )

Retrieves a node from the nodemap by name.

See also

[spinError](#)

#### Parameters

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

**Returns**

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

**6.25.2.2 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 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 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 [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 [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 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 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 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 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 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 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 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 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 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 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 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 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 SPINNAKERC\_API spinNodesAvailable ( 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 SPINNAKERC\_API spinNodesEqual ( 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 SPINNAKERC\_API spinNodesImplemented ( 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 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 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 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                                                                       |

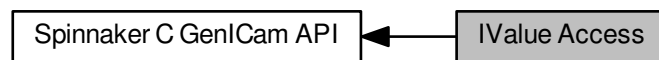
**Returns**

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 [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 SPINNAKER\_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 SPINNAKER\_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 **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 |

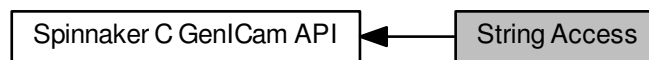
#### Returns

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 [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 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 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 SPINNAKER\_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 SPINNAKER\_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 [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 SPINNAKER\_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 SPINNAKER\_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 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 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 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 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 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 [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 SPINNAKER\_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 SPINNAKER\_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 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 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 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 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 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 [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 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 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 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 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 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](#).

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 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.*

#### 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 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](#).

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 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 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 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 SPINNAKER\_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                  |

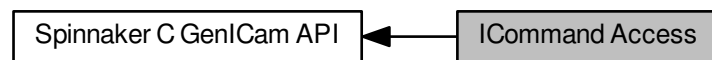
**Returns**

`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 SPINNAKERC\_API spinCommandExecute ( [spinNodeHandle](#) hNode )

Executes the action associated to a command node.

See also

[spinError](#)

Parameters

|                       |                             |
|-----------------------|-----------------------------|
| <a href="#">hNode</a> | The command node to execute |
|-----------------------|-----------------------------|

Returns

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

#### 6.34.2.2 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 [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 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 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 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 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 SPINNAKER\_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 SPINNAKER\_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 SPINNAKER\_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 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                          |

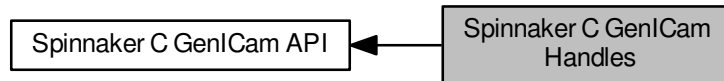
**Returns**

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 typedef void(\* [spinNodeCallbackFunction](#)) ([spinNodeHandle](#) hNode)

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

#### 6.37.2.2 typedef void\* [spinNodeCallbackHandle](#)

Handle for callback functionality.

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

#### 6.37.2.3 typedef void\* [spinNodeHandle](#)

Handle for node functionality.

Created by calling [spinNodeMapGetNode\(\)](#). No need to release, clear, or destroy.

#### 6.37.2.4 typedef void\* [spinNodeMapHandle](#)

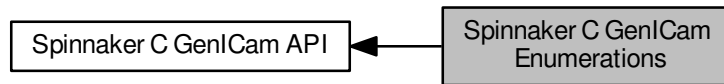
Handle for nodemap functionality.

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 enum `spinAccessMode`

Enumerator

***NI***  
***NA***  
***WO***  
***RO***  
***RW***  
***\_UndefinedAccesMode***  
***\_CycleDetectAccesMode***



## 6.38.2.2 enum spinCachingMode

Enumerator

**NoCache**  
**WriteThrough**  
**WriteAround**  
**\_UndefinedCachingMode**

## 6.38.2.3 enum spinDisplayNotation

typedef for float notation

Enumerator

**fnAutomatic**  
**fnFixed**      the notation if either scientific or fixed depending on what is shorter  
**fnScientific**      the notation is fixed, e.g. 123.4  
**\_UndefinedEDisplayNotation**      the notation is scientific, e.g. 1.234e2  
Object is not yet initialized

## 6.38.2.4 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 enum spinIncMode

typedef for increment mode

Enumerator

**noIncrement**  
**fixedIncrement**  
**listIncrement**

### 6.38.2.6 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 enum spinInterfaceType

typedef for interface type

Enumerator

**intflValue**

**intflBase** IValue interface

**intflInteger** IBase interface

**intflBoolean** IInteger interface

**intflCommand** IBoolean interface

**intflFloat** ICommand interface

**intflString** IFloat interface

**intflRegister** IString interface

**intflCategory** IRegister interface

**intflEnumeration** ICategory interface

**intflEnumEntry** IEnumeration interface

**intflPort** IEnumEntry interface

IPort interface

### 6.38.2.8 enum spinLinkType

typedef for link type

Enumerator

**ctAllDependingNodes**

**ctAllTerminalNodes** All nodes which will be invalidated if this node becomes invalid

**ctInvalidators** All terminal nodes which may be written to by this node

**ctReadingChildren** List of references to nodes which may invalidate this node

**ctWritingChildren** All child nodes which influence this node's AccessMode

**ctDependingChildren** All child nodes which may be written to

All child nodes which will cause this node to be invalidated

## 6.38.2.9 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 enum spinNodeType

Enumerator

- ValueNode**
- BaseNode**
- IntegerNode**
- BooleanNode**
- FloatNode**
- CommandNode**
- StringNode**
- RegisterNode**
- EnumerationNode**
- EnumEntryNode**
- CategoryNode**
- PortNode**
- UnknownNode**

## 6.38.2.11 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 enum spinSign

Enumerator

**Signed**  
**Unsigned**  
**\_UndefinedSign**

## 6.38.2.13 enum spinSlope

typedef for fomula type

Enumerator

**Increasing**  
**Decreasing** strictly monotonous increasing  
**Varying** strictly monotonous decreasing  
**Automatic** slope changes, e.g. at run-time  
**\_UndefinedESlope** slope is determined automatically by probing the function  
Object is not yet initialized

## 6.38.2.14 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 enum spinVisibility

Enumerator

**Beginner**  
**Expert**  
**Guru**  
**Invisible**  
**\_UndefinedVisibility**

## 6.38.2.16 enum spinXMLValidation

typedef describing the different validity checks which can be performed on an XML file

The enum values for a bitfield of length uint32\_t

Enumerator

**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 enum spinYesNo

Defines the choices of a Yes/No alternative.

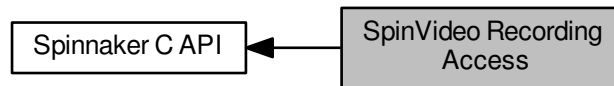
Enumerator

**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, [spinAVIOption](#) option)
- [SPINNAKERC\\_API spinVideoOpenMJPEG](#) ([spinVideo](#) \*phSpinVideo, const char \*pName, [spinMJPEGOption](#) option)
- [SPINNAKERC\\_API spinVideoOpenH264](#) ([spinVideo](#) \*phSpinVideo, const char \*pName, [spinH264Option](#) option)
- [SPINNAKERC\\_API spinVideoAppend](#) ([spinVideo](#) hSpinVideo, [spinImage](#) hImage)
- [SPINNAKERC\\_API spinVideoSetMaximumFileSize](#) ([spinVideo](#) hSpinVideo, unsigned int size)  
*Set the maximum file size (in megabytes) of a AVI/MP4 file.*
- [SPINNAKERC\\_API spinVideoClose](#) ([spinVideo](#) hSpinVideo)

### 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 **SPINNAKERC\_API spinVideoAppend** ( [spinVideo](#) hSpinVideo, [spinImage](#) hImage )

6.39.2.2 **SPINNAKERC\_API spinVideoClose** ( [spinVideo](#) hSpinVideo )

6.39.2.3 **SPINNAKERC\_API spinVideoOpenH264** ( [spinVideo](#) \* phSpinVideo, const char \* pName, [spinH264Option](#) option )

6.39.2.4 **SPINNAKERC\_API spinVideoOpenMJPEG** ( [spinVideo](#) \* phSpinVideo, const char \* pName, [spinMJPEGOption](#) option )

6.39.2.5 **SPINNAKERC\_API spinVideoOpenUncompressed** ( [spinVideo](#) \* phSpinVideo, const char \* pName, [spinAVIOption](#) option )

6.39.2.6 **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` }

### 6.40.1 Detailed Description

### 6.40.2 Enumeration Type Documentation

#### 6.40.2.1 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 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 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 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 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**

## 6.40.2.6 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**

## 6.40.2.7 enum spinTLGUIXMLLocationEnums

< Sets the location to load GUI XML.

## Enumerator

**GUIXMLLocation\_Device** Load XML from device

**GUIXMLLocation\_Host** Load XML from host

**NUMGUIXMLLOCATION**

## 6.40.2.8 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**

#### 6.40.2.9 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**

#### 6.40.2.10 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**

#### 6.40.2.11 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**

## 6.40.2.12 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:



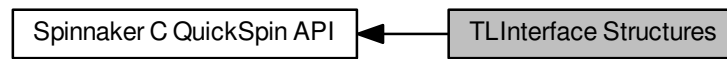
### Data Structures

- 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



## 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 unsigned int DeviceAddress

##### 7.1.2.2 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 [quickSpinBooleanNode AasRoiEnable](#)

7.2.1.2 [quickSpinIntegerNode AasRoiHeight](#)

7.2.1.3 [quickSpinIntegerNode AasRoiOffsetX](#)

7.2.1.4 [quickSpinIntegerNode AasRoiOffsetY](#)

7.2.1.5 [quickSpinIntegerNode AasRoiWidth](#)

7.2.1.6 [quickSpinCommandNode AcquisitionAbort](#)

7.2.1.7 [quickSpinCommandNode AcquisitionArm](#)

7.2.1.8 [quickSpinIntegerNode AcquisitionBurstFrameCount](#)

7.2.1.9 [quickSpinIntegerNode AcquisitionFrameCount](#)

7.2.1.10 [quickSpinFloatNode AcquisitionFrameRate](#)

7.2.1.11 [quickSpinBooleanNode AcquisitionFrameRateEnable](#)

7.2.1.12 [quickSpinFloatNode AcquisitionLineRate](#)

7.2.1.13 [quickSpinEnumerationNode AcquisitionMode](#)

7.2.1.14 [quickSpinFloatNode AcquisitionResultingFrameRate](#)

7.2.1.15 [quickSpinCommandNode AcquisitionStart](#)

- 7.2.1.16 quickSpinBooleanNode AcquisitionStatus
- 7.2.1.17 quickSpinEnumerationNode AcquisitionStatusSelector
- 7.2.1.18 quickSpinCommandNode AcquisitionStop
- 7.2.1.19 quickSpinIntegerNode ActionDeviceKey
- 7.2.1.20 quickSpinIntegerNode ActionGroupKey
- 7.2.1.21 quickSpinIntegerNode ActionGroupMask
- 7.2.1.22 quickSpinIntegerNode ActionQueueSize
- 7.2.1.23 quickSpinIntegerNode ActionSelector
- 7.2.1.24 quickSpinEnumerationNode ActionUnconditionalMode
- 7.2.1.25 quickSpinBooleanNode AdaptiveCompressionEnable
- 7.2.1.26 quickSpinEnumerationNode AdcBitDepth
- 7.2.1.27 quickSpinIntegerNode aPAUSEMACCtrlFramesReceived
- 7.2.1.28 quickSpinIntegerNode aPAUSEMACCtrlFramesTransmitted
- 7.2.1.29 quickSpinEnumerationNode AutoAlgorithmSelector
- 7.2.1.30 quickSpinFloatNode AutoExposureControlLoopDamping
- 7.2.1.31 quickSpinEnumerationNode AutoExposureControlPriority
- 7.2.1.32 quickSpinFloatNode AutoExposureEVCompensation
- 7.2.1.33 quickSpinFloatNode AutoExposureExposureTimeLowerLimit
- 7.2.1.34 quickSpinFloatNode AutoExposureExposureTimeUpperLimit
- 7.2.1.35 quickSpinFloatNode AutoExposureGainLowerLimit
- 7.2.1.36 quickSpinFloatNode AutoExposureGainUpperLimit
- 7.2.1.37 quickSpinFloatNode AutoExposureGreyValueLowerLimit
- 7.2.1.38 quickSpinFloatNode AutoExposureGreyValueUpperLimit

- 7.2.1.39 `quickSpinEnumerationNode` `AutoExposureLightingMode`
- 7.2.1.40 `quickSpinEnumerationNode` `AutoExposureMeteringMode`
- 7.2.1.41 `quickSpinFloatNode` `AutoExposureTargetGreyValue`
- 7.2.1.42 `quickSpinEnumerationNode` `AutoExposureTargetGreyValueAuto`
- 7.2.1.43 `quickSpinFloatNode` `BalanceRatio`
- 7.2.1.44 `quickSpinEnumerationNode` `BalanceRatioSelector`
- 7.2.1.45 `quickSpinEnumerationNode` `BalanceWhiteAuto`
- 7.2.1.46 `quickSpinFloatNode` `BalanceWhiteAutoDamping`
- 7.2.1.47 `quickSpinFloatNode` `BalanceWhiteAutoLowerLimit`
- 7.2.1.48 `quickSpinEnumerationNode` `BalanceWhiteAutoProfile`
- 7.2.1.49 `quickSpinFloatNode` `BalanceWhiteAutoUpperLimit`
- 7.2.1.50 `quickSpinIntegerNode` `BinningHorizontal`
- 7.2.1.51 `quickSpinEnumerationNode` `BinningHorizontalMode`
- 7.2.1.52 `quickSpinEnumerationNode` `BinningSelector`
- 7.2.1.53 `quickSpinIntegerNode` `BinningVertical`
- 7.2.1.54 `quickSpinEnumerationNode` `BinningVerticalMode`
- 7.2.1.55 `quickSpinFloatNode` `BlackLevel`
- 7.2.1.56 `quickSpinEnumerationNode` `BlackLevelAuto`
- 7.2.1.57 `quickSpinEnumerationNode` `BlackLevelAutoBalance`
- 7.2.1.58 `quickSpinBooleanNode` `BlackLevelClampingEnable`
- 7.2.1.59 `quickSpinIntegerNode` `BlackLevelRaw`
- 7.2.1.60 `quickSpinEnumerationNode` `BlackLevelSelector`
- 7.2.1.61 `quickSpinFloatNode` `ChunkBlackLevel`



- 7.2.1.62 quickSpinEnumerationNode ChunkBlackLevelSelector
- 7.2.1.63 quickSpinEnumerationNode ChunkCounterSelector
- 7.2.1.64 quickSpinIntegerNode ChunkCounterValue
- 7.2.1.65 quickSpinIntegerNode ChunkCRC
- 7.2.1.66 quickSpinBooleanNode ChunkEnable
- 7.2.1.67 quickSpinEnumerationNode ChunkEncoderSelector
- 7.2.1.68 quickSpinEnumerationNode ChunkEncoderStatus
- 7.2.1.69 quickSpinIntegerNode ChunkEncoderValue
- 7.2.1.70 quickSpinIntegerNode ChunkExposureEndLineStatusAll
- 7.2.1.71 quickSpinFloatNode ChunkExposureTime
- 7.2.1.72 quickSpinEnumerationNode ChunkExposureTimeSelector
- 7.2.1.73 quickSpinIntegerNode ChunkFrameID
- 7.2.1.74 quickSpinFloatNode ChunkGain
- 7.2.1.75 quickSpinEnumerationNode ChunkGainSelector
- 7.2.1.76 quickSpinIntegerNode ChunkHeight
- 7.2.1.77 quickSpinIntegerNode ChunkImage
- 7.2.1.78 quickSpinEnumerationNode ChunkImageComponent
- 7.2.1.79 quickSpinFloatNode ChunkInferenceConfidence
- 7.2.1.80 quickSpinIntegerNode ChunkInferenceResult
- 7.2.1.81 quickSpinIntegerNode ChunkLinePitch
- 7.2.1.82 quickSpinIntegerNode ChunkLineStatusAll
- 7.2.1.83 quickSpinBooleanNode ChunkModeActive
- 7.2.1.84 quickSpinIntegerNode ChunkOffsetX

- 7.2.1.85 `quickSpinIntegerNode ChunkOffsetY`
- 7.2.1.86 `quickSpinIntegerNode ChunkPartSelector`
- 7.2.1.87 `quickSpinIntegerNode ChunkPixelDynamicRangeMax`
- 7.2.1.88 `quickSpinIntegerNode ChunkPixelDynamicRangeMin`
- 7.2.1.89 `quickSpinEnumerationNode ChunkPixelFormat`
- 7.2.1.90 `quickSpinEnumerationNode ChunkRegionID`
- 7.2.1.91 `quickSpinFloatNode ChunkScan3dAxisMax`
- 7.2.1.92 `quickSpinFloatNode ChunkScan3dAxisMin`
- 7.2.1.93 `quickSpinFloatNode ChunkScan3dCoordinateOffset`
- 7.2.1.94 `quickSpinEnumerationNode ChunkScan3dCoordinateReferenceSelector`
- 7.2.1.95 `quickSpinFloatNode ChunkScan3dCoordinateReferenceValue`
- 7.2.1.96 `quickSpinFloatNode ChunkScan3dCoordinateScale`
- 7.2.1.97 `quickSpinEnumerationNode ChunkScan3dCoordinateSelector`
- 7.2.1.98 `quickSpinEnumerationNode ChunkScan3dCoordinateSystem`
- 7.2.1.99 `quickSpinEnumerationNode ChunkScan3dCoordinateSystemReference`
- 7.2.1.100 `quickSpinEnumerationNode ChunkScan3dCoordinateTransformSelector`
- 7.2.1.101 `quickSpinEnumerationNode ChunkScan3dDistanceUnit`
- 7.2.1.102 `quickSpinBooleanNode ChunkScan3dInvalidDataFlag`
- 7.2.1.103 `quickSpinFloatNode ChunkScan3dInvalidDataValue`
- 7.2.1.104 `quickSpinEnumerationNode ChunkScan3dOutputMode`
- 7.2.1.105 `quickSpinFloatNode ChunkScan3dTransformValue`
- 7.2.1.106 `quickSpinIntegerNode ChunkScanLineSelector`
- 7.2.1.107 `quickSpinEnumerationNode ChunkSelector`

- 7.2.1.108 quickSpinIntegerNode ChunkSequencerSetActive
- 7.2.1.109 quickSpinStringNode ChunkSerialData
- 7.2.1.110 quickSpinIntegerNode ChunkSerialDataLength
- 7.2.1.111 quickSpinBooleanNode ChunkSerialReceiveOverflow
- 7.2.1.112 quickSpinEnumerationNode ChunkSourceID
- 7.2.1.113 quickSpinIntegerNode ChunkStreamChannelID
- 7.2.1.114 quickSpinEnumerationNode ChunkTimerSelector
- 7.2.1.115 quickSpinFloatNode ChunkTimerValue
- 7.2.1.116 quickSpinIntegerNode ChunkTimestamp
- 7.2.1.117 quickSpinIntegerNode ChunkTimestampLatchValue
- 7.2.1.118 quickSpinIntegerNode ChunkTransferBlockID
- 7.2.1.119 quickSpinIntegerNode ChunkTransferQueueCurrentBlockCount
- 7.2.1.120 quickSpinEnumerationNode ChunkTransferStreamID
- 7.2.1.121 quickSpinIntegerNode ChunkWidth
- 7.2.1.122 quickSpinEnumerationNode CIConfiguration
- 7.2.1.123 quickSpinEnumerationNode CTimeSlotsCount
- 7.2.1.124 quickSpinBooleanNode ColorTransformationEnable
- 7.2.1.125 quickSpinEnumerationNode ColorTransformationSelector
- 7.2.1.126 quickSpinFloatNode ColorTransformationValue
- 7.2.1.127 quickSpinEnumerationNode ColorTransformationValueSelector
- 7.2.1.128 quickSpinFloatNode CompressionRatio
- 7.2.1.129 quickSpinIntegerNode CounterDelay
- 7.2.1.130 quickSpinIntegerNode CounterDuration

- 7.2.1.131 quickSpinEnumerationNode CounterEventActivation
- 7.2.1.132 quickSpinEnumerationNode CounterEventSource
- 7.2.1.133 quickSpinCommandNode CounterReset
- 7.2.1.134 quickSpinEnumerationNode CounterResetActivation
- 7.2.1.135 quickSpinEnumerationNode CounterResetSource
- 7.2.1.136 quickSpinEnumerationNode CounterSelector
- 7.2.1.137 quickSpinEnumerationNode CounterStatus
- 7.2.1.138 quickSpinEnumerationNode CounterTriggerActivation
- 7.2.1.139 quickSpinEnumerationNode CounterTriggerSource
- 7.2.1.140 quickSpinIntegerNode CounterValue
- 7.2.1.141 quickSpinIntegerNode CounterValueAtReset
- 7.2.1.142 quickSpinIntegerNode CxpConnectionSelector
- 7.2.1.143 quickSpinIntegerNode CxpConnectionTestErrorCount
- 7.2.1.144 quickSpinEnumerationNode CxpConnectionTestMode
- 7.2.1.145 quickSpinIntegerNode CxpConnectionTestPacketCount
- 7.2.1.146 quickSpinEnumerationNode CxpLinkConfiguration
- 7.2.1.147 quickSpinEnumerationNode CxpLinkConfigurationPreferred
- 7.2.1.148 quickSpinEnumerationNode CxpLinkConfigurationStatus
- 7.2.1.149 quickSpinCommandNode CxpPoCxpAuto
- 7.2.1.150 quickSpinEnumerationNode CxpPoCxpStatus
- 7.2.1.151 quickSpinCommandNode CxpPoCxpTripReset
- 7.2.1.152 quickSpinCommandNode CxpPoCxpTurnOff
- 7.2.1.153 quickSpinIntegerNode DecimationHorizontal

- 7.2.1.154 quickSpinEnumerationNode DecimationHorizontalMode
- 7.2.1.155 quickSpinEnumerationNode DecimationSelector
- 7.2.1.156 quickSpinIntegerNode DecimationVertical
- 7.2.1.157 quickSpinEnumerationNode DecimationVerticalMode
- 7.2.1.158 quickSpinEnumerationNode DefectCorrectionMode
- 7.2.1.159 quickSpinBooleanNode DefectCorrectStaticEnable
- 7.2.1.160 quickSpinCommandNode DefectTableApply
- 7.2.1.161 quickSpinIntegerNode DefectTableCoordinateX
- 7.2.1.162 quickSpinIntegerNode DefectTableCoordinateY
- 7.2.1.163 quickSpinCommandNode DefectTableFactoryRestore
- 7.2.1.164 quickSpinIntegerNode DefectTableIndex
- 7.2.1.165 quickSpinIntegerNode DefectTablePixelCount
- 7.2.1.166 quickSpinCommandNode DefectTableSave
- 7.2.1.167 quickSpinEnumerationNode Deinterlacing
- 7.2.1.168 quickSpinEnumerationNode DeviceCharacterSet
- 7.2.1.169 quickSpinFloatNode DeviceClockFrequency
- 7.2.1.170 quickSpinEnumerationNode DeviceClockSelector
- 7.2.1.171 quickSpinIntegerNode DeviceConnectionSelector
- 7.2.1.172 quickSpinIntegerNode DeviceConnectionSpeed
- 7.2.1.173 quickSpinEnumerationNode DeviceConnectionStatus
- 7.2.1.174 quickSpinIntegerNode DeviceEventChannelCount
- 7.2.1.175 quickSpinStringNode DeviceFamilyName
- 7.2.1.176 quickSpinCommandNode DeviceFeaturePersistenceEnd

- 7.2.1.177 quickSpinCommandNode DeviceFeaturePersistenceStart
- 7.2.1.178 quickSpinStringNode DeviceFirmwareVersion
- 7.2.1.179 quickSpinIntegerNode DeviceGenCPVersionMajor
- 7.2.1.180 quickSpinIntegerNode DeviceGenCPVersionMinor
- 7.2.1.181 quickSpinStringNode DeviceID
- 7.2.1.182 quickSpinEnumerationNode DeviceIndicatorMode
- 7.2.1.183 quickSpinFloatNode DeviceLinkBandwidthReserve
- 7.2.1.184 quickSpinFloatNode DeviceLinkCommandTimeout
- 7.2.1.185 quickSpinIntegerNode DeviceLinkConnectionCount
- 7.2.1.186 quickSpinIntegerNode DeviceLinkCurrentThroughput
- 7.2.1.187 quickSpinEnumerationNode DeviceLinkHeartbeatMode
- 7.2.1.188 quickSpinFloatNode DeviceLinkHeartbeatTimeout
- 7.2.1.189 quickSpinIntegerNode DeviceLinkSelector
- 7.2.1.190 quickSpinIntegerNode DeviceLinkSpeed
- 7.2.1.191 quickSpinIntegerNode DeviceLinkThroughputLimit
- 7.2.1.192 quickSpinEnumerationNode DeviceLinkThroughputLimitMode
- 7.2.1.193 quickSpinIntegerNode DeviceManifestEntrySelector
- 7.2.1.194 quickSpinStringNode DeviceManifestPrimaryURL
- 7.2.1.195 quickSpinIntegerNode DeviceManifestSchemaMajorVersion
- 7.2.1.196 quickSpinIntegerNode DeviceManifestSchemaMinorVersion
- 7.2.1.197 quickSpinStringNode DeviceManifestSecondaryURL
- 7.2.1.198 quickSpinIntegerNode DeviceManifestXMLMajorVersion
- 7.2.1.199 quickSpinIntegerNode DeviceManifestXMLMinorVersion

- 7.2.1.200 quickSpinIntegerNode DeviceManifestXMLSubMinorVersion
- 7.2.1.201 quickSpinStringNode DeviceManufacturerInfo
- 7.2.1.202 quickSpinIntegerNode DeviceMaxThroughput
- 7.2.1.203 quickSpinStringNode DeviceModelName
- 7.2.1.204 quickSpinEnumerationNode DevicePowerSupplySelector
- 7.2.1.205 quickSpinCommandNode DeviceRegistersCheck
- 7.2.1.206 quickSpinEnumerationNode DeviceRegistersEndianness
- 7.2.1.207 quickSpinCommandNode DeviceRegistersStreamingEnd
- 7.2.1.208 quickSpinCommandNode DeviceRegistersStreamingStart
- 7.2.1.209 quickSpinBooleanNode DeviceRegistersValid
- 7.2.1.210 quickSpinCommandNode DeviceReset
- 7.2.1.211 quickSpinEnumerationNode DeviceScanType
- 7.2.1.212 quickSpinStringNode DeviceSerialNumber
- 7.2.1.213 quickSpinEnumerationNode DeviceSerialPortBaudRate
- 7.2.1.214 quickSpinEnumerationNode DeviceSerialPortSelector
- 7.2.1.215 quickSpinIntegerNode DeviceSFNCVersionMajor
- 7.2.1.216 quickSpinIntegerNode DeviceSFNCVersionMinor
- 7.2.1.217 quickSpinIntegerNode DeviceSFNCVersionSubMinor
- 7.2.1.218 quickSpinIntegerNode DeviceStreamChannelCount
- 7.2.1.219 quickSpinEnumerationNode DeviceStreamChannelEndianness
- 7.2.1.220 quickSpinIntegerNode DeviceStreamChannelLink
- 7.2.1.221 quickSpinIntegerNode DeviceStreamChannelPacketSize
- 7.2.1.222 quickSpinIntegerNode DeviceStreamChannelSelector

- 7.2.1.223 `quickSpinEnumerationNode` `DeviceStreamChannelType`
- 7.2.1.224 `quickSpinEnumerationNode` `DeviceTapGeometry`
- 7.2.1.225 `quickSpinFloatNode` `DeviceTemperature`
- 7.2.1.226 `quickSpinEnumerationNode` `DeviceTemperatureSelector`
- 7.2.1.227 `quickSpinEnumerationNode` `DeviceTLType`
- 7.2.1.228 `quickSpinIntegerNode` `DeviceTLVersionMajor`
- 7.2.1.229 `quickSpinIntegerNode` `DeviceTLVersionMinor`
- 7.2.1.230 `quickSpinIntegerNode` `DeviceTLVersionSubMinor`
- 7.2.1.231 `quickSpinEnumerationNode` `DeviceType`
- 7.2.1.232 `quickSpinIntegerNode` `DeviceUptime`
- 7.2.1.233 `quickSpinStringNode` `DeviceUserID`
- 7.2.1.234 `quickSpinStringNode` `DeviceVendorName`
- 7.2.1.235 `quickSpinStringNode` `DeviceVersion`
- 7.2.1.236 `quickSpinIntegerNode` `EncoderDivider`
- 7.2.1.237 `quickSpinEnumerationNode` `EncoderMode`
- 7.2.1.238 `quickSpinEnumerationNode` `EncoderOutputMode`
- 7.2.1.239 `quickSpinCommandNode` `EncoderReset`
- 7.2.1.240 `quickSpinEnumerationNode` `EncoderResetActivation`
- 7.2.1.241 `quickSpinEnumerationNode` `EncoderResetSource`
- 7.2.1.242 `quickSpinEnumerationNode` `EncoderSelector`
- 7.2.1.243 `quickSpinEnumerationNode` `EncoderSourceA`
- 7.2.1.244 `quickSpinEnumerationNode` `EncoderSourceB`
- 7.2.1.245 `quickSpinEnumerationNode` `EncoderStatus`



- 7.2.1.246 quickSpinFloatNode EncoderTimeout
- 7.2.1.247 quickSpinIntegerNode EncoderValue
- 7.2.1.248 quickSpinIntegerNode EncoderValueAtReset
- 7.2.1.249 quickSpinIntegerNode EnumerationCount
- 7.2.1.250 quickSpinIntegerNode EventAcquisitionEnd
- 7.2.1.251 quickSpinIntegerNode EventAcquisitionEndFrameID
- 7.2.1.252 quickSpinIntegerNode EventAcquisitionEndTimestamp
- 7.2.1.253 quickSpinIntegerNode EventAcquisitionError
- 7.2.1.254 quickSpinIntegerNode EventAcquisitionErrorFrameID
- 7.2.1.255 quickSpinIntegerNode EventAcquisitionErrorTimestamp
- 7.2.1.256 quickSpinIntegerNode EventAcquisitionStart
- 7.2.1.257 quickSpinIntegerNode EventAcquisitionStartFrameID
- 7.2.1.258 quickSpinIntegerNode EventAcquisitionStartTimestamp
- 7.2.1.259 quickSpinIntegerNode EventAcquisitionTransferEnd
- 7.2.1.260 quickSpinIntegerNode EventAcquisitionTransferEndFrameID
- 7.2.1.261 quickSpinIntegerNode EventAcquisitionTransferEndTimestamp
- 7.2.1.262 quickSpinIntegerNode EventAcquisitionTransferStart
- 7.2.1.263 quickSpinIntegerNode EventAcquisitionTransferStartFrameID
- 7.2.1.264 quickSpinIntegerNode EventAcquisitionTransferStartTimestamp
- 7.2.1.265 quickSpinIntegerNode EventAcquisitionTrigger
- 7.2.1.266 quickSpinIntegerNode EventAcquisitionTriggerFrameID
- 7.2.1.267 quickSpinIntegerNode EventAcquisitionTriggerTimestamp
- 7.2.1.268 quickSpinIntegerNode EventActionLate

- 7.2.1.269 quickSpinIntegerNode EventActionLateFrameID
- 7.2.1.270 quickSpinIntegerNode EventActionLateTimestamp
- 7.2.1.271 quickSpinIntegerNode EventCounter0End
- 7.2.1.272 quickSpinIntegerNode EventCounter0EndFrameID
- 7.2.1.273 quickSpinIntegerNode EventCounter0EndTimestamp
- 7.2.1.274 quickSpinIntegerNode EventCounter0Start
- 7.2.1.275 quickSpinIntegerNode EventCounter0StartFrameID
- 7.2.1.276 quickSpinIntegerNode EventCounter0StartTimestamp
- 7.2.1.277 quickSpinIntegerNode EventCounter1End
- 7.2.1.278 quickSpinIntegerNode EventCounter1EndFrameID
- 7.2.1.279 quickSpinIntegerNode EventCounter1EndTimestamp
- 7.2.1.280 quickSpinIntegerNode EventCounter1Start
- 7.2.1.281 quickSpinIntegerNode EventCounter1StartFrameID
- 7.2.1.282 quickSpinIntegerNode EventCounter1StartTimestamp
- 7.2.1.283 quickSpinIntegerNode EventEncoder0Restarted
- 7.2.1.284 quickSpinIntegerNode EventEncoder0RestartedFrameID
- 7.2.1.285 quickSpinIntegerNode EventEncoder0RestartedTimestamp
- 7.2.1.286 quickSpinIntegerNode EventEncoder0Stopped
- 7.2.1.287 quickSpinIntegerNode EventEncoder0StoppedFrameID
- 7.2.1.288 quickSpinIntegerNode EventEncoder0StoppedTimestamp
- 7.2.1.289 quickSpinIntegerNode EventEncoder1Restarted
- 7.2.1.290 quickSpinIntegerNode EventEncoder1RestartedFrameID
- 7.2.1.291 quickSpinIntegerNode EventEncoder1RestartedTimestamp

- 7.2.1.292 quickSpinIntegerNode EventEncoder1Stopped
- 7.2.1.293 quickSpinIntegerNode EventEncoder1StoppedFrameID
- 7.2.1.294 quickSpinIntegerNode EventEncoder1StoppedTimestamp
- 7.2.1.295 quickSpinIntegerNode EventError
- 7.2.1.296 quickSpinIntegerNode EventErrorCode
- 7.2.1.297 quickSpinIntegerNode EventErrorFrameID
- 7.2.1.298 quickSpinIntegerNode EventErrorTimestamp
- 7.2.1.299 quickSpinIntegerNode EventExposureEnd
- 7.2.1.300 quickSpinIntegerNode EventExposureEndFrameID
- 7.2.1.301 quickSpinIntegerNode EventExposureEndTimestamp
- 7.2.1.302 quickSpinIntegerNode EventExposureStart
- 7.2.1.303 quickSpinIntegerNode EventExposureStartFrameID
- 7.2.1.304 quickSpinIntegerNode EventExposureStartTimestamp
- 7.2.1.305 quickSpinIntegerNode EventFrameBurstEnd
- 7.2.1.306 quickSpinIntegerNode EventFrameBurstEndFrameID
- 7.2.1.307 quickSpinIntegerNode EventFrameBurstEndTimestamp
- 7.2.1.308 quickSpinIntegerNode EventFrameBurstStart
- 7.2.1.309 quickSpinIntegerNode EventFrameBurstStartFrameID
- 7.2.1.310 quickSpinIntegerNode EventFrameBurstStartTimestamp
- 7.2.1.311 quickSpinIntegerNode EventFrameEnd
- 7.2.1.312 quickSpinIntegerNode EventFrameEndFrameID
- 7.2.1.313 quickSpinIntegerNode EventFrameEndTimestamp
- 7.2.1.314 quickSpinIntegerNode EventFrameStart

- 7.2.1.315 quickSpinIntegerNode EventFrameStartFrameID
- 7.2.1.316 quickSpinIntegerNode EventFrameStartTimestamp
- 7.2.1.317 quickSpinIntegerNode EventFrameTransferEnd
- 7.2.1.318 quickSpinIntegerNode EventFrameTransferEndFrameID
- 7.2.1.319 quickSpinIntegerNode EventFrameTransferEndTimestamp
- 7.2.1.320 quickSpinIntegerNode EventFrameTransferStart
- 7.2.1.321 quickSpinIntegerNode EventFrameTransferStartFrameID
- 7.2.1.322 quickSpinIntegerNode EventFrameTransferStartTimestamp
- 7.2.1.323 quickSpinIntegerNode EventFrameTrigger
- 7.2.1.324 quickSpinIntegerNode EventFrameTriggerFrameID
- 7.2.1.325 quickSpinIntegerNode EventFrameTriggerTimestamp
- 7.2.1.326 quickSpinIntegerNode EventLine0AnyEdge
- 7.2.1.327 quickSpinIntegerNode EventLine0AnyEdgeFrameID
- 7.2.1.328 quickSpinIntegerNode EventLine0AnyEdgeTimestamp
- 7.2.1.329 quickSpinIntegerNode EventLine0FallingEdge
- 7.2.1.330 quickSpinIntegerNode EventLine0FallingEdgeFrameID
- 7.2.1.331 quickSpinIntegerNode EventLine0FallingEdgeTimestamp
- 7.2.1.332 quickSpinIntegerNode EventLine0RisingEdge
- 7.2.1.333 quickSpinIntegerNode EventLine0RisingEdgeFrameID
- 7.2.1.334 quickSpinIntegerNode EventLine0RisingEdgeTimestamp
- 7.2.1.335 quickSpinIntegerNode EventLine1AnyEdge
- 7.2.1.336 quickSpinIntegerNode EventLine1AnyEdgeFrameID
- 7.2.1.337 quickSpinIntegerNode EventLine1AnyEdgeTimestamp

- 7.2.1.338 quickSpinIntegerNode EventLine1FallingEdge
- 7.2.1.339 quickSpinIntegerNode EventLine1FallingEdgeFrameID
- 7.2.1.340 quickSpinIntegerNode EventLine1FallingEdgeTimestamp
- 7.2.1.341 quickSpinIntegerNode EventLine1RisingEdge
- 7.2.1.342 quickSpinIntegerNode EventLine1RisingEdgeFrameID
- 7.2.1.343 quickSpinIntegerNode EventLine1RisingEdgeTimestamp
- 7.2.1.344 quickSpinIntegerNode EventLinkSpeedChange
- 7.2.1.345 quickSpinIntegerNode EventLinkSpeedChangeFrameID
- 7.2.1.346 quickSpinIntegerNode EventLinkSpeedChangeTimestamp
- 7.2.1.347 quickSpinIntegerNode EventLinkTrigger0
- 7.2.1.348 quickSpinIntegerNode EventLinkTrigger0FrameID
- 7.2.1.349 quickSpinIntegerNode EventLinkTrigger0Timestamp
- 7.2.1.350 quickSpinIntegerNode EventLinkTrigger1
- 7.2.1.351 quickSpinIntegerNode EventLinkTrigger1FrameID
- 7.2.1.352 quickSpinIntegerNode EventLinkTrigger1Timestamp
- 7.2.1.353 quickSpinEnumerationNode EventNotification
- 7.2.1.354 quickSpinEnumerationNode EventSelector
- 7.2.1.355 quickSpinIntegerNode EventSequencerSetChange
- 7.2.1.356 quickSpinIntegerNode EventSequencerSetChangeFrameID
- 7.2.1.357 quickSpinIntegerNode EventSequencerSetChangeTimestamp
- 7.2.1.358 quickSpinStringNode EventSerialData
- 7.2.1.359 quickSpinIntegerNode EventSerialDataLength
- 7.2.1.360 quickSpinIntegerNode EventSerialPortReceive

- 7.2.1.361 quickSpinIntegerNode EventSerialPortReceiveTimestamp
- 7.2.1.362 quickSpinBooleanNode EventSerialReceiveOverflow
- 7.2.1.363 quickSpinIntegerNode EventStream0TransferBlockEnd
- 7.2.1.364 quickSpinIntegerNode EventStream0TransferBlockEndFrameID
- 7.2.1.365 quickSpinIntegerNode EventStream0TransferBlockEndTimestamp
- 7.2.1.366 quickSpinIntegerNode EventStream0TransferBlockStart
- 7.2.1.367 quickSpinIntegerNode EventStream0TransferBlockStartFrameID
- 7.2.1.368 quickSpinIntegerNode EventStream0TransferBlockStartTimestamp
- 7.2.1.369 quickSpinIntegerNode EventStream0TransferBlockTrigger
- 7.2.1.370 quickSpinIntegerNode EventStream0TransferBlockTriggerFrameID
- 7.2.1.371 quickSpinIntegerNode EventStream0TransferBlockTriggerTimestamp
- 7.2.1.372 quickSpinIntegerNode EventStream0TransferBurstEnd
- 7.2.1.373 quickSpinIntegerNode EventStream0TransferBurstEndFrameID
- 7.2.1.374 quickSpinIntegerNode EventStream0TransferBurstEndTimestamp
- 7.2.1.375 quickSpinIntegerNode EventStream0TransferBurstStart
- 7.2.1.376 quickSpinIntegerNode EventStream0TransferBurstStartFrameID
- 7.2.1.377 quickSpinIntegerNode EventStream0TransferBurstStartTimestamp
- 7.2.1.378 quickSpinIntegerNode EventStream0TransferEnd
- 7.2.1.379 quickSpinIntegerNode EventStream0TransferEndFrameID
- 7.2.1.380 quickSpinIntegerNode EventStream0TransferEndTimestamp
- 7.2.1.381 quickSpinIntegerNode EventStream0TransferOverflow
- 7.2.1.382 quickSpinIntegerNode EventStream0TransferOverflowFrameID
- 7.2.1.383 quickSpinIntegerNode EventStream0TransferOverflowTimestamp

- 7.2.1.384 quickSpinIntegerNode EventStream0TransferPause
- 7.2.1.385 quickSpinIntegerNode EventStream0TransferPauseFrameID
- 7.2.1.386 quickSpinIntegerNode EventStream0TransferPauseTimestamp
- 7.2.1.387 quickSpinIntegerNode EventStream0TransferResume
- 7.2.1.388 quickSpinIntegerNode EventStream0TransferResumeFrameID
- 7.2.1.389 quickSpinIntegerNode EventStream0TransferResumeTimestamp
- 7.2.1.390 quickSpinIntegerNode EventStream0TransferStart
- 7.2.1.391 quickSpinIntegerNode EventStream0TransferStartFrameID
- 7.2.1.392 quickSpinIntegerNode EventStream0TransferStartTimestamp
- 7.2.1.393 quickSpinIntegerNode EventTest
- 7.2.1.394 quickSpinIntegerNode EventTestTimestamp
- 7.2.1.395 quickSpinIntegerNode EventTimer0End
- 7.2.1.396 quickSpinIntegerNode EventTimer0EndFrameID
- 7.2.1.397 quickSpinIntegerNode EventTimer0EndTimestamp
- 7.2.1.398 quickSpinIntegerNode EventTimer0Start
- 7.2.1.399 quickSpinIntegerNode EventTimer0StartFrameID
- 7.2.1.400 quickSpinIntegerNode EventTimer0StartTimestamp
- 7.2.1.401 quickSpinIntegerNode EventTimer1End
- 7.2.1.402 quickSpinIntegerNode EventTimer1EndFrameID
- 7.2.1.403 quickSpinIntegerNode EventTimer1EndTimestamp
- 7.2.1.404 quickSpinIntegerNode EventTimer1Start
- 7.2.1.405 quickSpinIntegerNode EventTimer1StartFrameID
- 7.2.1.406 quickSpinIntegerNode EventTimer1StartTimestamp

- 7.2.1.407 quickSpinEnumerationNode ExposureActiveMode
- 7.2.1.408 quickSpinEnumerationNode ExposureAuto
- 7.2.1.409 quickSpinEnumerationNode ExposureMode
- 7.2.1.410 quickSpinFloatNode ExposureTime
- 7.2.1.411 quickSpinEnumerationNode ExposureTimeMode
- 7.2.1.412 quickSpinEnumerationNode ExposureTimeSelector
- 7.2.1.413 quickSpinCommandNode FactoryReset
- 7.2.1.414 quickSpinRegisterNode FileAccessBuffer
- 7.2.1.415 quickSpinIntegerNode FileAccessLength
- 7.2.1.416 quickSpinIntegerNode FileAccessOffset
- 7.2.1.417 quickSpinEnumerationNode FileOpenMode
- 7.2.1.418 quickSpinCommandNode FileOperationExecute
- 7.2.1.419 quickSpinIntegerNode FileOperationResult
- 7.2.1.420 quickSpinEnumerationNode FileOperationSelector
- 7.2.1.421 quickSpinEnumerationNode FileOperationStatus
- 7.2.1.422 quickSpinEnumerationNode FileSelector
- 7.2.1.423 quickSpinIntegerNode FileSize
- 7.2.1.424 quickSpinFloatNode Gain
- 7.2.1.425 quickSpinEnumerationNode GainAuto
- 7.2.1.426 quickSpinEnumerationNode GainAutoBalance
- 7.2.1.427 quickSpinEnumerationNode GainSelector
- 7.2.1.428 quickSpinFloatNode Gamma
- 7.2.1.429 quickSpinBooleanNode GammaEnable



- 7.2.1.430 quickSpinIntegerNode GevActiveLinkCount
- 7.2.1.431 quickSpinEnumerationNode GevCCP
- 7.2.1.432 quickSpinIntegerNode GevCurrentDefaultGateway
- 7.2.1.433 quickSpinIntegerNode GevCurrentIPAddress
- 7.2.1.434 quickSpinBooleanNode GevCurrentIPConfigurationDHCP
- 7.2.1.435 quickSpinBooleanNode GevCurrentIPConfigurationLLA
- 7.2.1.436 quickSpinBooleanNode GevCurrentIPConfigurationPersistentIP
- 7.2.1.437 quickSpinEnumerationNode GevCurrentPhysicalLinkConfiguration
- 7.2.1.438 quickSpinIntegerNode GevCurrentSubnetMask
- 7.2.1.439 quickSpinIntegerNode GevDiscoveryAckDelay
- 7.2.1.440 quickSpinStringNode GevFirstURL
- 7.2.1.441 quickSpinBooleanNode GevGVCPExtendedStatusCodes
- 7.2.1.442 quickSpinEnumerationNode GevGVCPExtendedStatusCodesSelector
- 7.2.1.443 quickSpinBooleanNode GevGVCPHeartbeatDisable
- 7.2.1.444 quickSpinBooleanNode GevGVCPPendingAck
- 7.2.1.445 quickSpinIntegerNode GevGVCPPendingTimeout
- 7.2.1.446 quickSpinEnumerationNode GevGVSPExtendedIDMode
- 7.2.1.447 quickSpinIntegerNode GevHeartbeatTimeout
- 7.2.1.448 quickSpinBooleanNode GevIEEE1588
- 7.2.1.449 quickSpinEnumerationNode GevIEEE1588ClockAccuracy
- 7.2.1.450 quickSpinEnumerationNode GevIEEE1588Mode
- 7.2.1.451 quickSpinEnumerationNode GevIEEE1588Status
- 7.2.1.452 quickSpinIntegerNode GevInterfaceSelector

- 7.2.1.453 quickSpinEnumerationNode GevIPConfigurationStatus
- 7.2.1.454 quickSpinIntegerNode GevMACAddress
- 7.2.1.455 quickSpinIntegerNode GevMCDA
- 7.2.1.456 quickSpinIntegerNode GevMCPhostPort
- 7.2.1.457 quickSpinIntegerNode GevMCRC
- 7.2.1.458 quickSpinIntegerNode GevMCSP
- 7.2.1.459 quickSpinIntegerNode GevMCTT
- 7.2.1.460 quickSpinIntegerNode GevNumberOfInterfaces
- 7.2.1.461 quickSpinBooleanNode GevPAUSEFrameReception
- 7.2.1.462 quickSpinBooleanNode GevPAUSEFrameTransmission
- 7.2.1.463 quickSpinIntegerNode GevPersistentDefaultGateway
- 7.2.1.464 quickSpinIntegerNode GevPersistentIPAddress
- 7.2.1.465 quickSpinIntegerNode GevPersistentSubnetMask
- 7.2.1.466 quickSpinEnumerationNode GevPhysicalLinkConfiguration
- 7.2.1.467 quickSpinIntegerNode GevPrimaryApplicationIPAddress
- 7.2.1.468 quickSpinIntegerNode GevPrimaryApplicationSocket
- 7.2.1.469 quickSpinIntegerNode GevPrimaryApplicationSwitchoverKey
- 7.2.1.470 quickSpinBooleanNode GevSCCFGAllInTransmission
- 7.2.1.471 quickSpinBooleanNode GevSCCFGExtendedChunkData
- 7.2.1.472 quickSpinBooleanNode GevSCCFGPacketResendDestination
- 7.2.1.473 quickSpinBooleanNode GevSCCFGUnconditionalStreaming
- 7.2.1.474 quickSpinIntegerNode GevSCDA
- 7.2.1.475 quickSpinIntegerNode GevSCPD

- 7.2.1.476 quickSpinIntegerNode GevSCPDirection
- 7.2.1.477 quickSpinIntegerNode GevSCPHostPort
- 7.2.1.478 quickSpinIntegerNode GevSCPIInterfaceIndex
- 7.2.1.479 quickSpinBooleanNode GevSCPSBigEndian
- 7.2.1.480 quickSpinBooleanNode GevSCPSDoNotFragment
- 7.2.1.481 quickSpinBooleanNode GevSCPSFireTestPacket
- 7.2.1.482 quickSpinIntegerNode GevSCSPacketSize
- 7.2.1.483 quickSpinIntegerNode GevSCSP
- 7.2.1.484 quickSpinBooleanNode GevSCZoneConfigurationLock
- 7.2.1.485 quickSpinIntegerNode GevSCZoneCount
- 7.2.1.486 quickSpinIntegerNode GevSCZoneDirectionAll
- 7.2.1.487 quickSpinStringNode GevSecondURL
- 7.2.1.488 quickSpinIntegerNode GevStreamChannelSelector
- 7.2.1.489 quickSpinBooleanNode GevSupportedOption
- 7.2.1.490 quickSpinEnumerationNode GevSupportedOptionSelector
- 7.2.1.491 quickSpinIntegerNode GevTimestampTickFrequency
- 7.2.1.492 quickSpinIntegerNode GuiXmlManifestAddress
- 7.2.1.493 quickSpinIntegerNode Height
- 7.2.1.494 quickSpinIntegerNode HeightMax
- 7.2.1.495 quickSpinBooleanNode ImageComponentEnable
- 7.2.1.496 quickSpinEnumerationNode ImageComponentSelector
- 7.2.1.497 quickSpinFloatNode ImageCompressionBitrate
- 7.2.1.498 quickSpinEnumerationNode ImageCompressionJPEGFormatOption

- 7.2.1.499 quickSpinEnumerationNode ImageCompressionMode
- 7.2.1.500 quickSpinIntegerNode ImageCompressionQuality
- 7.2.1.501 quickSpinEnumerationNode ImageCompressionRateOption
- 7.2.1.502 quickSpinBooleanNode IspEnable
- 7.2.1.503 quickSpinFloatNode LineFilterWidth
- 7.2.1.504 quickSpinEnumerationNode LineFormat
- 7.2.1.505 quickSpinEnumerationNode LineInputFilterSelector
- 7.2.1.506 quickSpinBooleanNode LineInverter
- 7.2.1.507 quickSpinEnumerationNode LineMode
- 7.2.1.508 quickSpinIntegerNode LinePitch
- 7.2.1.509 quickSpinEnumerationNode LineSelector
- 7.2.1.510 quickSpinEnumerationNode LineSource
- 7.2.1.511 quickSpinBooleanNode LineStatus
- 7.2.1.512 quickSpinIntegerNode LineStatusAll
- 7.2.1.513 quickSpinIntegerNode LinkErrorCount
- 7.2.1.514 quickSpinIntegerNode LinkUptime
- 7.2.1.515 quickSpinEnumerationNode LogicBlockLUTInputActivation
- 7.2.1.516 quickSpinEnumerationNode LogicBlockLUTInputSelector
- 7.2.1.517 quickSpinEnumerationNode LogicBlockLUTInputSource
- 7.2.1.518 quickSpinBooleanNode LogicBlockLUTOutputValue
- 7.2.1.519 quickSpinIntegerNode LogicBlockLUTOutputValueAll
- 7.2.1.520 quickSpinIntegerNode LogicBlockLUTRowIndex
- 7.2.1.521 quickSpinEnumerationNode LogicBlockLUTSelector

- 7.2.1.522 quickSpinEnumerationNode LogicBlockSelector
- 7.2.1.523 quickSpinBooleanNode LUTEnable
- 7.2.1.524 quickSpinIntegerNode LUTIndex
- 7.2.1.525 quickSpinEnumerationNode LUTSelector
- 7.2.1.526 quickSpinIntegerNode LUTValue
- 7.2.1.527 quickSpinRegisterNode LUTValueAll
- 7.2.1.528 quickSpinIntegerNode MaxDeviceResetTime
- 7.2.1.529 quickSpinIntegerNode OffsetX
- 7.2.1.530 quickSpinIntegerNode OffsetY
- 7.2.1.531 quickSpinIntegerNode PacketResendRequestCount
- 7.2.1.532 quickSpinIntegerNode PayloadSize
- 7.2.1.533 quickSpinEnumerationNode PixelColorFilter
- 7.2.1.534 quickSpinIntegerNode PixelDynamicRangeMax
- 7.2.1.535 quickSpinIntegerNode PixelDynamicRangeMin
- 7.2.1.536 quickSpinEnumerationNode PixelFormat
- 7.2.1.537 quickSpinIntegerNode PixelFormatInfoID
- 7.2.1.538 quickSpinEnumerationNode PixelFormatInfoSelector
- 7.2.1.539 quickSpinEnumerationNode PixelSize
- 7.2.1.540 quickSpinFloatNode PowerSupplyCurrent
- 7.2.1.541 quickSpinFloatNode PowerSupplyVoltage
- 7.2.1.542 quickSpinEnumerationNode RegionDestination
- 7.2.1.543 quickSpinEnumerationNode RegionMode
- 7.2.1.544 quickSpinEnumerationNode RegionSelector

- 7.2.1.545 `quickSpinBooleanNode ReverseX`
- 7.2.1.546 `quickSpinBooleanNode ReverseY`
- 7.2.1.547 `quickSpinEnumerationNode RgbTransformLightSource`
- 7.2.1.548 `quickSpinFloatNode Saturation`
- 7.2.1.549 `quickSpinBooleanNode SaturationEnable`
- 7.2.1.550 `quickSpinFloatNode Scan3dAxisMax`
- 7.2.1.551 `quickSpinFloatNode Scan3dAxisMin`
- 7.2.1.552 `quickSpinFloatNode Scan3dCoordinateOffset`
- 7.2.1.553 `quickSpinEnumerationNode Scan3dCoordinateReferenceSelector`
- 7.2.1.554 `quickSpinFloatNode Scan3dCoordinateReferenceValue`
- 7.2.1.555 `quickSpinFloatNode Scan3dCoordinateScale`
- 7.2.1.556 `quickSpinEnumerationNode Scan3dCoordinateSelector`
- 7.2.1.557 `quickSpinEnumerationNode Scan3dCoordinateSystem`
- 7.2.1.558 `quickSpinEnumerationNode Scan3dCoordinateSystemReference`
- 7.2.1.559 `quickSpinEnumerationNode Scan3dCoordinateTransformSelector`
- 7.2.1.560 `quickSpinEnumerationNode Scan3dDistanceUnit`
- 7.2.1.561 `quickSpinBooleanNode Scan3dInvalidDataFlag`
- 7.2.1.562 `quickSpinFloatNode Scan3dInvalidDataValue`
- 7.2.1.563 `quickSpinEnumerationNode Scan3dOutputMode`
- 7.2.1.564 `quickSpinFloatNode Scan3dTransformValue`
- 7.2.1.565 `quickSpinStringNode SensorDescription`
- 7.2.1.566 `quickSpinEnumerationNode SensorDigitizationTaps`
- 7.2.1.567 `quickSpinIntegerNode SensorHeight`

- 7.2.1.568 quickSpinEnumerationNode SensorShutterMode
- 7.2.1.569 quickSpinEnumerationNode SensorTaps
- 7.2.1.570 quickSpinIntegerNode SensorWidth
- 7.2.1.571 quickSpinEnumerationNode SequencerConfigurationMode
- 7.2.1.572 quickSpinEnumerationNode SequencerConfigurationValid
- 7.2.1.573 quickSpinBooleanNode SequencerFeatureEnable
- 7.2.1.574 quickSpinEnumerationNode SequencerMode
- 7.2.1.575 quickSpinIntegerNode SequencerPathSelector
- 7.2.1.576 quickSpinIntegerNode SequencerSetActive
- 7.2.1.577 quickSpinCommandNode SequencerSetLoad
- 7.2.1.578 quickSpinIntegerNode SequencerSetNext
- 7.2.1.579 quickSpinCommandNode SequencerSetSave
- 7.2.1.580 quickSpinIntegerNode SequencerSetSelector
- 7.2.1.581 quickSpinIntegerNode SequencerSetStart
- 7.2.1.582 quickSpinEnumerationNode SequencerSetValid
- 7.2.1.583 quickSpinEnumerationNode SequencerTriggerActivation
- 7.2.1.584 quickSpinEnumerationNode SequencerTriggerSource
- 7.2.1.585 quickSpinEnumerationNode SerialPortBaudRate
- 7.2.1.586 quickSpinIntegerNode SerialPortDataBits
- 7.2.1.587 quickSpinEnumerationNode SerialPortParity
- 7.2.1.588 quickSpinEnumerationNode SerialPortSelector
- 7.2.1.589 quickSpinEnumerationNode SerialPortSource
- 7.2.1.590 quickSpinEnumerationNode SerialPortStopBits

- 7.2.1.591 quickSpinIntegerNode SerialReceiveFramingErrorCount
- 7.2.1.592 quickSpinIntegerNode SerialReceiveParityErrorCount
- 7.2.1.593 quickSpinCommandNode SerialReceiveQueueClear
- 7.2.1.594 quickSpinIntegerNode SerialReceiveQueueCurrentCharacterCount
- 7.2.1.595 quickSpinIntegerNode SerialReceiveQueueMaxCharacterCount
- 7.2.1.596 quickSpinIntegerNode SerialTransmitQueueCurrentCharacterCount
- 7.2.1.597 quickSpinIntegerNode SerialTransmitQueueMaxCharacterCount
- 7.2.1.598 quickSpinFloatNode Sharpening
- 7.2.1.599 quickSpinBooleanNode SharpeningAuto
- 7.2.1.600 quickSpinBooleanNode SharpeningEnable
- 7.2.1.601 quickSpinFloatNode SharpeningThreshold
- 7.2.1.602 quickSpinCommandNode SoftwareSignalPulse
- 7.2.1.603 quickSpinEnumerationNode SoftwareSignalSelector
- 7.2.1.604 quickSpinIntegerNode SourceCount
- 7.2.1.605 quickSpinEnumerationNode SourceSelector
- 7.2.1.606 quickSpinIntegerNode Test0001
- 7.2.1.607 quickSpinCommandNode TestEventGenerate
- 7.2.1.608 quickSpinEnumerationNode TestPattern
- 7.2.1.609 quickSpinEnumerationNode TestPatternGeneratorSelector
- 7.2.1.610 quickSpinIntegerNode TestPendingAck
- 7.2.1.611 quickSpinFloatNode TimerDelay
- 7.2.1.612 quickSpinFloatNode TimerDuration
- 7.2.1.613 quickSpinCommandNode TimerReset



- 7.2.1.614 quickSpinEnumerationNode TimerSelector
- 7.2.1.615 quickSpinEnumerationNode TimerStatus
- 7.2.1.616 quickSpinEnumerationNode TimerTriggerActivation
- 7.2.1.617 quickSpinEnumerationNode TimerTriggerSource
- 7.2.1.618 quickSpinFloatNode TimerValue
- 7.2.1.619 quickSpinIntegerNode Timestamp
- 7.2.1.620 quickSpinCommandNode TimestampLatch
- 7.2.1.621 quickSpinIntegerNode TimestampLatchValue
- 7.2.1.622 quickSpinCommandNode TimestampReset
- 7.2.1.623 quickSpinIntegerNode TLParamsLocked
- 7.2.1.624 quickSpinCommandNode TransferAbort
- 7.2.1.625 quickSpinIntegerNode TransferBlockCount
- 7.2.1.626 quickSpinIntegerNode TransferBurstCount
- 7.2.1.627 quickSpinEnumerationNode TransferComponentSelector
- 7.2.1.628 quickSpinEnumerationNode TransferControlMode
- 7.2.1.629 quickSpinEnumerationNode TransferOperationMode
- 7.2.1.630 quickSpinCommandNode TransferPause
- 7.2.1.631 quickSpinIntegerNode TransferQueueCurrentBlockCount
- 7.2.1.632 quickSpinIntegerNode TransferQueueMaxBlockCount
- 7.2.1.633 quickSpinEnumerationNode TransferQueueMode
- 7.2.1.634 quickSpinIntegerNode TransferQueueOverflowCount
- 7.2.1.635 quickSpinCommandNode TransferResume
- 7.2.1.636 quickSpinEnumerationNode TransferSelector

- 7.2.1.637 quickSpinCommandNode TransferStart
- 7.2.1.638 quickSpinBooleanNode TransferStatus
- 7.2.1.639 quickSpinEnumerationNode TransferStatusSelector
- 7.2.1.640 quickSpinCommandNode TransferStop
- 7.2.1.641 quickSpinIntegerNode TransferStreamChannel
- 7.2.1.642 quickSpinEnumerationNode TransferTriggerActivation
- 7.2.1.643 quickSpinEnumerationNode TransferTriggerMode
- 7.2.1.644 quickSpinEnumerationNode TransferTriggerSelector
- 7.2.1.645 quickSpinEnumerationNode TransferTriggerSource
- 7.2.1.646 quickSpinEnumerationNode TriggerActivation
- 7.2.1.647 quickSpinFloatNode TriggerDelay
- 7.2.1.648 quickSpinIntegerNode TriggerDivider
- 7.2.1.649 quickSpinCommandNode TriggerEventTest
- 7.2.1.650 quickSpinEnumerationNode TriggerMode
- 7.2.1.651 quickSpinIntegerNode TriggerMultiplier
- 7.2.1.652 quickSpinEnumerationNode TriggerOverlap
- 7.2.1.653 quickSpinEnumerationNode TriggerSelector
- 7.2.1.654 quickSpinCommandNode TriggerSoftware
- 7.2.1.655 quickSpinEnumerationNode TriggerSource
- 7.2.1.656 quickSpinEnumerationNode UserOutputSelector
- 7.2.1.657 quickSpinBooleanNode UserOutputValue
- 7.2.1.658 quickSpinIntegerNode UserOutputValueAll
- 7.2.1.659 quickSpinIntegerNode UserOutputValueAllMask

- 7.2.1.660 [quickSpinEnumerationNode UserSetDefault](#)
- 7.2.1.661 [quickSpinBooleanNode UserSetFeatureEnable](#)
- 7.2.1.662 [quickSpinCommandNode UserSetLoad](#)
- 7.2.1.663 [quickSpinCommandNode UserSetSave](#)
- 7.2.1.664 [quickSpinEnumerationNode UserSetSelector](#)
- 7.2.1.665 [quickSpinBooleanNode V3\\_3Enable](#)
- 7.2.1.666 [quickSpinFloatNode WhiteClip](#)
- 7.2.1.667 [quickSpinEnumerationNode WhiteClipSelector](#)
- 7.2.1.668 [quickSpinIntegerNode Width](#)
- 7.2.1.669 [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](#)
- [quickSpinBooleanNode DeviceMulticastMonitorMode](#)
- [quickSpinEnumerationNode DeviceEndiannessMechanism](#)
- [quickSpinStringNode DeviceInstanceld](#)
- [quickSpinEnumerationNode DeviceCurrentSpeed](#)
- [quickSpinBooleanNode DeviceU3VProtocol](#)

### 7.3.1 Field Documentation

- 7.3.1.1 [quickSpinEnumerationNode DeviceAccessStatus](#)
- 7.3.1.2 [quickSpinEnumerationNode DeviceCurrentSpeed](#)
- 7.3.1.3 [quickSpinStringNode DeviceDisplayName](#)
- 7.3.1.4 [quickSpinStringNode DeviceDriverVersion](#)
- 7.3.1.5 [quickSpinEnumerationNode DeviceEndiannessMechanism](#)
- 7.3.1.6 [quickSpinStringNode DeviceId](#)
- 7.3.1.7 [quickSpinStringNode DeviceInstanceld](#)
- 7.3.1.8 [quickSpinBooleanNode DevicesUpdater](#)
- 7.3.1.9 [quickSpinIntegerNode DeviceLinkSpeed](#)
- 7.3.1.10 [quickSpinStringNode DeviceModelName](#)
- 7.3.1.11 [quickSpinBooleanNode DeviceMulticastMonitorMode](#)
- 7.3.1.12 [quickSpinStringNode DeviceSerialNumber](#)
- 7.3.1.13 [quickSpinEnumerationNode DeviceType](#)
- 7.3.1.14 [quickSpinBooleanNode DeviceU3VProtocol](#)
- 7.3.1.15 [quickSpinStringNode DeviceUserId](#)

- 7.3.1.16 `quickSpinStringNode` DeviceVendorName
- 7.3.1.17 `quickSpinStringNode` DeviceVersion
- 7.3.1.18 `quickSpinEnumerationNode` GenICamXMLLocation
- 7.3.1.19 `quickSpinStringNode` GenICamXMLPath
- 7.3.1.20 `quickSpinEnumerationNode` GevCCP
- 7.3.1.21 `quickSpinCommandNode` GevDeviceDiscoverMaximumPacketSize
- 7.3.1.22 `quickSpinIntegerNode` GevDeviceGateway
- 7.3.1.23 `quickSpinIntegerNode` GevDeviceIPAddress
- 7.3.1.24 `quickSpinBooleanNode` GevDevicesWrongSubnet
- 7.3.1.25 `quickSpinIntegerNode` GevDeviceMACAddress
- 7.3.1.26 `quickSpinIntegerNode` GevDeviceMaximumPacketSize
- 7.3.1.27 `quickSpinIntegerNode` GevDeviceMaximumRetryCount
- 7.3.1.28 `quickSpinBooleanNode` GevDeviceModelsBigEndian
- 7.3.1.29 `quickSpinIntegerNode` GevDevicePort
- 7.3.1.30 `quickSpinIntegerNode` GevDeviceReadAndWriteTimeout
- 7.3.1.31 `quickSpinIntegerNode` GevDeviceSubnetMask
- 7.3.1.32 `quickSpinIntegerNode` GevVersionMajor
- 7.3.1.33 `quickSpinIntegerNode` GevVersionMinor
- 7.3.1.34 `quickSpinEnumerationNode` GUIXMLLocation
- 7.3.1.35 `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](#)
- [quickSpinEnumerationNode POEStatus](#)
- [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 quickSpinCommandNode ActionCommand

#### 7.4.1.2 quickSpinCommandNode AutoForceIP

#### 7.4.1.3 quickSpinEnumerationNode DeviceAccessStatus

#### 7.4.1.4 quickSpinIntegerNode DeviceCount

- 7.4.1.5 quickSpinStringNode DeviceID
- 7.4.1.6 quickSpinStringNode DeviceModelName
- 7.4.1.7 quickSpinIntegerNode DeviceSelector
- 7.4.1.8 quickSpinStringNode DeviceUnlock
- 7.4.1.9 quickSpinCommandNode DeviceUpdateList
- 7.4.1.10 quickSpinStringNode DeviceVendorName
- 7.4.1.11 quickSpinIntegerNode GevActionDeviceKey
- 7.4.1.12 quickSpinIntegerNode GevActionGroupKey
- 7.4.1.13 quickSpinIntegerNode GevActionGroupMask
- 7.4.1.14 quickSpinIntegerNode GevActionTime
- 7.4.1.15 quickSpinIntegerNode GevDeviceIPAddress
- 7.4.1.16 quickSpinIntegerNode GevDeviceMACAddress
- 7.4.1.17 quickSpinIntegerNode GevDeviceSubnetMask
- 7.4.1.18 quickSpinIntegerNode GevInterfaceGateway
- 7.4.1.19 quickSpinIntegerNode GevInterfaceIPAddress
- 7.4.1.20 quickSpinIntegerNode GevInterfaceMACAddress
- 7.4.1.21 quickSpinIntegerNode GevInterfaceSubnetMask
- 7.4.1.22 quickSpinStringNode HostAdapterDriverVersion
- 7.4.1.23 quickSpinStringNode HostAdapterName
- 7.4.1.24 quickSpinStringNode HostAdapterVendor
- 7.4.1.25 quickSpinIntegerNode IncompatibleDeviceCount
- 7.4.1.26 quickSpinStringNode IncompatibleDeviceID
- 7.4.1.27 quickSpinStringNode IncompatibleDeviceModelName

7.4.1.28 **quickSpinIntegerNode** IncompatibleDeviceSelector

7.4.1.29 **quickSpinStringNode** IncompatibleDeviceVendorName

7.4.1.30 **quickSpinIntegerNode** IncompatibleGevDeviceIPAddress

7.4.1.31 **quickSpinIntegerNode** IncompatibleGevDeviceMACAddress

7.4.1.32 **quickSpinIntegerNode** IncompatibleGevDeviceSubnetMask

7.4.1.33 **quickSpinStringNode** InterfaceDisplayName

7.4.1.34 **quickSpinStringNode** InterfaceID

7.4.1.35 **quickSpinStringNode** InterfaceType

7.4.1.36 **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 quickSpinIntegerNode GevFailedPacketCount
- 7.5.1.2 quickSpinIntegerNode GevMaximumNumberResendBuffers
- 7.5.1.3 quickSpinIntegerNode GevMaximumNumberResendRequests
- 7.5.1.4 quickSpinBooleanNode GevPacketResendMode
- 7.5.1.5 quickSpinIntegerNode GevPacketResendTimeout
- 7.5.1.6 quickSpinIntegerNode GevResendPacketCount
- 7.5.1.7 quickSpinIntegerNode GevResendRequestCount
- 7.5.1.8 quickSpinIntegerNode GevTotalPacketCount
- 7.5.1.9 quickSpinIntegerNode StreamBlockTransferSize
- 7.5.1.10 quickSpinIntegerNode StreamBufferCountManual
- 7.5.1.11 quickSpinIntegerNode StreamBufferCountMax
- 7.5.1.12 quickSpinEnumerationNode StreamBufferCountMode
- 7.5.1.13 quickSpinIntegerNode StreamBufferCountResult
- 7.5.1.14 quickSpinEnumerationNode StreamBufferHandlingMode
- 7.5.1.15 quickSpinIntegerNode StreamBufferUnderrunCount
- 7.5.1.16 quickSpinBooleanNode StreamCRCCheckEnable
- 7.5.1.17 quickSpinIntegerNode StreamDefaultBufferCount
- 7.5.1.18 quickSpinIntegerNode StreamDefaultBufferCountMax
- 7.5.1.19 quickSpinEnumerationNode StreamDefaultBufferCountMode
- 7.5.1.20 quickSpinIntegerNode StreamFailedBufferCount
- 7.5.1.21 quickSpinStringNode StreamID
- 7.5.1.22 quickSpinIntegerNode StreamTotalBufferCount
- 7.5.1.23 quickSpinEnumerationNode StreamType

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

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

## 7.6 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.*

### 7.6.1 Detailed Description

Options for saving uncompressed videos.

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

### 7.6.2 Field Documentation

#### 7.6.2.1 float frameRate

Frame rate of the stream.

#### 7.6.2.2 unsigned int reserved[256]

Reserved for future use.

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

- `include/spinc/SpinnakerDefsC.h`

## 7.7 spinBMPOption Struct Reference

Options for saving BMP images.

### Data Fields

- [bool8\\_t indexedColor\\_8bit](#)
- unsigned int [reserved](#) [16]  
*Reserved for future use.*

### 7.7.1 Detailed Description

Options for saving BMP images.

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

### 7.7.2 Field Documentation

#### 7.7.2.1 `bool8_t indexedColor_8bit`

#### 7.7.2.2 `unsigned int reserved[16]`

Reserved for future use.

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

- `include/spinc/SpinnakerDefsC.h`

## 7.8 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`

### 7.8.1 Detailed Description

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

### 7.8.2 Field Documentation

7.8.2.1 `double m_blackLevel`

7.8.2.2 `int64_t m_counterValue`

7.8.2.3 `int64_t m_cRC`

7.8.2.4 `int64_t m_encoderValue`

7.8.2.5 `int64_t m_exposureEndLineStatusAll`

7.8.2.6 `double m_exposureTime`

7.8.2.7 `int64_t m_frameID`

7.8.2.8 `double m_gain`

7.8.2.9 `int64_t m_height`

7.8.2.10 `int64_t m_image`

7.8.2.11 `double m_inferenceConfidence`

7.8.2.12 `int64_t m_inferenceResult`

7.8.2.13 `int64_t m_linePitch`

7.8.2.14 `int64_t m_lineStatusAll`

- 7.8.2.15 int64\_t m\_offsetX
- 7.8.2.16 int64\_t m\_offsetY
- 7.8.2.17 int64\_t m\_partSelector
- 7.8.2.18 int64\_t m\_pixelDynamicRangeMax
- 7.8.2.19 int64\_t m\_pixelDynamicRangeMin
- 7.8.2.20 double m\_scan3dAxisMax
- 7.8.2.21 double m\_scan3dAxisMin
- 7.8.2.22 double m\_scan3dCoordinateOffset
- 7.8.2.23 double m\_scan3dCoordinateReferenceValue
- 7.8.2.24 double m\_scan3dCoordinateScale
- 7.8.2.25 double m\_scan3dInvalidDataValue
- 7.8.2.26 double m\_scan3dTransformValue
- 7.8.2.27 int64\_t m\_scanLineSelector
- 7.8.2.28 int64\_t m\_sequencerSetActive
- 7.8.2.29 int64\_t m\_serialDataLength
- 7.8.2.30 int64\_t m\_streamChannelID
- 7.8.2.31 double m\_timerValue
- 7.8.2.32 int64\_t m\_timestamp
- 7.8.2.33 int64\_t m\_timestampLatchValue
- 7.8.2.34 int64\_t m\_transferBlockID
- 7.8.2.35 int64\_t m\_transferQueueCurrentBlockCount
- 7.8.2.36 int64\_t m\_width

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

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

## 7.9 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.*

### 7.9.1 Detailed Description

Options for saving H264 videos.

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

### 7.9.2 Field Documentation

#### 7.9.2.1 unsigned int bitrate

Bitrate to encode at.

#### 7.9.2.2 float frameRate

Frame rate of the stream.

#### 7.9.2.3 unsigned int height

Height of source image.

#### 7.9.2.4 unsigned int reserved[256]

Reserved for future use.

#### 7.9.2.5 unsigned int width

Width of source image.

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

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

## 7.10 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.*

### 7.10.1 Detailed Description

Options for saving JPEG images.

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

### 7.10.2 Field Documentation

#### 7.10.2.1 bool8\_t progressive

Whether to save as a progressive JPEG file.

#### 7.10.2.2 unsigned int quality

JPEG image quality in range (0-100).

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

#### 7.10.2.3 unsigned int reserved[16]

Reserved for future use.

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

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

## 7.11 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.*

### 7.11.1 Detailed Description

Options for saving JPEG 2000 images.

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

### 7.11.2 Field Documentation

#### 7.11.2.1 unsigned int quality

JPEG saving quality in range (1-512).

#### 7.11.2.2 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 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.*

### 7.12.1 Detailed Description

Provides easier access to the current version of Spinnaker.

### 7.12.2 Field Documentation

#### 7.12.2.1 unsigned int build

Build number of the library.

#### 7.12.2.2 unsigned int major

Major version of the library.

#### 7.12.2.3 unsigned int minor

Minor version of the library.

#### 7.12.2.4 unsigned int type

Version type of the library.

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

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

## 7.13 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]

### 7.13.1 Detailed Description

Options for saving MJPG videos.

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

### 7.13.2 Field Documentation

#### 7.13.2.1 float frameRate

Frame rate of the stream.

#### 7.13.2.2 unsigned int quality

Image quality (1-100)

#### 7.13.2.3 unsigned int reserved[256]

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

- `include/spinc/SpinnakerDefsC.h`

## 7.14 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.*

### 7.14.1 Detailed Description

Options for saving PGM images.

### 7.14.2 Field Documentation

#### 7.14.2.1 bool8\_t binaryFile

Whether to save the PPM as a binary file.

#### 7.14.2.2 unsigned int reserved[16]

Reserved for future use.

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

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

## 7.15 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.*

### 7.15.1 Detailed Description

Options for saving PNG images.

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

### 7.15.2 Field Documentation

#### 7.15.2.1 unsigned int compressionLevel

Compression level (0-9).

0 is no compression, 9 is best compression.

#### 7.15.2.2 bool8\_t interlaced

Whether to save the PNG as interlaced.

#### 7.15.2.3 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 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.*

### 7.16.1 Detailed Description

Options for saving PPM images.

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

### 7.16.2 Field Documentation

#### 7.16.2.1 bool8\_t binaryFile

Whether to save the PPM as a binary file.

#### 7.16.2.2 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 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.*

### 7.17.1 Detailed Description

Options for saving TIFF images.

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

### 7.17.2 Field Documentation

#### 7.17.2.1 [spinCompressionMethod compression](#)

Compression method to use for encoding TIFF images.

#### 7.17.2.2 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

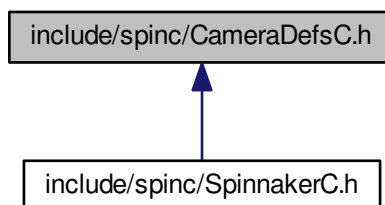
# File Documentation

### 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_GEVIIEEE1588STATUS` }
- enum `spinGevIEEE1588ModeEnums` {  
    `GevIEEE1588Mode_Auto`,  
    `GevIEEE1588Mode_SlaveOnly`,  
    `NUM_GEVIIEEE1588MODE` }
- enum `spinGevIEEE1588ClockAccuracyEnums` {  
    `GevIEEE1588ClockAccuracy_Unknown`,  
    `NUM_GEVIIEEE1588CLOCKACCURACY` }
- enum `spinGevCCPEnums` {  
    `GevCCP_OpenAccess`,  
    `GevCCP_ExclusiveAccess`,  
    `GevCCP_ControlAccess`,  
    `NUM_GEVCPP` }
- 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_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,

- 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\\_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\\_RGB16\\_Planar,](#)  
[PixelFormatInfoSelector\\_RGB565p,](#)  
[PixelFormatInfoSelector\\_BGRa8,](#)  
[PixelFormatInfoSelector\\_BGRa10,](#)  
[PixelFormatInfoSelector\\_BGRa10p,](#)  
[PixelFormatInfoSelector\\_BGRa12,](#)  
[PixelFormatInfoSelector\\_BGRa12p,](#)  
[PixelFormatInfoSelector\\_BGRa14,](#)  
[PixelFormatInfoSelector\\_BGRa16,](#)  
[PixelFormatInfoSelector\\_BGR8,](#)  
[PixelFormatInfoSelector\\_BGR10,](#)  
[PixelFormatInfoSelector\\_BGR10p,](#)

- ```
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` {

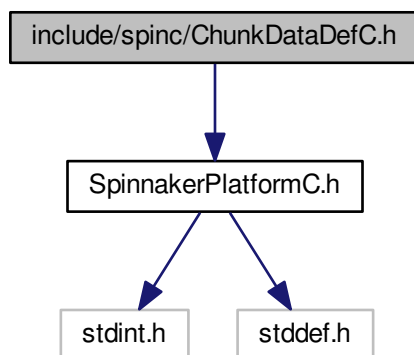
```
CxpLinkConfiguration_Auto,  
CxpLinkConfiguration_CXP1_X1,  
CxpLinkConfiguration_CXP2_X1,  
CxpLinkConfiguration_CXP3_X1,  
CxpLinkConfiguration_CXP5_X1,  
CxpLinkConfiguration_CXP6_X1,  
CxpLinkConfiguration_CXP1_X2,  
CxpLinkConfiguration_CXP2_X2,  
CxpLinkConfiguration_CXP3_X2,  
CxpLinkConfiguration_CXP5_X2,  
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 }`

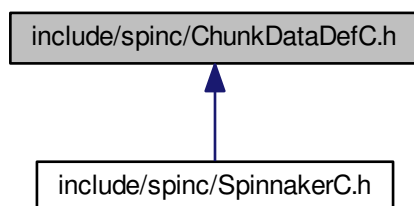
- `enum spinCxpPoCxpStatusEnums {`
 `CxpPoCxpStatus_Auto,`
 `CxpPoCxpStatus_Off,`
 `CxpPoCxpStatus_Tripped,`
 `NUM_CXPPOCXPSTATUS }`

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:



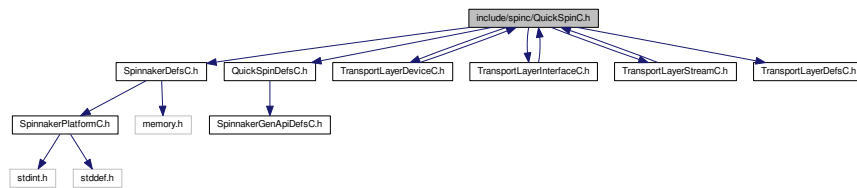
Data Structures

- struct [spinChunkData](#)

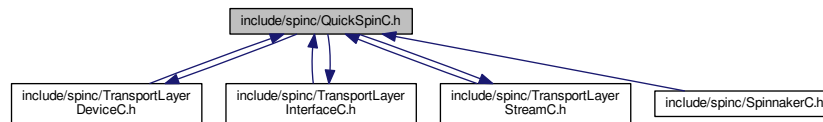
The type of information that can be obtained from image chunk data.

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:

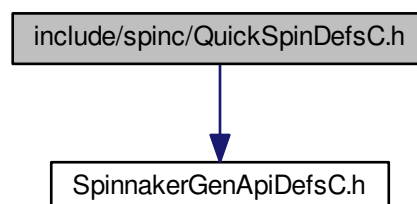


Functions

- `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)`

8.6 include/spinc/QuickSpinDefsC.h File Reference

Include dependency graph for QuickSpinDefsC.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 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](#) *pImage)
- Retrieves an image from a camera.*
- [SPINNAKERC_API spinCameraGetNextImageEx](#) ([spinCamera](#) hCamera, uint64_t grabTimeout, [spinImage](#) *pImage)
- Retrieves an image from a camera; manually set the timeout in milliseconds.*
- [SPINNAKERC_API 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 spinCameraIsValid](#) ([spinCamera](#) hCamera, bool8_t *pIsValid)
- Checks whether a camera is still valid for use.*
- [SPINNAKERC_API spinCameraIsInitialized](#) ([spinCamera](#) hCamera, bool8_t *pIsInit)
- 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 spinImageCreateEmpty](#) ([spinImage](#) *pImage)
- Creates an empty image; images created this way must be destroyed.*
- [SPINNAKERC_API spinImageCreate](#) ([spinImage](#) hSrcImage, [spinImage](#) *pDestImage)
- Creates an image from another; images created this way must be destroyed.*
- [SPINNAKERC_API spinImageCreateEx](#) ([spinImage](#) *pImage, 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](#) *pbIsIncomplete)

Checks whether an image is incomplete.
- [SPINNAKERC_API spinImageGetValidPayloadSize](#) ([spinImage](#) hImage, [size_t](#) *pSize)

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

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

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

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

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

Saves an image as an PGM image.
- [SPINNAKERC_API spinImageSaveTiff](#) ([spinImage](#) hImage, [const char](#) *pFilename, [const spinTIFFOption](#) *pOption)

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

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

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

Saves an image as a BMP image.
- [SPINNAKERC_API spinImageGetChunkLayoutID](#) ([spinImage](#) hImage, [uint64_t](#) *pId)

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

Calculates the image statistics of an image.
- [SPINNAKERC_API spinImageGetStatus](#) ([spinImage](#) hImage, [spinImageStatus](#) *pStatus)

Retrieves the image status of an image.
- [SPINNAKERC_API spinImageGetStatusDescription](#) ([spinImageStatus](#) status, [char](#) *pBuf, [size_t](#) *pBufLen)

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

Releases an image.
- [SPINNAKERC_API spinImageHasCRC](#) ([spinImage](#) hImage, [bool8_t](#) *pbHasCRC)

Checks whether an image has CRC.
- [SPINNAKERC_API spinImageCheckCRC](#) ([spinImage](#) hImage, [bool8_t](#) *pbCheckCRC)

Checks whether the CRC of an image is correct.

- [SPINNAKERC_API spinImageGetBitsPerPixel](#) ([spinImage](#) hImage, [size_t](#) *pBitsPerPixel)
Retrieves the number of bits per pixel of an image.
- [SPINNAKERC_API spinImageGetSize](#) ([spinImage](#) hImage, [size_t](#) *pImageSize)
Retrieves the size of an image.
- [SPINNAKERC_API spinImageGetStride](#) ([spinImage](#) hImage, [size_t](#) *pStride)
Retrieves the stride of an image.
- [SPINNAKERC_API 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.

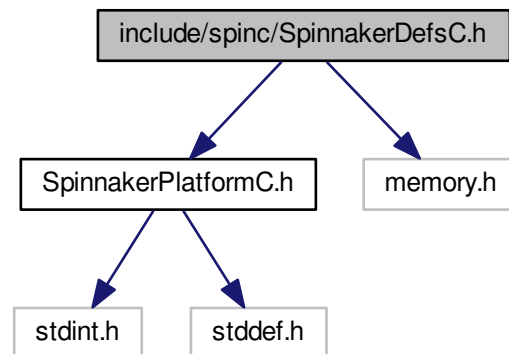
- [SPINNAKERC_API spinDeviceEventGetName](#) ([spinDeviceEventData](#) hDeviceEventData, char *pBuf, size_t *pBufLen)

Retrieves the event name of a device event.

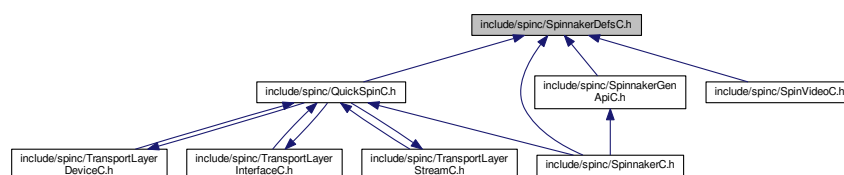
- [SPINNAKERC_API_DEPRECATED](#) ("spinAVIRecorderOpenUncompressed is deprecated, use [spinVideoOpenUncompressed](#) instead.", spinAVIRecorderOpenUncompressed([spinAVIRecorder](#) *phRecorder, const char *pName, [spinAVIOption](#) option))
- [SPINNAKERC_API_DEPRECATED](#) ("spinAVIRecorderOpenMJPEG is deprecated, use [spinVideoOpenMJPEG](#) instead.", spinAVIRecorderOpenMJPEG([spinAVIRecorder](#) *phRecorder, const char *pName, [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))
- [SPINNAKERC_API spinImageChunkDataGetIntValue](#) ([spinImage](#) hImage, const char *pName, int64_t *pValue)
- [SPINNAKERC_API spinImageChunkDataGetFloatValue](#) ([spinImage](#) hImage, const char *pName, double *pValue)

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 [spinMJPEGOption](#)
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`,
 - `EDGE_SENSING`,
 - `HQ_LINEAR`,
 - `RIGOROUS`,
 - `IPP`,
 - `DIRECTIONAL_FILTER`,
 - `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 }

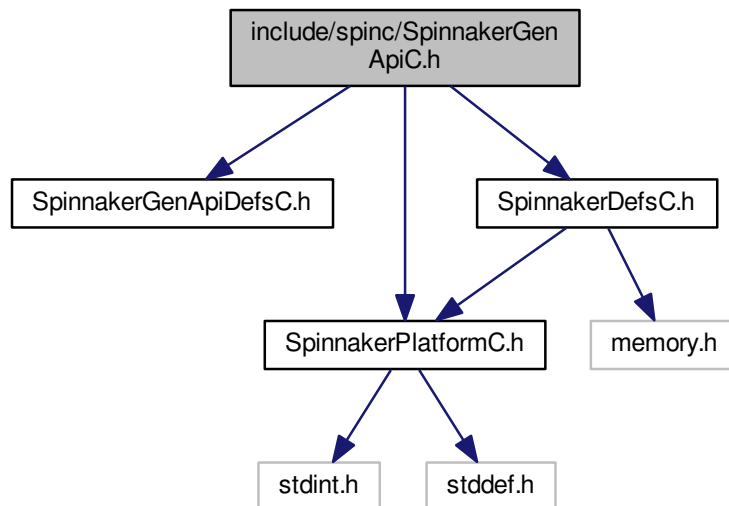
Possible Status Codes Returned from Action Command.

Variables

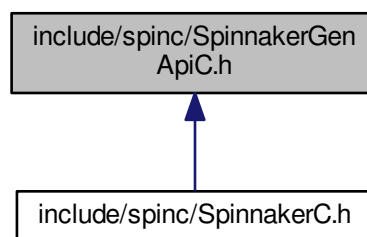
- static const `bool8_t False` = 0
- static const `bool8_t True` = 1

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

- [SPINNAKERC_API spinNodeMapGetNode](#) ([spinNodeMapHandle](#) hNodeMap, const char *pName, [spinNodeHandle](#) *phNode)
Retrieves a node from the nodemap by name.
- [SPINNAKERC_API spinNodeMapGetNumNodes](#) ([spinNodeMapHandle](#) hNodeMap, size_t *pValue)
Gets the number of nodes in the map.
- [SPINNAKERC_API spinNodeMapGetNodeByIndex](#) ([spinNodeMapHandle](#) hNodeMap, size_t index, [spinNodeHandle](#) *phNode)

- Retrieves a node from the nodemap by index.*

 - [SPINNAKERC_API 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)

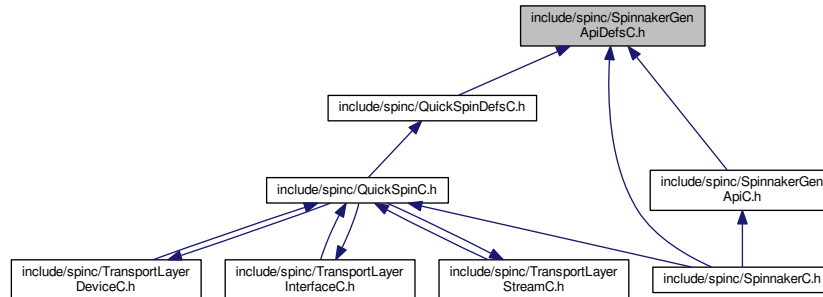
Sets the value of a register node; manually set whether to verify the node.

 - [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 alternative.
- enum `spinSlope` {
`Increasing`,
`Decreasing`,
`Varying`,
`Automatic`,

`_UndefinedESlope }`

typedef for fomula type

- enum `spinXMLValidation` {
`xvLoad` = 0x00000001L,
`xvCycles` = 0x00000002L,
`xvSFNC` = 0x00000004L,
`xvDefault` = 0x00000000L,
`xvAll` = 0xffffffffL,
`_UndefinedEXMLValidation` = 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` {
`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` }

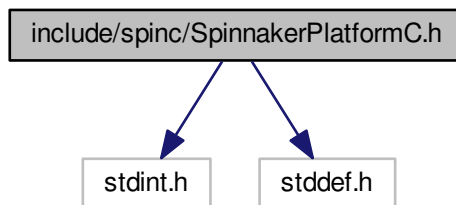
typedef for increment mode

- enum `spinInputDirection` {
`idFrom`,
`idTo`,
`idNone` }

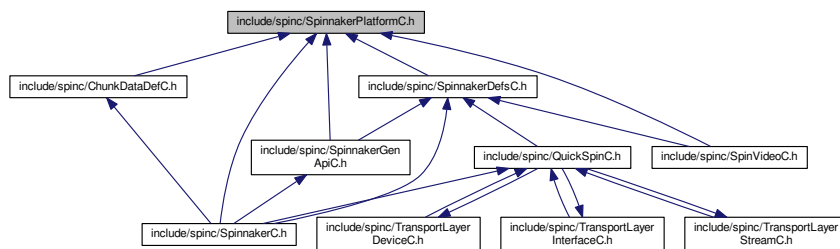
typedef for link type

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

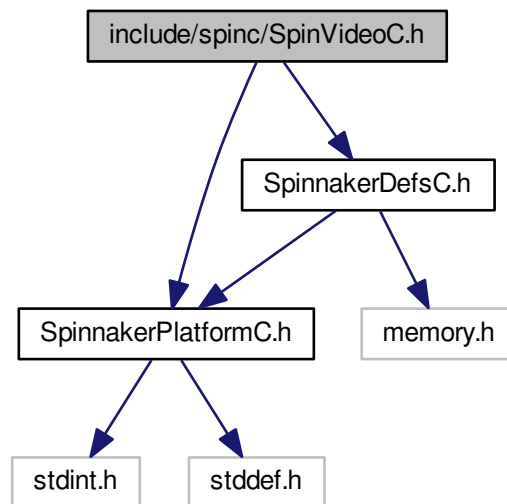
- `#define SPINNAKERC_API SPINC_IMPORT_EXPORT spinError SPINC_CALLTYPE`

8.11.1 Macro Definition Documentation

8.11.1.1 `#define SPINNAKERC_API SPINC_IMPORT_EXPORT spinError SPINC_CALLTYPE`

8.12 include/spinc/SpinVideoC.h File Reference

Include dependency graph for SpinVideoC.h:

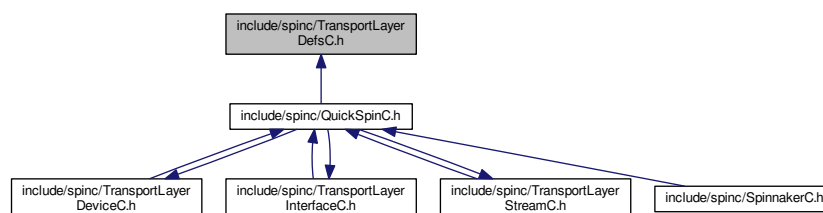


Functions

- [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:



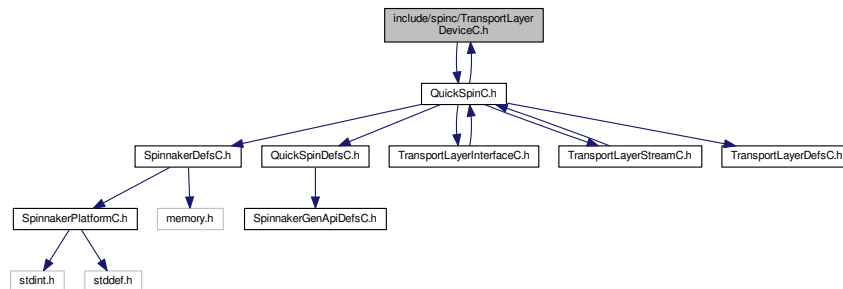
Enumerations

- 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](#) }

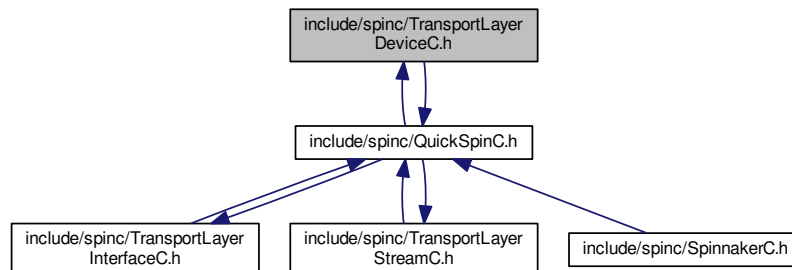
- enum [spinTLGenICamXMLLocationEnums](#) {
GenICamXMLLocation_Device,
GenICamXMLLocation_Host,
NUMGENICAMXMLLOCATION }
- enum [spinTLDeviceEndiannessMechanismEnums](#) {
DeviceEndiannessMechanism_Legacy,
DeviceEndiannessMechanism_Standard,
NUMDEVICEENDIANESSMECHANISM }
- enum [spinTLDeviceCurrentSpeedEnums](#) {
DeviceCurrentSpeed_UnknownSpeed,
DeviceCurrentSpeed_LowSpeed,
DeviceCurrentSpeed_FullSpeed,
DeviceCurrentSpeed_HighSpeed,
DeviceCurrentSpeed_SuperSpeed,
NUMDEVICECURRENTSPEED }
- enum [spinTLPOEStatusEnums](#) {
POEStatus_NotSupported,
POEStatus_PowerOff,
POEStatus_PowerOn,
NUMPOESTATUS }

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:

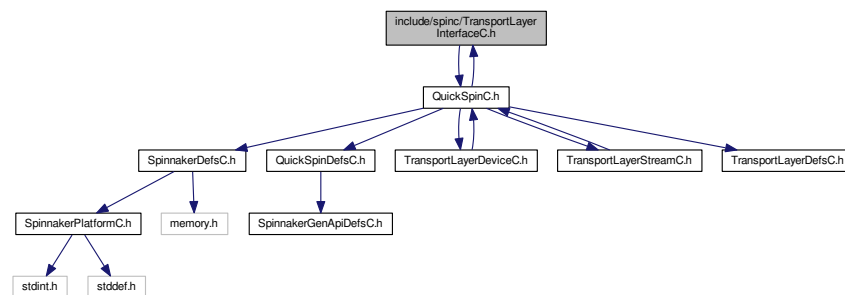


Data Structures

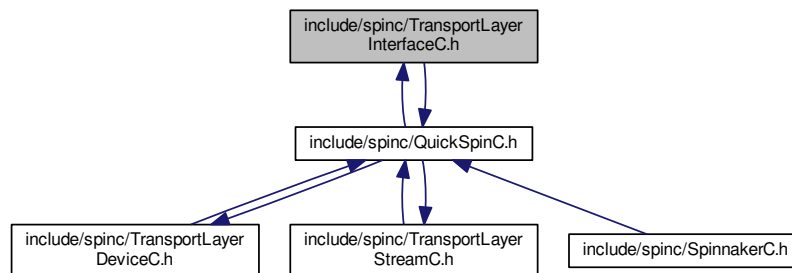
- struct [quickSpinTLDevice](#)

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:

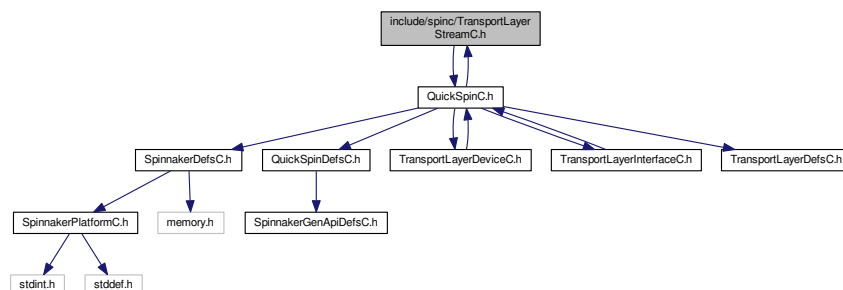


Data Structures

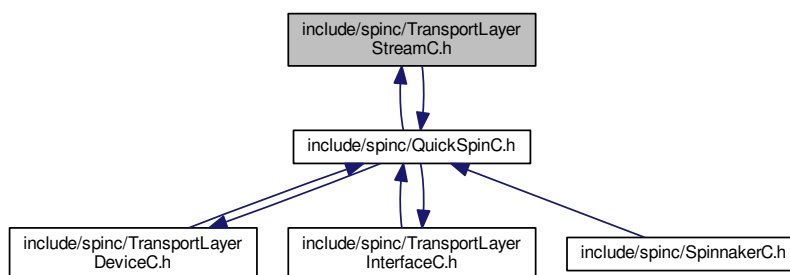
- struct [quickSpinTLInterface](#)

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](#)

Index

- [_CycleDetectAccessMode](#)
Spinnaker C GenICam Enumerations, [264](#)
 - [_UndefinedAccessMode](#)
Spinnaker C GenICam Enumerations, [264](#)
 - [_UndefinedCachingMode](#)
Spinnaker C GenICam Enumerations, [265](#)
 - [_UndefinedEDisplayNotation](#)
Spinnaker C GenICam Enumerations, [265](#)
 - [_UndefinedESlope](#)
Spinnaker C GenICam Enumerations, [268](#)
 - [_UndefinedEXMLValidation](#)
Spinnaker C GenICam Enumerations, [269](#)
 - [_UndefinedEndian](#)
Spinnaker C GenICam Enumerations, [265](#)
 - [_UndefinedNameSpace](#)
Spinnaker C GenICam Enumerations, [267](#)
 - [_UndefinedRepresentation](#)
Spinnaker C GenICam Enumerations, [267](#)
 - [_UndefinedSign](#)
Spinnaker C GenICam Enumerations, [268](#)
 - [_UndefinedStandardNameSpace](#)
Spinnaker C GenICam Enumerations, [268](#)
 - [_UndefinedVisibility](#)
Spinnaker C GenICam Enumerations, [268](#)
 - [_UndefinedYesNo](#)
Spinnaker C GenICam Enumerations, [269](#)
- [ACTION_COMMAND_STATUS_ACTION_LATE](#)
Spinnaker C Structures, [215](#)
- [ACTION_COMMAND_STATUS_ERROR](#)
Spinnaker C Structures, [215](#)
- [ACTION_COMMAND_STATUS_NO_REF_TIME](#)
Spinnaker C Structures, [215](#)
- [ACTION_COMMAND_STATUS_OVERFLOW](#)
Spinnaker C Structures, [215](#)
- [ACTION_COMMAND_STATUS_OK](#)
Spinnaker C Structures, [215](#)
- [ADOBE_DEFLATE](#)
Spinnaker C Structures, [215](#)
- [aPAUSEMACCtrlFramesReceived](#)
quickSpin, [295](#)
- [aPAUSEMACCtrlFramesTransmitted](#)
quickSpin, [295](#)
- [AVIRecorder Access](#), [198](#)
SPINNAKER_API_DEPRECATED, [198](#), [199](#)
- [AasRoiEnable](#)
quickSpin, [294](#)
- [AasRoiHeight](#)
quickSpin, [294](#)
- [AasRoiOffsetX](#)
quickSpin, [294](#)
- [AasRoiOffsetY](#)
quickSpin, [294](#)
- [AasRoiWidth](#)
quickSpin, [294](#)
- [AcquisitionAbort](#)
quickSpin, [294](#)
- [AcquisitionArm](#)
quickSpin, [294](#)
- [AcquisitionBurstFrameCount](#)
quickSpin, [294](#)
- [AcquisitionFrameCount](#)
quickSpin, [294](#)
- [AcquisitionFrameRate](#)
quickSpin, [294](#)
- [AcquisitionFrameRateEnable](#)
quickSpin, [294](#)
- [AcquisitionLineRate](#)
quickSpin, [294](#)
- [AcquisitionMode](#)
quickSpin, [294](#)
- [AcquisitionMode_Continuous](#)
Camera Enumerations, [43](#)
- [AcquisitionMode_MultiFrame](#)
Camera Enumerations, [43](#)
- [AcquisitionMode_SingleFrame](#)
Camera Enumerations, [43](#)
- [AcquisitionResultingFrameRate](#)
quickSpin, [294](#)
- [AcquisitionStart](#)
quickSpin, [294](#)
- [AcquisitionStatus](#)
quickSpin, [294](#)
- [AcquisitionStatusSelector](#)
quickSpin, [295](#)
- [AcquisitionStatusSelector_AcquisitionActive](#)
Camera Enumerations, [44](#)
- [AcquisitionStatusSelector_AcquisitionTransfer](#)
Camera Enumerations, [44](#)
- [AcquisitionStatusSelector_AcquisitionTriggerWait](#)
Camera Enumerations, [44](#)
- [AcquisitionStatusSelector_ExposureActive](#)
Camera Enumerations, [44](#)
- [AcquisitionStatusSelector_FrameActive](#)
Camera Enumerations, [44](#)
- [AcquisitionStatusSelector_FrameTriggerWait](#)
Camera Enumerations, [44](#)
- [AcquisitionStop](#)
quickSpin, [295](#)

- ActionCommand
 - quickSpinTLInterface, [326](#)
- actionCommandResult, [281](#)
 - DeviceAddress, [281](#)
 - Status, [281](#)
- actionCommandStatus
 - Spinnaker C Structures, [215](#)
- ActionDeviceKey
 - quickSpin, [295](#)
- ActionGroupKey
 - quickSpin, [295](#)
- ActionGroupMask
 - quickSpin, [295](#)
- ActionQueueSize
 - quickSpin, [295](#)
- ActionSelector
 - quickSpin, [295](#)
- ActionUnconditionalMode
 - quickSpin, [295](#)
- ActionUnconditionalMode_Off
 - Camera Enumerations, [44](#)
- ActionUnconditionalMode_On
 - Camera Enumerations, [44](#)
- AdaptiveCompressionEnable
 - quickSpin, [295](#)
- AdcBitDepth
 - quickSpin, [295](#)
- AdcBitDepth_Bit10
 - Camera Enumerations, [44](#)
- AdcBitDepth_Bit12
 - Camera Enumerations, [44](#)
- AdcBitDepth_Bit14
 - Camera Enumerations, [44](#)
- AdcBitDepth_Bit8
 - Camera Enumerations, [44](#)
- AutoAlgorithmSelector
 - quickSpin, [295](#)
- AutoAlgorithmSelector_Ae
 - Camera Enumerations, [44](#)
- AutoAlgorithmSelector_Awb
 - Camera Enumerations, [44](#)
- AutoExposureControlLoopDamping
 - quickSpin, [295](#)
- AutoExposureControlPriority
 - quickSpin, [295](#)
- AutoExposureControlPriority_ExposureTime
 - Camera Enumerations, [45](#)
- AutoExposureControlPriority_Gain
 - Camera Enumerations, [45](#)
- AutoExposureEVCompensation
 - quickSpin, [295](#)
- AutoExposureExposureTimeLowerLimit
 - quickSpin, [295](#)
- AutoExposureExposureTimeUpperLimit
 - quickSpin, [295](#)
- AutoExposureGainLowerLimit
 - quickSpin, [295](#)
- AutoExposureGainUpperLimit
 - quickSpin, [295](#)
- AutoExposureGreyValueLowerLimit
 - quickSpin, [295](#)
- AutoExposureGreyValueUpperLimit
 - quickSpin, [295](#)
- AutoExposureLightingMode
 - quickSpin, [295](#)
- AutoExposureLightingMode_AutoDetect
 - Camera Enumerations, [45](#)
- AutoExposureLightingMode_Backlight
 - Camera Enumerations, [45](#)
- AutoExposureLightingMode_Frontlight
 - Camera Enumerations, [45](#)
- AutoExposureLightingMode_Normal
 - Camera Enumerations, [45](#)
- AutoExposureMeteringMode
 - quickSpin, [296](#)
- AutoExposureMeteringMode_Average
 - Camera Enumerations, [45](#)
- AutoExposureMeteringMode_CenterWeighted
 - Camera Enumerations, [45](#)
- AutoExposureMeteringMode_HistogramPeak
 - Camera Enumerations, [45](#)
- AutoExposureMeteringMode_Partial
 - Camera Enumerations, [45](#)
- AutoExposureMeteringMode_Spot
 - Camera Enumerations, [45](#)
- AutoExposureTargetGreyValue
 - quickSpin, [296](#)
- AutoExposureTargetGreyValueAuto
 - quickSpin, [296](#)
- AutoExposureTargetGreyValueAuto_Continuous
 - Camera Enumerations, [46](#)
- AutoExposureTargetGreyValueAuto_Off
 - Camera Enumerations, [46](#)
- AutoForceIP
 - quickSpinTLInterface, [326](#)
- Automatic
 - Spinnaker C GenICam Enumerations, [268](#)
- BLUE
 - Spinnaker C Enumerations, [213](#)
- BMP
 - Spinnaker C Enumerations, [211](#)
- BalanceRatio
 - quickSpin, [296](#)
- BalanceRatioSelector
 - quickSpin, [296](#)
- BalanceRatioSelector_Blue
 - Camera Enumerations, [46](#)
- BalanceRatioSelector_Red
 - Camera Enumerations, [46](#)
- BalanceWhiteAuto
 - quickSpin, [296](#)
- BalanceWhiteAuto_Continuous
 - Camera Enumerations, [46](#)
- BalanceWhiteAuto_Off
 - Camera Enumerations, [46](#)
- BalanceWhiteAuto_Once

- Camera Enumerations, [46](#)
- BalanceWhiteAutoDamping
 - quickSpin, [296](#)
- BalanceWhiteAutoLowerLimit
 - quickSpin, [296](#)
- BalanceWhiteAutoProfile
 - quickSpin, [296](#)
- BalanceWhiteAutoProfile_Indoor
 - Camera Enumerations, [46](#)
- BalanceWhiteAutoProfile_Outdoor
 - Camera Enumerations, [46](#)
- BalanceWhiteAutoUpperLimit
 - quickSpin, [296](#)
- BaseNode
 - Spinnaker C GenICam Enumerations, [267](#)
- Beginner
 - Spinnaker C GenICam Enumerations, [268](#)
- BigEndian
 - Spinnaker C GenICam Enumerations, [265](#)
- binaryFile
 - spinPGMOption, [339](#)
 - spinPPMOption, [340](#)
- BinningHorizontal
 - quickSpin, [296](#)
- BinningHorizontalMode
 - quickSpin, [296](#)
- BinningHorizontalMode_Average
 - Camera Enumerations, [47](#)
- BinningHorizontalMode_Sum
 - Camera Enumerations, [47](#)
- BinningSelector
 - quickSpin, [296](#)
- BinningSelector_All
 - Camera Enumerations, [47](#)
- BinningSelector_ISP
 - Camera Enumerations, [47](#)
- BinningSelector_Sensor
 - Camera Enumerations, [47](#)
- BinningVertical
 - quickSpin, [296](#)
- BinningVerticalMode
 - quickSpin, [296](#)
- BinningVerticalMode_Average
 - Camera Enumerations, [47](#)
- BinningVerticalMode_Sum
 - Camera Enumerations, [47](#)
- bitrate
 - spinH264Option, [334](#)
- BlackLevel
 - quickSpin, [296](#)
- BlackLevelAuto
 - quickSpin, [296](#)
- BlackLevelAuto_Continuous
 - Camera Enumerations, [48](#)
- BlackLevelAuto_Off
 - Camera Enumerations, [48](#)
- BlackLevelAuto_Once
 - Camera Enumerations, [48](#)
- BlackLevelAutoBalance
 - quickSpin, [296](#)
- BlackLevelAutoBalance_Continuous
 - Camera Enumerations, [47](#)
- BlackLevelAutoBalance_Off
 - Camera Enumerations, [47](#)
- BlackLevelAutoBalance_Once
 - Camera Enumerations, [47](#)
- BlackLevelClampingEnable
 - quickSpin, [296](#)
- BlackLevelRaw
 - quickSpin, [296](#)
- BlackLevelSelector
 - quickSpin, [296](#)
- BlackLevelSelector_All
 - Camera Enumerations, [48](#)
- BlackLevelSelector_Analog
 - Camera Enumerations, [48](#)
- BlackLevelSelector_Digital
 - Camera Enumerations, [48](#)
- bool8_t
 - Spinnaker C Definitions, [12](#)
- Boolean
 - Spinnaker C GenICam Enumerations, [267](#)
- BooleanNode
 - Spinnaker C GenICam Enumerations, [267](#)
- build
 - spinLibraryVersion, [337](#)
- CCITTFFAX3
 - Spinnaker C Structures, [215](#)
- CCITTFFAX4
 - Spinnaker C Structures, [215](#)
- Camera Access, [145](#)
 - spinCameraBeginAcquisition, [146](#)
 - spinCameraDelInit, [146](#)
 - spinCameraEndAcquisition, [147](#)
 - spinCameraGetAccessMode, [147](#)
 - spinCameraGetGuiXml, [147](#)
 - spinCameraGetNextImage, [148](#)
 - spinCameraGetNextImageEx, [148](#)
 - spinCameraGetNodeMap, [149](#)
 - spinCameraGetTLDeviceNodeMap, [149](#)
 - spinCameraGetTLStreamNodeMap, [149](#)
 - spinCameraGetUniqueID, [150](#)
 - spinCameraInit, [150](#)
 - spinCameraIsInitialized, [151](#)
 - spinCameraIsStreaming, [151](#)
 - spinCameraIsValid, [151](#)
 - spinCameraReadPort, [152](#)
 - spinCameraRegisterDeviceEvent, [152](#)
 - spinCameraRegisterDeviceEventEx, [152](#)
 - spinCameraRegisterImageEvent, [153](#)
 - spinCameraRelease, [153](#)
 - spinCameraUnregisterDeviceEvent, [153](#)
 - spinCameraUnregisterImageEvent, [154](#)
 - spinCameraWritePort, [154](#)
- Camera Enumerations, [13](#)
 - AcquisitionMode_Continuous, [43](#)

- AcquisitionMode_MultiFrame, 43
- AcquisitionMode_SingleFrame, 43
- AcquisitionStatusSelector_AcquisitionActive, 44
- AcquisitionStatusSelector_AcquisitionTransfer, 44
- AcquisitionStatusSelector_AcquisitionTriggerWait, 44
- AcquisitionStatusSelector_ExposureActive, 44
- AcquisitionStatusSelector_FrameActive, 44
- AcquisitionStatusSelector_FrameTriggerWait, 44
- ActionUnconditionalMode_Off, 44
- ActionUnconditionalMode_On, 44
- AdcBitDepth_Bit10, 44
- AdcBitDepth_Bit12, 44
- AdcBitDepth_Bit14, 44
- AdcBitDepth_Bit8, 44
- AutoAlgorithmSelector_Ae, 44
- AutoAlgorithmSelector_Awb, 44
- AutoExposureControlPriority_ExposureTime, 45
- AutoExposureControlPriority_Gain, 45
- AutoExposureLightingMode_AutoDetect, 45
- AutoExposureLightingMode_Backlight, 45
- AutoExposureLightingMode_Frontlight, 45
- AutoExposureLightingMode_Normal, 45
- AutoExposureMeteringMode_Average, 45
- AutoExposureMeteringMode_CenterWeighted, 45
- AutoExposureMeteringMode_HistogramPeak, 45
- AutoExposureMeteringMode_Partial, 45
- AutoExposureMeteringMode_Spot, 45
- AutoExposureTargetGreyValueAuto_Continuous, 46
- AutoExposureTargetGreyValueAuto_Off, 46
- BalanceRatioSelector_Blue, 46
- BalanceRatioSelector_Red, 46
- BalanceWhiteAuto_Continuous, 46
- BalanceWhiteAuto_Off, 46
- BalanceWhiteAuto_Once, 46
- BalanceWhiteAutoProfile_Indoor, 46
- BalanceWhiteAutoProfile_Outdoor, 46
- BinningHorizontalMode_Average, 47
- BinningHorizontalMode_Sum, 47
- BinningSelector_All, 47
- BinningSelector_ISP, 47
- BinningSelector_Sensor, 47
- BinningVerticalMode_Average, 47
- BinningVerticalMode_Sum, 47
- BlackLevelAuto_Continuous, 48
- BlackLevelAuto_Off, 48
- BlackLevelAuto_Once, 48
- BlackLevelAutoBalance_Continuous, 47
- BlackLevelAutoBalance_Off, 47
- BlackLevelAutoBalance_Once, 47
- BlackLevelSelector_All, 48
- BlackLevelSelector_Analog, 48
- BlackLevelSelector_Digital, 48
- ChunkBlackLevelSelector_All, 48
- ChunkCounterSelector_Counter0, 48
- ChunkCounterSelector_Counter1, 48
- ChunkCounterSelector_Counter2, 48
- ChunkEncoderSelector_Encoder0, 49
- ChunkEncoderSelector_Encoder1, 49
- ChunkEncoderSelector_Encoder2, 49
- ChunkEncoderStatus_EncoderDown, 49
- ChunkEncoderStatus_EncoderIdle, 49
- ChunkEncoderStatus_EncoderStatic, 49
- ChunkEncoderStatus_EncoderUp, 49
- ChunkExposureTimeSelector_Blue, 49
- ChunkExposureTimeSelector_Common, 49
- ChunkExposureTimeSelector_Cyan, 49
- ChunkExposureTimeSelector_Green, 49
- ChunkExposureTimeSelector_Infrared, 49
- ChunkExposureTimeSelector_Magenta, 49
- ChunkExposureTimeSelector_Red, 49
- ChunkExposureTimeSelector_Stage1, 49
- ChunkExposureTimeSelector_Stage2, 49
- ChunkExposureTimeSelector_Ultraviolet, 49
- ChunkExposureTimeSelector_Yellow, 49
- ChunkGainSelector_All, 50
- ChunkGainSelector_Blue, 50
- ChunkGainSelector_Green, 50
- ChunkGainSelector_Red, 50
- ChunkImageComponent_Color, 50
- ChunkImageComponent_Confidence, 50
- ChunkImageComponent_Disparity, 50
- ChunkImageComponent_Infrared, 50
- ChunkImageComponent_Intensity, 50
- ChunkImageComponent_Range, 50
- ChunkImageComponent_Scatter, 50
- ChunkImageComponent_Ultraviolet, 50
- ChunkPixelFormat_BayerBG8, 50
- ChunkPixelFormat_BayerGB8, 50
- ChunkPixelFormat_BayerGR8, 50
- ChunkPixelFormat_BayerRG8, 50
- ChunkPixelFormat_Mono12Packed, 50
- ChunkPixelFormat_Mono16, 50
- ChunkPixelFormat_Mono8, 50
- ChunkPixelFormat_RGB8Packed, 50
- ChunkPixelFormat_YCbCr601_422_8_CbYCrY, 50
- ChunkPixelFormat_YUV422Packed, 50
- ChunkRegionID_Region0, 51
- ChunkRegionID_Region1, 51
- ChunkRegionID_Region2, 51
- ChunkScan3dCoordinateReferenceSelector_↔
RotationX, 51
- ChunkScan3dCoordinateReferenceSelector_↔
RotationY, 51
- ChunkScan3dCoordinateReferenceSelector_↔
RotationZ, 51
- ChunkScan3dCoordinateReferenceSelector_↔
TranslationX, 51
- ChunkScan3dCoordinateReferenceSelector_↔
TranslationY, 51
- ChunkScan3dCoordinateReferenceSelector_↔
TranslationZ, 51
- ChunkScan3dCoordinateSelector_CoordinateA, 51

- ChunkScan3dCoordinateSelector_CoordinateB, [51](#)
- ChunkScan3dCoordinateSelector_CoordinateC, [51](#)
- ChunkScan3dCoordinateSystem_Cartesian, [51](#)
- ChunkScan3dCoordinateSystem_Cylindrical, [51](#)
- ChunkScan3dCoordinateSystem_Spherical, [51](#)
- ChunkScan3dCoordinateSystemReference_↔
Anchor, [52](#)
- ChunkScan3dCoordinateSystemReference_↔
Transformed, [52](#)
- ChunkScan3dCoordinateTransformSelector_↔
RotationX, [52](#)
- ChunkScan3dCoordinateTransformSelector_↔
RotationY, [52](#)
- ChunkScan3dCoordinateTransformSelector_↔
RotationZ, [52](#)
- ChunkScan3dCoordinateTransformSelector_↔
TranslationX, [52](#)
- ChunkScan3dCoordinateTransformSelector_↔
TranslationY, [52](#)
- ChunkScan3dCoordinateTransformSelector_↔
TranslationZ, [52](#)
- ChunkScan3dDistanceUnit_Inch, [52](#)
- ChunkScan3dDistanceUnit_Millimeter, [52](#)
- ChunkScan3dOutputMode_CalibratedABC_Grid, [53](#)
- ChunkScan3dOutputMode_CalibratedABC_↔
PointCloud, [53](#)
- ChunkScan3dOutputMode_CalibratedAC_↔
Linescan, [53](#)
- ChunkScan3dOutputMode_CalibratedAC, [53](#)
- ChunkScan3dOutputMode_CalibratedC_Linescan, [53](#)
- ChunkScan3dOutputMode_CalibratedC, [53](#)
- ChunkScan3dOutputMode_DisparityC_Linescan, [53](#)
- ChunkScan3dOutputMode_DisparityC, [53](#)
- ChunkScan3dOutputMode_RectifiedC_Linescan, [53](#)
- ChunkScan3dOutputMode_RectifiedC, [53](#)
- ChunkScan3dOutputMode_UncalibratedC, [53](#)
- ChunkSelector_BlackLevel, [54](#)
- ChunkSelector_CRC, [53](#)
- ChunkSelector_ExposureEndLineStatusAll, [54](#)
- ChunkSelector_ExposureTime, [53](#)
- ChunkSelector_FrameID, [53](#)
- ChunkSelector_Gain, [54](#)
- ChunkSelector_Height, [53](#)
- ChunkSelector_Image, [53](#)
- ChunkSelector_OffsetX, [53](#)
- ChunkSelector_OffsetY, [53](#)
- ChunkSelector_PixelFormat, [54](#)
- ChunkSelector_SequencerSetActive, [54](#)
- ChunkSelector_SerialData, [54](#)
- ChunkSelector_Timestamp, [54](#)
- ChunkSelector_Width, [53](#)
- ChunkSourceID_Source0, [54](#)
- ChunkSourceID_Source1, [54](#)
- ChunkSourceID_Source2, [54](#)
- ChunkTimerSelector_Timer0, [54](#)
- ChunkTimerSelector_Timer1, [54](#)
- ChunkTimerSelector_Timer2, [54](#)
- ChunkTransferStreamID_Stream0, [54](#)
- ChunkTransferStreamID_Stream1, [54](#)
- ChunkTransferStreamID_Stream2, [54](#)
- ChunkTransferStreamID_Stream3, [54](#)
- CIConfiguration_Base, [55](#)
- CIConfiguration_DualBase, [55](#)
- CIConfiguration_EightyBit, [55](#)
- CIConfiguration_Full, [55](#)
- CIConfiguration_Medium, [55](#)
- CITimeSlotsCount_One, [55](#)
- CITimeSlotsCount_Three, [55](#)
- CITimeSlotsCount_Two, [55](#)
- ColorTransformationSelector_RGBtoRGB, [55](#)
- ColorTransformationSelector_RGBtoYUV, [55](#)
- ColorTransformationValueSelector_Gain00, [56](#)
- ColorTransformationValueSelector_Gain01, [56](#)
- ColorTransformationValueSelector_Gain02, [56](#)
- ColorTransformationValueSelector_Gain10, [56](#)
- ColorTransformationValueSelector_Gain11, [56](#)
- ColorTransformationValueSelector_Gain12, [56](#)
- ColorTransformationValueSelector_Gain20, [56](#)
- ColorTransformationValueSelector_Gain21, [56](#)
- ColorTransformationValueSelector_Gain22, [56](#)
- ColorTransformationValueSelector_Offset0, [56](#)
- ColorTransformationValueSelector_Offset1, [56](#)
- ColorTransformationValueSelector_Offset2, [56](#)
- CounterEventActivation_AnyEdge, [56](#)
- CounterEventActivation_FallingEdge, [56](#)
- CounterEventActivation_LevelHigh, [56](#)
- CounterEventActivation_LevelLow, [56](#)
- CounterEventActivation_RisingEdge, [56](#)
- CounterEventSource_Counter0End, [57](#)
- CounterEventSource_Counter0Start, [57](#)
- CounterEventSource_Counter1End, [57](#)
- CounterEventSource_Counter1Start, [57](#)
- CounterEventSource_ExposureEnd, [57](#)
- CounterEventSource_ExposureStart, [57](#)
- CounterEventSource_FrameTriggerWait, [57](#)
- CounterEventSource_Line0, [56](#)
- CounterEventSource_Line1, [56](#)
- CounterEventSource_Line2, [56](#)
- CounterEventSource_Line3, [56](#)
- CounterEventSource_LogicBlock0, [57](#)
- CounterEventSource_LogicBlock1, [57](#)
- CounterEventSource_MHzTick, [56](#)
- CounterEventSource_Off, [56](#)
- CounterEventSource_UserOutput0, [56](#)
- CounterEventSource_UserOutput1, [56](#)
- CounterEventSource_UserOutput2, [57](#)
- CounterEventSource_UserOutput3, [57](#)
- CounterResetActivation_AnyEdge, [57](#)
- CounterResetActivation_FallingEdge, [57](#)
- CounterResetActivation_LevelHigh, [57](#)

- CounterResetActivation_LevelLow, 57
- CounterResetActivation_RisingEdge, 57
- CounterResetSource_Counter0End, 57
- CounterResetSource_Counter0Start, 57
- CounterResetSource_Counter1End, 57
- CounterResetSource_Counter1Start, 57
- CounterResetSource_ExposureEnd, 57
- CounterResetSource_ExposureStart, 57
- CounterResetSource_FrameTriggerWait, 57
- CounterResetSource_Line0, 57
- CounterResetSource_Line1, 57
- CounterResetSource_Line2, 57
- CounterResetSource_Line3, 57
- CounterResetSource_LogicBlock0, 57
- CounterResetSource_LogicBlock1, 57
- CounterResetSource_Off, 57
- CounterResetSource_UserOutput0, 57
- CounterResetSource_UserOutput1, 57
- CounterResetSource_UserOutput2, 57
- CounterResetSource_UserOutput3, 57
- CounterSelector_Counter0, 58
- CounterSelector_Counter1, 58
- CounterStatus_CounterActive, 58
- CounterStatus_CounterCompleted, 58
- CounterStatus_CounterIdle, 58
- CounterStatus_CounterOverflow, 58
- CounterStatus_CounterTriggerWait, 58
- CounterTriggerActivation_AnyEdge, 58
- CounterTriggerActivation_FallingEdge, 58
- CounterTriggerActivation_LevelHigh, 58
- CounterTriggerActivation_LevelLow, 58
- CounterTriggerActivation_RisingEdge, 58
- CounterTriggerSource_Counter0End, 59
- CounterTriggerSource_Counter0Start, 59
- CounterTriggerSource_Counter1End, 59
- CounterTriggerSource_Counter1Start, 59
- CounterTriggerSource_ExposureEnd, 59
- CounterTriggerSource_ExposureStart, 59
- CounterTriggerSource_FrameTriggerWait, 59
- CounterTriggerSource_Line0, 59
- CounterTriggerSource_Line1, 59
- CounterTriggerSource_Line2, 59
- CounterTriggerSource_Line3, 59
- CounterTriggerSource_LogicBlock0, 59
- CounterTriggerSource_LogicBlock1, 59
- CounterTriggerSource_Off, 59
- CounterTriggerSource_UserOutput0, 59
- CounterTriggerSource_UserOutput1, 59
- CounterTriggerSource_UserOutput2, 59
- CounterTriggerSource_UserOutput3, 59
- CxpConnectionTestMode_Mode1, 59
- CxpConnectionTestMode_Off, 59
- CxpLinkConfiguration_Auto, 59
- CxpLinkConfiguration_CXP1_X1, 59
- CxpLinkConfiguration_CXP1_X2, 60
- CxpLinkConfiguration_CXP1_X3, 60
- CxpLinkConfiguration_CXP1_X4, 60
- CxpLinkConfiguration_CXP1_X5, 60
- CxpLinkConfiguration_CXP1_X6, 60
- CxpLinkConfiguration_CXP2_X1, 59
- CxpLinkConfiguration_CXP2_X2, 60
- CxpLinkConfiguration_CXP2_X3, 60
- CxpLinkConfiguration_CXP2_X4, 60
- CxpLinkConfiguration_CXP2_X5, 60
- CxpLinkConfiguration_CXP2_X6, 60
- CxpLinkConfiguration_CXP3_X1, 60
- CxpLinkConfiguration_CXP3_X2, 60
- CxpLinkConfiguration_CXP3_X3, 60
- CxpLinkConfiguration_CXP3_X4, 60
- CxpLinkConfiguration_CXP3_X5, 60
- CxpLinkConfiguration_CXP3_X6, 60
- CxpLinkConfiguration_CXP5_X1, 60
- CxpLinkConfiguration_CXP5_X2, 60
- CxpLinkConfiguration_CXP5_X3, 60
- CxpLinkConfiguration_CXP5_X4, 60
- CxpLinkConfiguration_CXP5_X5, 60
- CxpLinkConfiguration_CXP5_X6, 60
- CxpLinkConfiguration_CXP6_X1, 60
- CxpLinkConfiguration_CXP6_X2, 60
- CxpLinkConfiguration_CXP6_X3, 60
- CxpLinkConfiguration_CXP6_X4, 60
- CxpLinkConfiguration_CXP6_X5, 60
- CxpLinkConfiguration_CXP6_X6, 60
- CxpLinkConfigurationPreferred_CXP1_X1, 60
- CxpLinkConfigurationPreferred_CXP1_X2, 61
- CxpLinkConfigurationPreferred_CXP1_X3, 61
- CxpLinkConfigurationPreferred_CXP1_X4, 61
- CxpLinkConfigurationPreferred_CXP1_X5, 61
- CxpLinkConfigurationPreferred_CXP1_X6, 61
- CxpLinkConfigurationPreferred_CXP2_X1, 60
- CxpLinkConfigurationPreferred_CXP2_X2, 61
- CxpLinkConfigurationPreferred_CXP2_X3, 61
- CxpLinkConfigurationPreferred_CXP2_X4, 61
- CxpLinkConfigurationPreferred_CXP2_X5, 61
- CxpLinkConfigurationPreferred_CXP2_X6, 61
- CxpLinkConfigurationPreferred_CXP3_X1, 60
- CxpLinkConfigurationPreferred_CXP3_X2, 61
- CxpLinkConfigurationPreferred_CXP3_X3, 61
- CxpLinkConfigurationPreferred_CXP3_X4, 61
- CxpLinkConfigurationPreferred_CXP3_X5, 61
- CxpLinkConfigurationPreferred_CXP3_X6, 61
- CxpLinkConfigurationPreferred_CXP5_X1, 60
- CxpLinkConfigurationPreferred_CXP5_X2, 61
- CxpLinkConfigurationPreferred_CXP5_X3, 61
- CxpLinkConfigurationPreferred_CXP5_X4, 61
- CxpLinkConfigurationPreferred_CXP5_X5, 61
- CxpLinkConfigurationPreferred_CXP5_X6, 61
- CxpLinkConfigurationPreferred_CXP6_X1, 61
- CxpLinkConfigurationPreferred_CXP6_X2, 61
- CxpLinkConfigurationPreferred_CXP6_X3, 61
- CxpLinkConfigurationPreferred_CXP6_X4, 61
- CxpLinkConfigurationPreferred_CXP6_X5, 61
- CxpLinkConfigurationPreferred_CXP6_X6, 61
- CxpLinkConfigurationStatus_CXP1_X1, 61
- CxpLinkConfigurationStatus_CXP1_X2, 61
- CxpLinkConfigurationStatus_CXP1_X3, 62

- CxpLinkConfigurationStatus_CXP1_X4, [62](#)
- CxpLinkConfigurationStatus_CXP1_X5, [62](#)
- CxpLinkConfigurationStatus_CXP1_X6, [62](#)
- CxpLinkConfigurationStatus_CXP2_X1, [61](#)
- CxpLinkConfigurationStatus_CXP2_X2, [61](#)
- CxpLinkConfigurationStatus_CXP2_X3, [62](#)
- CxpLinkConfigurationStatus_CXP2_X4, [62](#)
- CxpLinkConfigurationStatus_CXP2_X5, [62](#)
- CxpLinkConfigurationStatus_CXP2_X6, [62](#)
- CxpLinkConfigurationStatus_CXP3_X1, [61](#)
- CxpLinkConfigurationStatus_CXP3_X2, [62](#)
- CxpLinkConfigurationStatus_CXP3_X3, [62](#)
- CxpLinkConfigurationStatus_CXP3_X4, [62](#)
- CxpLinkConfigurationStatus_CXP3_X5, [62](#)
- CxpLinkConfigurationStatus_CXP3_X6, [62](#)
- CxpLinkConfigurationStatus_CXP5_X1, [61](#)
- CxpLinkConfigurationStatus_CXP5_X2, [62](#)
- CxpLinkConfigurationStatus_CXP5_X3, [62](#)
- CxpLinkConfigurationStatus_CXP5_X4, [62](#)
- CxpLinkConfigurationStatus_CXP5_X5, [62](#)
- CxpLinkConfigurationStatus_CXP5_X6, [62](#)
- CxpLinkConfigurationStatus_CXP6_X1, [61](#)
- CxpLinkConfigurationStatus_CXP6_X2, [62](#)
- CxpLinkConfigurationStatus_CXP6_X3, [62](#)
- CxpLinkConfigurationStatus_CXP6_X4, [62](#)
- CxpLinkConfigurationStatus_CXP6_X5, [62](#)
- CxpLinkConfigurationStatus_CXP6_X6, [62](#)
- CxpLinkConfigurationStatus_None, [61](#)
- CxpLinkConfigurationStatus_Pending, [61](#)
- CxpPoCxpStatus_Auto, [62](#)
- CxpPoCxpStatus_Off, [62](#)
- CxpPoCxpStatus_Tripped, [62](#)
- DecimationHorizontalMode_Discard, [62](#)
- DecimationSelector_All, [63](#)
- DecimationSelector_Sensor, [63](#)
- DecimationVerticalMode_Discard, [63](#)
- DefectCorrectionMode_Average, [63](#)
- DefectCorrectionMode_Highlight, [63](#)
- DefectCorrectionMode_Zero, [63](#)
- Deinterlacing_LineDuplication, [63](#)
- Deinterlacing_Off, [63](#)
- Deinterlacing_Weave, [63](#)
- DeviceCharacterSet_ASCII, [64](#)
- DeviceCharacterSet_UTF8, [64](#)
- DeviceClockSelector_CameraLink, [64](#)
- DeviceClockSelector_Sensor, [64](#)
- DeviceClockSelector_SensorDigitization, [64](#)
- DeviceConnectionStatus_Active, [64](#)
- DeviceConnectionStatus_Inactive, [64](#)
- DeviceIndicatorMode_Active, [64](#)
- DeviceIndicatorMode_ErrorStatus, [64](#)
- DeviceIndicatorMode_Inactive, [64](#)
- DeviceLinkHeartbeatMode_Off, [65](#)
- DeviceLinkHeartbeatMode_On, [65](#)
- DeviceLinkThroughputLimitMode_Off, [65](#)
- DeviceLinkThroughputLimitMode_On, [65](#)
- DevicePowerSupplySelector_External, [65](#)
- DeviceRegistersEndianness_Big, [65](#)
- DeviceRegistersEndianness_Little, [65](#)
- DeviceScanType_Areascan, [65](#)
- DeviceSerialPortBaudRate_Baud115200, [66](#)
- DeviceSerialPortBaudRate_Baud19200, [66](#)
- DeviceSerialPortBaudRate_Baud230400, [66](#)
- DeviceSerialPortBaudRate_Baud38400, [66](#)
- DeviceSerialPortBaudRate_Baud460800, [66](#)
- DeviceSerialPortBaudRate_Baud57600, [66](#)
- DeviceSerialPortBaudRate_Baud921600, [66](#)
- DeviceSerialPortBaudRate_Baud9600, [66](#)
- DeviceSerialPortSelector_CameraLink, [66](#)
- DeviceStreamChannelEndianness_Big, [66](#)
- DeviceStreamChannelEndianness_Little, [66](#)
- DeviceStreamChannelType_Receiver, [66](#)
- DeviceStreamChannelType_Transmitter, [66](#)
- DeviceTLType_CameraLink, [68](#)
- DeviceTLType_CameraLinkHS, [68](#)
- DeviceTLType_CoaXPress, [68](#)
- DeviceTLType_Custom, [68](#)
- DeviceTLType_GigEVision, [68](#)
- DeviceTLType_USB3Vision, [68](#)
- DeviceTapGeometry_Geometry_10X_1Y, [68](#)
- DeviceTapGeometry_Geometry_10X, [68](#)
- DeviceTapGeometry_Geometry_1X10, [68](#)
- DeviceTapGeometry_Geometry_1X10_1Y, [68](#)
- DeviceTapGeometry_Geometry_1X2, [67](#)
- DeviceTapGeometry_Geometry_1X2_1Y2, [67](#)
- DeviceTapGeometry_Geometry_1X2_1Y, [67](#)
- DeviceTapGeometry_Geometry_1X2_2YE, [67](#)
- DeviceTapGeometry_Geometry_1X3, [67](#)
- DeviceTapGeometry_Geometry_1X3_1Y, [67](#)
- DeviceTapGeometry_Geometry_1X4, [67](#)
- DeviceTapGeometry_Geometry_1X4_1Y, [67](#)
- DeviceTapGeometry_Geometry_1X8, [67](#)
- DeviceTapGeometry_Geometry_1X8_1Y, [67](#)
- DeviceTapGeometry_Geometry_1X_1Y2, [67](#)
- DeviceTapGeometry_Geometry_1X_1Y, [67](#)
- DeviceTapGeometry_Geometry_1X_2YE, [67](#)
- DeviceTapGeometry_Geometry_1X, [67](#)
- DeviceTapGeometry_Geometry_2X2, [67](#)
- DeviceTapGeometry_Geometry_2X2_1Y, [67](#)
- DeviceTapGeometry_Geometry_2X2E_1Y↔
Geometry_2X2M_1Y, [67](#)
- DeviceTapGeometry_Geometry_2X2E_2YE, [67](#)
- DeviceTapGeometry_Geometry_2X2E, [67](#)
- DeviceTapGeometry_Geometry_2X2M, [67](#)
- DeviceTapGeometry_Geometry_2X_1Y2Geometry↔
_2XE_1Y, [67](#)
- DeviceTapGeometry_Geometry_2X_1Y, [67](#)
- DeviceTapGeometry_Geometry_2X_2YE, [67](#)
- DeviceTapGeometry_Geometry_2XE_1Y2, [67](#)
- DeviceTapGeometry_Geometry_2XE_2YE, [67](#)
- DeviceTapGeometry_Geometry_2XM_1Y2, [67](#)
- DeviceTapGeometry_Geometry_2XM_1Y, [67](#)
- DeviceTapGeometry_Geometry_2XM_2YE, [67](#)
- DeviceTapGeometry_Geometry_2XE, [67](#)
- DeviceTapGeometry_Geometry_2XM, [67](#)
- DeviceTapGeometry_Geometry_2X, [67](#)

- DeviceTapGeometry_Geometry_3X_1Y, 67
- DeviceTapGeometry_Geometry_3X, 67
- DeviceTapGeometry_Geometry_4X2, 67
- DeviceTapGeometry_Geometry_4X2_1Y, 67
- DeviceTapGeometry_Geometry_4X2E_1Y, 68
- DeviceTapGeometry_Geometry_4X2E, 68
- DeviceTapGeometry_Geometry_4X_1Y, 67
- DeviceTapGeometry_Geometry_4X, 67
- DeviceTapGeometry_Geometry_8X_1Y, 67
- DeviceTapGeometry_Geometry_8X, 67
- DeviceTemperatureSelector_Sensor, 68
- DeviceType_Peripheral, 68
- DeviceType_Receiver, 68
- DeviceType_Transceiver, 68
- DeviceType_Transmitter, 68
- EncoderMode_FourPhase, 69
- EncoderMode_HighResolution, 69
- EncoderOutputMode_DirectionDown, 69
- EncoderOutputMode_DirectionUp, 69
- EncoderOutputMode_Motion, 69
- EncoderOutputMode_Off, 69
- EncoderOutputMode_PositionDown, 69
- EncoderOutputMode_PositionUp, 69
- EncoderResetActivation_AnyEdge, 69
- EncoderResetActivation_FallingEdge, 69
- EncoderResetActivation_LevelHigh, 69
- EncoderResetActivation_LevelLow, 69
- EncoderResetActivation_RisingEdge, 69
- EncoderResetSource_AcquisitionEnd, 70
- EncoderResetSource_AcquisitionStart, 70
- EncoderResetSource_AcquisitionTrigger, 70
- EncoderResetSource_Action0, 70
- EncoderResetSource_Action1, 70
- EncoderResetSource_Action2, 70
- EncoderResetSource_Counter0End, 70
- EncoderResetSource_Counter0Start, 70
- EncoderResetSource_Counter1End, 70
- EncoderResetSource_Counter1Start, 70
- EncoderResetSource_Counter2End, 70
- EncoderResetSource_Counter2Start, 70
- EncoderResetSource_ExposureEnd, 70
- EncoderResetSource_ExposureStart, 70
- EncoderResetSource_FrameEnd, 70
- EncoderResetSource_FrameStart, 70
- EncoderResetSource_FrameTrigger, 70
- EncoderResetSource_Line0, 70
- EncoderResetSource_Line1, 70
- EncoderResetSource_Line2, 70
- EncoderResetSource_LinkTrigger0, 70
- EncoderResetSource_LinkTrigger1, 70
- EncoderResetSource_LinkTrigger2, 70
- EncoderResetSource_Off, 70
- EncoderResetSource_SoftwareSignal0, 70
- EncoderResetSource_SoftwareSignal1, 70
- EncoderResetSource_SoftwareSignal2, 70
- EncoderResetSource_Timer0End, 70
- EncoderResetSource_Timer0Start, 70
- EncoderResetSource_Timer1End, 70
- EncoderResetSource_Timer1Start, 70
- EncoderResetSource_Timer2End, 70
- EncoderResetSource_Timer2Start, 70
- EncoderResetSource_UserOutput0, 70
- EncoderResetSource_UserOutput1, 70
- EncoderResetSource_UserOutput2, 70
- EncoderSelector_Encoder0, 71
- EncoderSelector_Encoder1, 71
- EncoderSelector_Encoder2, 71
- EncoderSourceA_Line0, 71
- EncoderSourceA_Line1, 71
- EncoderSourceA_Line2, 71
- EncoderSourceA_Off, 71
- EncoderSourceB_Line0, 71
- EncoderSourceB_Line1, 71
- EncoderSourceB_Line2, 71
- EncoderSourceB_Off, 71
- EncoderStatus_EncoderDown, 71
- EncoderStatus_EncoderIdle, 71
- EncoderStatus_EncoderStatic, 71
- EncoderStatus_EncoderUp, 71
- EventNotification_Off, 72
- EventNotification_On, 72
- EventSelector_Error, 72
- EventSelector_ExposureEnd, 72
- EventSelector_SerialPortReceive, 72
- ExposureActiveMode_AllPixels, 72
- ExposureActiveMode_AnyPixels, 72
- ExposureActiveMode_Line1, 72
- ExposureAuto_Continuous, 72
- ExposureAuto_Off, 72
- ExposureAuto_Once, 72
- ExposureMode_Timed, 73
- ExposureMode_TriggerWidth, 73
- ExposureTimeMode_Common, 73
- ExposureTimeMode_Individual, 73
- ExposureTimeSelector_Blue, 73
- ExposureTimeSelector_Common, 73
- ExposureTimeSelector_Cyan, 73
- ExposureTimeSelector_Green, 73
- ExposureTimeSelector_Infrared, 73
- ExposureTimeSelector_Magenta, 73
- ExposureTimeSelector_Red, 73
- ExposureTimeSelector_Stage1, 73
- ExposureTimeSelector_Stage2, 73
- ExposureTimeSelector_Ultraviolet, 73
- ExposureTimeSelector_Yellow, 73
- FileOpenMode_Read, 74
- FileOpenMode_ReadWrite, 74
- FileOpenMode_Write, 74
- FileOperationSelector_Close, 74
- FileOperationSelector_Delete, 74
- FileOperationSelector_Open, 74
- FileOperationSelector_Read, 74
- FileOperationSelector_Write, 74
- FileOperationStatus_Failure, 74
- FileOperationStatus_Overflow, 74
- FileOperationStatus_Success, 74

- FileSelector_SerialPort0, [74](#)
- FileSelector_UserFile1, [74](#)
- FileSelector_UserSet0, [74](#)
- FileSelector_UserSet1, [74](#)
- FileSelector_UserSetDefault, [74](#)
- GainAuto_Continuous, [75](#)
- GainAuto_Off, [75](#)
- GainAuto_Once, [75](#)
- GainAutoBalance_Continuous, [75](#)
- GainAutoBalance_Off, [75](#)
- GainAutoBalance_Once, [75](#)
- GainSelector_All, [75](#)
- GevCCP_ControlAccess, [75](#)
- GevCCP_ExclusiveAccess, [75](#)
- GevCCP_OpenAccess, [75](#)
- GevCurrentPhysicalLinkConfiguration_Dynamic↔
LAG, [76](#)
- GevCurrentPhysicalLinkConfiguration_MultiLink,
[76](#)
- GevCurrentPhysicalLinkConfiguration_SingleLink,
[76](#)
- GevCurrentPhysicalLinkConfiguration_StaticLAG,
[76](#)
- GevGVCPExtendedStatusCodesSelector_↔
Version1_1, [76](#)
- GevGVCPExtendedStatusCodesSelector_↔
Version2_0, [76](#)
- GevGVSPExtendedIDMode_Off, [76](#)
- GevGVSPExtendedIDMode_On, [76](#)
- GevIEEE1588ClockAccuracy_Unknown, [76](#)
- GevIEEE1588Mode_Auto, [77](#)
- GevIEEE1588Mode_SlaveOnly, [77](#)
- GevIEEE1588Status_Disabled, [77](#)
- GevIEEE1588Status_Faulty, [77](#)
- GevIEEE1588Status_Initializing, [77](#)
- GevIEEE1588Status_Listening, [77](#)
- GevIEEE1588Status_Master, [77](#)
- GevIEEE1588Status_Passive, [77](#)
- GevIEEE1588Status_PreMaster, [77](#)
- GevIEEE1588Status_Slave, [77](#)
- GevIEEE1588Status_Uncalibrated, [77](#)
- GevIPConfigurationStatus_DHCP, [77](#)
- GevIPConfigurationStatus_ForceIP, [77](#)
- GevIPConfigurationStatus_LLA, [77](#)
- GevIPConfigurationStatus_None, [77](#)
- GevIPConfigurationStatus_PersistentIP, [77](#)
- GevPhysicalLinkConfiguration_DynamicLAG, [77](#)
- GevPhysicalLinkConfiguration_MultiLink, [77](#)
- GevPhysicalLinkConfiguration_SingleLink, [77](#)
- GevPhysicalLinkConfiguration_StaticLAG, [77](#)
- GevSupportedOptionSelector_Action, [78](#)
- GevSupportedOptionSelector_CCPApplication↔
Socket, [78](#)
- GevSupportedOptionSelector_Commands↔
Concatenation, [78](#)
- GevSupportedOptionSelector_DiscoveryAckDelay,
[78](#)
- GevSupportedOptionSelector_DiscoveryAck↔
DelayWritable, [78](#)
- GevSupportedOptionSelector_Event, [78](#)
- GevSupportedOptionSelector_EventData, [78](#)
- GevSupportedOptionSelector_ExtendedStatus↔
Codes, [78](#)
- GevSupportedOptionSelector_HeartbeatDisable,
[78](#)
- GevSupportedOptionSelector_IPConfigurationD↔
HCP, [78](#)
- GevSupportedOptionSelector_IPConfigurationL↔
LA, [78](#)
- GevSupportedOptionSelector_IPConfiguration↔
PersistentIP, [78](#)
- GevSupportedOptionSelector_LinkSpeed, [78](#)
- GevSupportedOptionSelector_ManifestTable, [78](#)
- GevSupportedOptionSelector_MessageChannel↔
SourceSocket, [78](#)
- GevSupportedOptionSelector_PacketResend, [78](#)
- GevSupportedOptionSelector_PendingAck, [78](#)
- GevSupportedOptionSelector_SerialNumber, [78](#)
- GevSupportedOptionSelector_StreamChannel↔
SourceSocket, [78](#)
- GevSupportedOptionSelector_TestData, [78](#)
- GevSupportedOptionSelector_UserDefinedName,
[78](#)
- GevSupportedOptionSelector_WriteMem, [78](#)
- ImageComponentSelector_Color, [78](#)
- ImageComponentSelector_Confidence, [78](#)
- ImageComponentSelector_Disparity, [78](#)
- ImageComponentSelector_Infrared, [78](#)
- ImageComponentSelector_Intensity, [78](#)
- ImageComponentSelector_Range, [78](#)
- ImageComponentSelector_Scatter, [79](#)
- ImageComponentSelector_Ultraviolet, [78](#)
- ImageCompressionJPEGFormatOption_Baseline↔
Optimized, [79](#)
- ImageCompressionJPEGFormatOption_Baseline↔
Standard, [79](#)
- ImageCompressionJPEGFormatOption_Lossless,
[79](#)
- ImageCompressionJPEGFormatOption_Progressive,
[79](#)
- ImageCompressionMode_Lossless, [79](#)
- ImageCompressionMode_Off, [79](#)
- ImageCompressionRateOption_FixBitrate, [79](#)
- ImageCompressionRateOption_FixQuality, [79](#)
- LUTSelector_LUT1, [83](#)
- LineFormat_LVDS, [80](#)
- LineFormat_NoConnect, [80](#)
- LineFormat_OpenDrain, [80](#)
- LineFormat_OptoCoupled, [80](#)
- LineFormat_RS422, [80](#)
- LineFormat_TTL, [80](#)
- LineFormat_TriState, [80](#)
- LineInputFilterSelector_Debounce, [80](#)
- LineInputFilterSelector_Deglitch, [80](#)
- LineMode_Input, [80](#)

- LineMode_Output, 80
- LineSelector_Line0, 80
- LineSelector_Line1, 80
- LineSelector_Line2, 80
- LineSelector_Line3, 80
- LineSource_AllPixel, 81
- LineSource_AnyPixel, 81
- LineSource_Counter0Active, 81
- LineSource_Counter1Active, 81
- LineSource_ExposureActive, 81
- LineSource_FrameTriggerWait, 81
- LineSource_Line0, 81
- LineSource_Line1, 81
- LineSource_Line2, 81
- LineSource_Line3, 81
- LineSource_LogicBlock0, 81
- LineSource_LogicBlock1, 81
- LineSource_Off, 81
- LineSource_PPSSignal, 81
- LineSource_SerialPort0, 81
- LineSource_UserOutput0, 81
- LineSource_UserOutput1, 81
- LineSource_UserOutput2, 81
- LineSource_UserOutput3, 81
- LogicBlockLUTInputActivation_AnyEdge, 81
- LogicBlockLUTInputActivation_FallingEdge, 81
- LogicBlockLUTInputActivation_LevelHigh, 81
- LogicBlockLUTInputActivation_LevelLow, 81
- LogicBlockLUTInputActivation_RisingEdge, 81
- LogicBlockLUTInputSelector_Input0, 81
- LogicBlockLUTInputSelector_Input1, 81
- LogicBlockLUTInputSelector_Input2, 81
- LogicBlockLUTInputSelector_Input3, 81
- LogicBlockLUTInputSource_AcquisitionActive, 82
- LogicBlockLUTInputSource_Counter0End, 82
- LogicBlockLUTInputSource_Counter0Start, 82
- LogicBlockLUTInputSource_Counter1End, 82
- LogicBlockLUTInputSource_Counter1Start, 82
- LogicBlockLUTInputSource_ExposureEnd, 82
- LogicBlockLUTInputSource_ExposureStart, 82
- LogicBlockLUTInputSource_FrameTriggerWait, 82
- LogicBlockLUTInputSource_Line0, 82
- LogicBlockLUTInputSource_Line1, 82
- LogicBlockLUTInputSource_Line2, 82
- LogicBlockLUTInputSource_Line3, 82
- LogicBlockLUTInputSource_LogicBlock0, 82
- LogicBlockLUTInputSource_LogicBlock1, 82
- LogicBlockLUTInputSource_UserOutput0, 82
- LogicBlockLUTInputSource_UserOutput1, 82
- LogicBlockLUTInputSource_UserOutput2, 82
- LogicBlockLUTInputSource_UserOutput3, 82
- LogicBlockLUTInputSource_Zero, 82
- LogicBlockLUTSelector_Enable, 82
- LogicBlockLUTSelector_Value, 82
- LogicBlockSelector_LogicBlock0, 82
- LogicBlockSelector_LogicBlock1, 82
- NUM_ACQUISITIONMODE, 43
- NUM_ACQUISITIONSTATUSSELECTOR, 44
- NUM_ACTIONUNCONDITIONALMODE, 44
- NUM_ADCBITDEPTH, 44
- NUM_AUTOALGORITHMSELECTOR, 44
- NUM_AUTOEXPOSURECONTROLPRIORITY, 45
- NUM_AUTOEXPOSURELIGHTINGMODE, 45
- NUM_AUTOEXPOSUREMETERINGMODE, 45
- NUM_AUTOEXPOSURETARGETGREYVALUE↔
AUTO, 46
- NUM_BALANCERATIOSELECTOR, 46
- NUM_BALANCEWHITEAUTOPROFILE, 46
- NUM_BALANCEWHITEAUTO, 46
- NUM_BINNINGHORIZONTALMODE, 47
- NUM_BINNINGSELECTOR, 47
- NUM_BINNINGVERTICALMODE, 47
- NUM_BLACKLEVELAUTOBALANCE, 47
- NUM_BLACKLEVELAUTO, 48
- NUM_BLACKLEVELSELECTOR, 48
- NUM_CHUNKBLACKLEVELSELECTOR, 48
- NUM_CHUNKCOUNTERSELECTOR, 48
- NUM_CHUNKENCODERSELECTOR, 49
- NUM_CHUNKENCODERSTATUS, 49
- NUM_CHUNKEXPOSURETIMESELECTOR, 49
- NUM_CHUNKGAINSELECTOR, 50
- NUM_CHUNKIMAGECOMPONENT, 50
- NUM_CHUNKPIXELFORMAT, 50
- NUM_CHUNKREGIONID, 51
- NUM_CHUNKSCAN3DCOORDINATEREFERE↔
NCSELECTOR, 51
- NUM_CHUNKSCAN3DCOORDINATESELECT↔
OR, 51
- NUM_CHUNKSCAN3DCOORDINATESYSTEM↔
REFERENCE, 52
- NUM_CHUNKSCAN3DCOORDINATESYSTEM,
51
- NUM_CHUNKSCAN3DCOORDINATETRANSF↔
ORMSELECTOR, 52
- NUM_CHUNKSCAN3DDISTANCEUNIT, 52
- NUM_CHUNKSCAN3DOUTPUTMODE, 53
- NUM_CHUNKSELECTOR, 54
- NUM_CHUNKSOURCEID, 54
- NUM_CHUNKTIMERSELECTOR, 54
- NUM_CHUNKTRANSFERSTREAMID, 54
- NUM_CLCONFIGURATION, 55
- NUM_CLTIMESLOTSCOUNT, 55
- NUM_COLORTRANSFORMATIONSELECTOR,
55
- NUM_COLORTRANSFORMATIONVALUESEL↔
ECTOR, 56
- NUM_COUNTEREVENTACTIVATION, 56
- NUM_COUNTEREVENTSOURCE, 57
- NUM_COUNTERRESETACTIVATION, 57
- NUM_COUNTERRESETSOURCE, 57
- NUM_COUNTERSELECTOR, 58
- NUM_COUNTERSTATUS, 58
- NUM_COUNTERTRIGGERACTIVATION, 58
- NUM_COUNTERTRIGGERSOURCE, 59
- NUM_CXPCONNECTIONTESTMODE, 59

NUM_CXPLINKCONFIGURATIONPREFERRED, 61
 NUM_CXPLINKCONFIGURATIONSTATUS, 62
 NUM_CXPLINKCONFIGURATION, 60
 NUM_CXPPOCXSTATUS, 62
 NUM_DECIMATIONHORIZONTALMODE, 62
 NUM_DECIMATIONSELECTOR, 63
 NUM_DECIMATIONVERTICALMODE, 63
 NUM_DEFECTCORRECTIONMODE, 63
 NUM_DEINTERLACING, 63
 NUM_DEVICECHARACTERSET, 64
 NUM_DEVICECLOCKSELECTOR, 64
 NUM_DEVICECONNECTIONSTATUS, 64
 NUM_DEVICEINDICATORMODE, 64
 NUM_DEVICELINKHEARTBEATMODE, 65
 NUM_DEVICELINKTHROUGHPUTLIMITMODE, 65
 NUM_DEVICEPOWERSUPPLYSELECTOR, 65
 NUM_DEVICEREGISTERSENDIANNES, 65
 NUM_DEVICESCANTYPE, 65
 NUM_DEVICESERIALPORTBAUDRATE, 66
 NUM_DEVICESERIALPORTSELECTOR, 66
 NUM_DEVICESTREAMCHANNELENDIANNES, 66
 NUM_DEVICESTREAMCHANNELTYPE, 66
 NUM_DEVICETAPGEOMETRY, 68
 NUM_DEVICETEMPERATURESELECTOR, 68
 NUM_DEVICETLTYPE, 68
 NUM_DEVICETYPE, 68
 NUM_ENCODERMODE, 69
 NUM_ENCODEROUTPUTMODE, 69
 NUM_ENCODERRESETACTIVATION, 69
 NUM_ENCODERRESETSOURCE, 70
 NUM_ENCODERSELECTOR, 71
 NUM_ENCODERSOURCEA, 71
 NUM_ENCODERSOURCEB, 71
 NUM_ENCODERSTATUS, 71
 NUM_EVENTNOTIFICATION, 72
 NUM_EVENTSELECTOR, 72
 NUM_EXPOSUREACTIVEMODE, 72
 NUM_EXPOSUREAUTO, 72
 NUM_EXPOSUREMODE, 73
 NUM_EXPOSURETIMEMODE, 73
 NUM_EXPOSURETIMESELECTOR, 73
 NUM_FILEOPENMODE, 74
 NUM_FILEOPERATIONSELECTOR, 74
 NUM_FILEOPERATIONSTATUS, 74
 NUM_FILESELECTOR, 74
 NUM_GAINAUTOBALANCE, 75
 NUM_GAINAUTO, 75
 NUM_GAINSELECTOR, 75
 NUM_GEVCCP, 75
 NUM_GEVCURRENTPHYSICALLINKCONFIGURATION, 76
 NUM_GEVGVCPEXTENDEDSTATUSCODESELECTOR, 76
 NUM_GEVGVSPEXTENDEDIDMODE, 76
 NUM_GEVIIEEE1588CLOCKACCURACY, 76
 NUM_GEVIIEEE1588MODE, 77
 NUM_GEVIIEEE1588STATUS, 77
 NUM_GEVIPCONFIGURATIONSTATUS, 77
 NUM_GEVPHYSICALLINKCONFIGURATION, 77
 NUM_GEVSUPPORTEDOPTIONSELECTOR, 78
 NUM_IMAGECOMPONENTSELECTOR, 79
 NUM_IMAGECOMPRESSIONJPEGFORMATOPTION, 79
 NUM_IMAGECOMPRESSIONMODE, 79
 NUM_IMAGECOMPRESSIONRATEOPTION, 79
 NUM_LINEFORMAT, 80
 NUM_LINEINPUTFILTERSELECTOR, 80
 NUM_LINEMODE, 80
 NUM_LINESELECTOR, 80
 NUM_LINESOURCE, 81
 NUM_LOGICBLOCKLUTINPUTACTIVATION, 81
 NUM_LOGICBLOCKLUTINPUTSELECTOR, 81
 NUM_LOGICBLOCKLUTINPUTSOURCE, 82
 NUM_LOGICBLOCKLUTSELECTOR, 82
 NUM_LOGICBLOCKSELECTOR, 82
 NUM_LUTSELECTOR, 83
 NUM_PIXELCOLORFILTER, 83
 NUM_PIXELFORMATINFOSELECTOR, 94
 NUM_PIXELFORMAT, 88
 NUM_PIXELSIZE, 94
 NUM_REGIONDESTINATION, 94
 NUM_REGIONMODE, 95
 NUM_REGIONSELECTOR, 95
 NUM_RGBTRANSFORMLIGHTSOURCE, 95
 NUM_SCAN3DCOORDINATEREFERENCESELECTOR, 96
 NUM_SCAN3DCOORDINATESELECTOR, 96
 NUM_SCAN3DCOORDINATESYSTEMREFERENCE, 96
 NUM_SCAN3DCOORDINATESYSTEM, 96
 NUM_SCAN3DCOORDINATETRANSFORMSELECTOR, 97
 NUM_SCAN3DDISTANCEUNIT, 97
 NUM_SCAN3DOUTPUTMODE, 98
 NUM_SENSORDIGITIZATIONTAPS, 98
 NUM_SENSORSHUTTERMODE, 98
 NUM_SENSORTAPS, 98
 NUM_SEQUENCERCONFIGURATIONMODE, 99
 NUM_SEQUENCERCONFIGURATIONVALID, 99
 NUM_SEQUENCERMODE, 99
 NUM_SEQUENCERSETVALID, 99
 NUM_SEQUENCERTRIGGERACTIVATION, 99
 NUM_SEQUENCERTRIGGERSOURCE, 100
 NUM_SERIALPORTBAUDRATE, 100
 NUM_SERIALPORTPARITY, 100
 NUM_SERIALPORTSELECTOR, 101
 NUM_SERIALPORTSOURCE, 101
 NUM_SERIALPORTSTOPBITS, 101
 NUM_SOFTWARESIGNALSELECTOR, 101
 NUM_SOURCESELECTOR, 102
 NUM_TESTPATTERNGENERATORSELECTOR, 102
 NUM_TESTPATTERN, 102

NUM_TIMERSELECTOR, 102
 NUM_TIMERSTATUS, 103
 NUM_TIMERTRIGGERACTIVATION, 103
 NUM_TIMERTRIGGERSOURCE, 104
 NUM_TRANSFERCOMPONENTSELECTOR, 105
 NUM_TRANSFERCONTROLMODE, 105
 NUM_TRANSFEROPERATIONMODE, 105
 NUM_TRANSFERQUEUEMODE, 105
 NUM_TRANSFERSELECTOR, 106
 NUM_TRANSFERSTATUSSELECTOR, 106
 NUM_TRANSFERTRIGGERACTIVATION, 106
 NUM_TRANSFERTRIGGERMODE, 107
 NUM_TRANSFERTRIGGERSELECTOR, 107
 NUM_TRANSFERTRIGGERSOURCE, 108
 NUM_TRIGGERACTIVATION, 108
 NUM_TRIGGERMODE, 108
 NUM_TRIGGEROVERLAP, 109
 NUM_TRIGGERSELECTOR, 109
 NUM_TRIGGERSOURCE, 109
 NUM_USEROUTPUTSELECTOR, 110
 NUM_USERSETDEFAULT, 110
 NUM_USERSETSELECTOR, 110
 NUM_WHITECLIPSELECTOR, 110
 PixelColorFilter_BayerBG, 83
 PixelColorFilter_BayerGB, 83
 PixelColorFilter_BayerGR, 83
 PixelColorFilter_BayerRG, 83
 PixelColorFilter_None, 83
 PixelFormat_B10, 85
 PixelFormat_B12, 85
 PixelFormat_B12_Jpeg, 88
 PixelFormat_B16, 85
 PixelFormat_B8, 85
 PixelFormat_BGR10, 85
 PixelFormat_BGR10p, 85
 PixelFormat_BGR12, 85
 PixelFormat_BGR12p, 85
 PixelFormat_BGR14, 85
 PixelFormat_BGR16, 85
 PixelFormat_BGR565p, 85
 PixelFormat_BGR8, 84
 PixelFormat_BGRa10, 85
 PixelFormat_BGRa10p, 85
 PixelFormat_BGRa12, 85
 PixelFormat_BGRa12p, 85
 PixelFormat_BGRa14, 85
 PixelFormat_BGRa16, 85
 PixelFormat_BGRa8, 84
 PixelFormat_BayerBG10, 84
 PixelFormat_BayerBG10Packed, 84
 PixelFormat_BayerBG10p, 84
 PixelFormat_BayerBG12, 84
 PixelFormat_BayerBG12Packed, 83
 PixelFormat_BayerBG12p, 84
 PixelFormat_BayerBG16, 83
 PixelFormat_BayerBG8, 83
 PixelFormat_BayerGB10, 84
 PixelFormat_BayerGB10Packed, 84
 PixelFormat_BayerGB12, 84
 PixelFormat_BayerGB12Packed, 83
 PixelFormat_BayerGB12p, 84
 PixelFormat_BayerGB16, 83
 PixelFormat_BayerGB8, 83
 PixelFormat_BayerGR10, 84
 PixelFormat_BayerGR10Packed, 84
 PixelFormat_BayerGR10p, 84
 PixelFormat_BayerGR12, 84
 PixelFormat_BayerGR12Packed, 83
 PixelFormat_BayerGR12p, 84
 PixelFormat_BayerGR16, 83
 PixelFormat_BayerGR8, 83
 PixelFormat_BayerRG10, 84
 PixelFormat_BayerRG10Packed, 84
 PixelFormat_BayerRG10p, 84
 PixelFormat_BayerRG12, 84
 PixelFormat_BayerRG12Packed, 83
 PixelFormat_BayerRG12p, 84
 PixelFormat_BayerRG16, 83
 PixelFormat_BayerRG8, 83
 PixelFormat_BayerRGPolarized10p, 88
 PixelFormat_BayerRGPolarized12p, 88
 PixelFormat_BayerRGPolarized16, 88
 PixelFormat_BayerRGPolarized8, 88
 PixelFormat_BiColorBGRG10, 86
 PixelFormat_BiColorBGRG10p, 86
 PixelFormat_BiColorBGRG12, 86
 PixelFormat_BiColorBGRG12p, 86
 PixelFormat_BiColorBGRG8, 86
 PixelFormat_BiColorRGBG10, 86
 PixelFormat_BiColorRGBG10p, 86
 PixelFormat_BiColorRGBG12, 86
 PixelFormat_BiColorRGBG12p, 86
 PixelFormat_BiColorRGBG8, 86
 PixelFormat_Confidence1, 86
 PixelFormat_Confidence16, 86
 PixelFormat_Confidence1p, 86
 PixelFormat_Confidence32f, 86
 PixelFormat_Confidence8, 86
 PixelFormat_Coord3D_A10p, 86
 PixelFormat_Coord3D_A12p, 86
 PixelFormat_Coord3D_A16, 86
 PixelFormat_Coord3D_A32f, 86
 PixelFormat_Coord3D_A8, 86
 PixelFormat_Coord3D_ABC10p, 85
 PixelFormat_Coord3D_ABC10p_Planar, 85
 PixelFormat_Coord3D_ABC12p, 85
 PixelFormat_Coord3D_ABC12p_Planar, 85
 PixelFormat_Coord3D_ABC16, 85
 PixelFormat_Coord3D_ABC16_Planar, 85
 PixelFormat_Coord3D_ABC32f, 85
 PixelFormat_Coord3D_ABC32f_Planar, 86
 PixelFormat_Coord3D_ABC8, 85
 PixelFormat_Coord3D_ABC8_Planar, 85
 PixelFormat_Coord3D_AC10p, 86
 PixelFormat_Coord3D_AC10p_Planar, 86

PixelFormat_Coord3D_AC12p, [86](#)
PixelFormat_Coord3D_AC12p_Planar, [86](#)
PixelFormat_Coord3D_AC16, [86](#)
PixelFormat_Coord3D_AC16_Planar, [86](#)
PixelFormat_Coord3D_AC32f, [86](#)
PixelFormat_Coord3D_AC32f_Planar, [86](#)
PixelFormat_Coord3D_AC8, [86](#)
PixelFormat_Coord3D_AC8_Planar, [86](#)
PixelFormat_Coord3D_B10p, [86](#)
PixelFormat_Coord3D_B12p, [86](#)
PixelFormat_Coord3D_B16, [86](#)
PixelFormat_Coord3D_B32f, [86](#)
PixelFormat_Coord3D_B8, [86](#)
PixelFormat_Coord3D_C10p, [86](#)
PixelFormat_Coord3D_C12p, [86](#)
PixelFormat_Coord3D_C16, [86](#)
PixelFormat_Coord3D_C32f, [86](#)
PixelFormat_Coord3D_C8, [86](#)
PixelFormat_G10, [85](#)
PixelFormat_G12, [85](#)
PixelFormat_G16, [85](#)
PixelFormat_G8, [85](#)
PixelFormat_GB12_Jpeg, [88](#)
PixelFormat_GR12_Jpeg, [88](#)
PixelFormat_Mono10, [84](#)
PixelFormat_Mono10Packed, [84](#)
PixelFormat_Mono10p, [84](#)
PixelFormat_Mono12, [84](#)
PixelFormat_Mono12Packed, [83](#)
PixelFormat_Mono12p, [84](#)
PixelFormat_Mono14, [84](#)
PixelFormat_Mono16, [83](#)
PixelFormat_Mono1p, [84](#)
PixelFormat_Mono2p, [84](#)
PixelFormat_Mono4p, [84](#)
PixelFormat_Mono8, [83](#)
PixelFormat_Mono8s, [84](#)
PixelFormat_Polarized10p, [88](#)
PixelFormat_Polarized12p, [88](#)
PixelFormat_Polarized16, [88](#)
PixelFormat_Polarized8, [88](#)
PixelFormat_R10, [85](#)
PixelFormat_R12, [85](#)
PixelFormat_R12_Jpeg, [88](#)
PixelFormat_R16, [85](#)
PixelFormat_R8, [85](#)
PixelFormat_RGB10, [85](#)
PixelFormat_RGB10_Planar, [85](#)
PixelFormat_RGB10p, [85](#)
PixelFormat_RGB10p32, [85](#)
PixelFormat_RGB12, [85](#)
PixelFormat_RGB12_Planar, [85](#)
PixelFormat_RGB12p, [85](#)
PixelFormat_RGB14, [85](#)
PixelFormat_RGB16, [85](#)
PixelFormat_RGB16_Planar, [85](#)
PixelFormat_RGB565p, [85](#)
PixelFormat_RGB8, [84](#)
PixelFormat_RGB8_Planar, [84](#)
PixelFormat_RGB8Packed, [83](#)
PixelFormat_RGBa10, [84](#)
PixelFormat_RGBa10p, [84](#)
PixelFormat_RGBa12, [84](#)
PixelFormat_RGBa12p, [84](#)
PixelFormat_RGBa14, [84](#)
PixelFormat_RGBa16, [84](#)
PixelFormat_RGBa8, [84](#)
PixelFormat_Raw16, [88](#)
PixelFormat_Raw8, [88](#)
PixelFormat_SCF1WBWG10, [86](#)
PixelFormat_SCF1WBWG10p, [86](#)
PixelFormat_SCF1WBWG12, [86](#)
PixelFormat_SCF1WBWG12p, [87](#)
PixelFormat_SCF1WBWG14, [87](#)
PixelFormat_SCF1WBWG16, [87](#)
PixelFormat_SCF1WBWG8, [86](#)
PixelFormat_SCF1WGBW10, [87](#)
PixelFormat_SCF1WGBW10p, [87](#)
PixelFormat_SCF1WGBW12, [87](#)
PixelFormat_SCF1WGBW12p, [87](#)
PixelFormat_SCF1WGBW14, [87](#)
PixelFormat_SCF1WGBW16, [87](#)
PixelFormat_SCF1WGBW8, [87](#)
PixelFormat_SCF1WGWR10, [87](#)
PixelFormat_SCF1WGWR10p, [87](#)
PixelFormat_SCF1WGWR12, [87](#)
PixelFormat_SCF1WGWR12p, [87](#)
PixelFormat_SCF1WGWR14, [87](#)
PixelFormat_SCF1WGWR16, [87](#)
PixelFormat_SCF1WGWR8, [87](#)
PixelFormat_SCF1WRWG10, [87](#)
PixelFormat_SCF1WRWG10p, [87](#)
PixelFormat_SCF1WRWG12, [87](#)
PixelFormat_SCF1WRWG12p, [87](#)
PixelFormat_SCF1WRWG14, [87](#)
PixelFormat_SCF1WRWG16, [87](#)
PixelFormat_SCF1WRWG8, [87](#)
PixelFormat_YCbCr10_CbYCr, [87](#)
PixelFormat_YCbCr10p_CbYCr, [87](#)
PixelFormat_YCbCr12_CbYCr, [87](#)
PixelFormat_YCbCr12p_CbYCr, [87](#)
PixelFormat_YCbCr411_8, [84](#)
PixelFormat_YCbCr411_8_CbYYCrYY, [87](#)
PixelFormat_YCbCr422_10, [87](#)
PixelFormat_YCbCr422_10_CbYCrY, [87](#)
PixelFormat_YCbCr422_10p, [87](#)
PixelFormat_YCbCr422_10p_CbYCrY, [87](#)
PixelFormat_YCbCr422_12, [87](#)
PixelFormat_YCbCr422_12_CbYCrY, [87](#)
PixelFormat_YCbCr422_12p, [87](#)
PixelFormat_YCbCr422_12p_CbYCrY, [87](#)
PixelFormat_YCbCr422_8, [84](#)
PixelFormat_YCbCr422_8_CbYCrY, [87](#)
PixelFormat_YCbCr601_10_CbYCr, [87](#)
PixelFormat_YCbCr601_10p_CbYCr, [87](#)
PixelFormat_YCbCr601_12_CbYCr, [87](#)

- PixelFormat_YCbCr601_12p_CbYCr, [87](#)
- PixelFormat_YCbCr601_411_8_CbYYCrYY, [87](#)
- PixelFormat_YCbCr601_422_10, [88](#)
- PixelFormat_YCbCr601_422_10_CbYCrY, [88](#)
- PixelFormat_YCbCr601_422_10p, [88](#)
- PixelFormat_YCbCr601_422_10p_CbYCrY, [88](#)
- PixelFormat_YCbCr601_422_12, [88](#)
- PixelFormat_YCbCr601_422_12_CbYCrY, [88](#)
- PixelFormat_YCbCr601_422_12p, [88](#)
- PixelFormat_YCbCr601_422_12p_CbYCrY, [88](#)
- PixelFormat_YCbCr601_422_8, [88](#)
- PixelFormat_YCbCr601_422_8_CbYCrY, [88](#)
- PixelFormat_YCbCr601_8_CbYCr, [87](#)
- PixelFormat_YCbCr709_10_CbYCr, [88](#)
- PixelFormat_YCbCr709_10p_CbYCr, [88](#)
- PixelFormat_YCbCr709_12_CbYCr, [88](#)
- PixelFormat_YCbCr709_12p_CbYCr, [88](#)
- PixelFormat_YCbCr709_411_8_CbYYCrYY, [88](#)
- PixelFormat_YCbCr709_422_10, [88](#)
- PixelFormat_YCbCr709_422_10_CbYCrY, [88](#)
- PixelFormat_YCbCr709_422_10p, [88](#)
- PixelFormat_YCbCr709_422_10p_CbYCrY, [88](#)
- PixelFormat_YCbCr709_422_12, [88](#)
- PixelFormat_YCbCr709_422_12_CbYCrY, [88](#)
- PixelFormat_YCbCr709_422_12p, [88](#)
- PixelFormat_YCbCr709_422_12p_CbYCrY, [88](#)
- PixelFormat_YCbCr709_422_8, [88](#)
- PixelFormat_YCbCr709_422_8_CbYCrY, [88](#)
- PixelFormat_YCbCr709_8_CbYCr, [88](#)
- PixelFormat_YCbCr8, [84](#)
- PixelFormat_YCbCr8_CbYCr, [87](#)
- PixelFormat_YUV411_8_UYYVYY, [88](#)
- PixelFormat_YUV411Packed, [83](#)
- PixelFormat_YUV422_8, [88](#)
- PixelFormat_YUV422_8_UYVY, [88](#)
- PixelFormat_YUV422Packed, [83](#)
- PixelFormat_YUV444Packed, [84](#)
- PixelFormat_YUV8_UYV, [88](#)
- PixelFormatInfoSelector_B10, [90](#)
- PixelFormatInfoSelector_B12, [90](#)
- PixelFormatInfoSelector_B16, [90](#)
- PixelFormatInfoSelector_B8, [90](#)
- PixelFormatInfoSelector_BGR10, [90](#)
- PixelFormatInfoSelector_BGR10p, [90](#)
- PixelFormatInfoSelector_BGR12, [90](#)
- PixelFormatInfoSelector_BGR12p, [90](#)
- PixelFormatInfoSelector_BGR14, [90](#)
- PixelFormatInfoSelector_BGR16, [90](#)
- PixelFormatInfoSelector_BGR565p, [90](#)
- PixelFormatInfoSelector_BGR8, [90](#)
- PixelFormatInfoSelector_BGRa10, [90](#)
- PixelFormatInfoSelector_BGRa10p, [90](#)
- PixelFormatInfoSelector_BGRa12, [90](#)
- PixelFormatInfoSelector_BGRa12p, [90](#)
- PixelFormatInfoSelector_BGRa14, [90](#)
- PixelFormatInfoSelector_BGRa16, [90](#)
- PixelFormatInfoSelector_BGRa8, [90](#)
- PixelFormatInfoSelector_BayerBG10, [89](#)
- PixelFormatInfoSelector_BayerBG10p, [89](#)
- PixelFormatInfoSelector_BayerBG12, [89](#)
- PixelFormatInfoSelector_BayerBG12p, [89](#)
- PixelFormatInfoSelector_BayerBG16, [89](#)
- PixelFormatInfoSelector_BayerBG8, [89](#)
- PixelFormatInfoSelector_BayerGB10, [89](#)
- PixelFormatInfoSelector_BayerGB10p, [89](#)
- PixelFormatInfoSelector_BayerGB12, [89](#)
- PixelFormatInfoSelector_BayerGB12p, [89](#)
- PixelFormatInfoSelector_BayerGB16, [89](#)
- PixelFormatInfoSelector_BayerGB8, [89](#)
- PixelFormatInfoSelector_BayerGR10, [89](#)
- PixelFormatInfoSelector_BayerGR10p, [89](#)
- PixelFormatInfoSelector_BayerGR12, [89](#)
- PixelFormatInfoSelector_BayerGR12p, [89](#)
- PixelFormatInfoSelector_BayerGR16, [89](#)
- PixelFormatInfoSelector_BayerGR8, [89](#)
- PixelFormatInfoSelector_BayerRG10, [89](#)
- PixelFormatInfoSelector_BayerRG10p, [89](#)
- PixelFormatInfoSelector_BayerRG12, [89](#)
- PixelFormatInfoSelector_BayerRG12p, [89](#)
- PixelFormatInfoSelector_BayerRG16, [89](#)
- PixelFormatInfoSelector_BayerRG8, [89](#)
- PixelFormatInfoSelector_BayerRGPolarized10p, [94](#)
- PixelFormatInfoSelector_BayerRGPolarized12p, [94](#)
- PixelFormatInfoSelector_BayerRGPolarized16, [94](#)
- PixelFormatInfoSelector_BayerRGPolarized8, [94](#)
- PixelFormatInfoSelector_BiColorBGRG10, [91](#)
- PixelFormatInfoSelector_BiColorBGRG10p, [91](#)
- PixelFormatInfoSelector_BiColorBGRG12, [91](#)
- PixelFormatInfoSelector_BiColorBGRG12p, [91](#)
- PixelFormatInfoSelector_BiColorBGRG8, [91](#)
- PixelFormatInfoSelector_BiColorRGBG10, [91](#)
- PixelFormatInfoSelector_BiColorRGBG10p, [91](#)
- PixelFormatInfoSelector_BiColorRGBG12, [91](#)
- PixelFormatInfoSelector_BiColorRGBG12p, [92](#)
- PixelFormatInfoSelector_BiColorRGBG8, [91](#)
- PixelFormatInfoSelector_Confidence1, [91](#)
- PixelFormatInfoSelector_Confidence16, [91](#)
- PixelFormatInfoSelector_Confidence1p, [91](#)
- PixelFormatInfoSelector_Confidence32f, [91](#)
- PixelFormatInfoSelector_Confidence8, [91](#)
- PixelFormatInfoSelector_Coord3D_A10p, [91](#)
- PixelFormatInfoSelector_Coord3D_A12p, [91](#)
- PixelFormatInfoSelector_Coord3D_A16, [91](#)
- PixelFormatInfoSelector_Coord3D_A32f, [91](#)
- PixelFormatInfoSelector_Coord3D_A8, [91](#)
- PixelFormatInfoSelector_Coord3D_ABC10p, [90](#)
- PixelFormatInfoSelector_Coord3D_ABC10p_↔ Planar, [90](#)
- PixelFormatInfoSelector_Coord3D_ABC12p, [91](#)
- PixelFormatInfoSelector_Coord3D_ABC12p_↔ Planar, [91](#)
- PixelFormatInfoSelector_Coord3D_ABC16, [91](#)
- PixelFormatInfoSelector_Coord3D_ABC16_↔ Planar, [91](#)

- PixelFormatInfoSelector_Coord3D_ABC32f, [91](#)
- PixelFormatInfoSelector_Coord3D_ABC32f_↔
Planar, [91](#)
- PixelFormatInfoSelector_Coord3D_ABC8, [90](#)
- PixelFormatInfoSelector_Coord3D_ABC8_Planar,
[90](#)
- PixelFormatInfoSelector_Coord3D_AC10p, [91](#)
- PixelFormatInfoSelector_Coord3D_AC10p_Planar,
[91](#)
- PixelFormatInfoSelector_Coord3D_AC12p, [91](#)
- PixelFormatInfoSelector_Coord3D_AC12p_Planar,
[91](#)
- PixelFormatInfoSelector_Coord3D_AC16, [91](#)
- PixelFormatInfoSelector_Coord3D_AC16_Planar,
[91](#)
- PixelFormatInfoSelector_Coord3D_AC32f, [91](#)
- PixelFormatInfoSelector_Coord3D_AC32f_Planar,
[91](#)
- PixelFormatInfoSelector_Coord3D_AC8, [91](#)
- PixelFormatInfoSelector_Coord3D_AC8_Planar,
[91](#)
- PixelFormatInfoSelector_Coord3D_B10p, [91](#)
- PixelFormatInfoSelector_Coord3D_B12p, [91](#)
- PixelFormatInfoSelector_Coord3D_B16, [91](#)
- PixelFormatInfoSelector_Coord3D_B32f, [91](#)
- PixelFormatInfoSelector_Coord3D_B8, [91](#)
- PixelFormatInfoSelector_Coord3D_C10p, [91](#)
- PixelFormatInfoSelector_Coord3D_C12p, [91](#)
- PixelFormatInfoSelector_Coord3D_C16, [91](#)
- PixelFormatInfoSelector_Coord3D_C32f, [91](#)
- PixelFormatInfoSelector_Coord3D_C8, [91](#)
- PixelFormatInfoSelector_G10, [90](#)
- PixelFormatInfoSelector_G12, [90](#)
- PixelFormatInfoSelector_G16, [90](#)
- PixelFormatInfoSelector_G8, [90](#)
- PixelFormatInfoSelector_Mono10, [89](#)
- PixelFormatInfoSelector_Mono10p, [89](#)
- PixelFormatInfoSelector_Mono12, [89](#)
- PixelFormatInfoSelector_Mono12p, [89](#)
- PixelFormatInfoSelector_Mono14, [89](#)
- PixelFormatInfoSelector_Mono16, [89](#)
- PixelFormatInfoSelector_Mono1p, [89](#)
- PixelFormatInfoSelector_Mono2p, [89](#)
- PixelFormatInfoSelector_Mono4p, [89](#)
- PixelFormatInfoSelector_Mono8, [89](#)
- PixelFormatInfoSelector_Mono8s, [89](#)
- PixelFormatInfoSelector_Polarized10p, [94](#)
- PixelFormatInfoSelector_Polarized12p, [94](#)
- PixelFormatInfoSelector_Polarized16, [94](#)
- PixelFormatInfoSelector_Polarized8, [94](#)
- PixelFormatInfoSelector_R10, [90](#)
- PixelFormatInfoSelector_R12, [90](#)
- PixelFormatInfoSelector_R16, [90](#)
- PixelFormatInfoSelector_R8, [90](#)
- PixelFormatInfoSelector_RGB10, [90](#)
- PixelFormatInfoSelector_RGB10_Planar, [90](#)
- PixelFormatInfoSelector_RGB10p, [90](#)
- PixelFormatInfoSelector_RGB10p32, [90](#)
- PixelFormatInfoSelector_RGB12, [90](#)
- PixelFormatInfoSelector_RGB12_Planar, [90](#)
- PixelFormatInfoSelector_RGB12p, [90](#)
- PixelFormatInfoSelector_RGB14, [90](#)
- PixelFormatInfoSelector_RGB16, [90](#)
- PixelFormatInfoSelector_RGB16_Planar, [90](#)
- PixelFormatInfoSelector_RGB565p, [90](#)
- PixelFormatInfoSelector_RGB8, [90](#)
- PixelFormatInfoSelector_RGB8_Planar, [90](#)
- PixelFormatInfoSelector_RGBa10, [89](#)
- PixelFormatInfoSelector_RGBa10p, [89](#)
- PixelFormatInfoSelector_RGBa12, [89](#)
- PixelFormatInfoSelector_RGBa12p, [89](#)
- PixelFormatInfoSelector_RGBa14, [89](#)
- PixelFormatInfoSelector_RGBa16, [90](#)
- PixelFormatInfoSelector_RGBa8, [89](#)
- PixelFormatInfoSelector_SCF1WBWG10, [92](#)
- PixelFormatInfoSelector_SCF1WBWG10p, [92](#)
- PixelFormatInfoSelector_SCF1WBWG12, [92](#)
- PixelFormatInfoSelector_SCF1WBWG12p, [92](#)
- PixelFormatInfoSelector_SCF1WBWG14, [92](#)
- PixelFormatInfoSelector_SCF1WBWG16, [92](#)
- PixelFormatInfoSelector_SCF1WBWG8, [92](#)
- PixelFormatInfoSelector_SCF1WGWB10, [92](#)
- PixelFormatInfoSelector_SCF1WGWB10p, [92](#)
- PixelFormatInfoSelector_SCF1WGWB12, [92](#)
- PixelFormatInfoSelector_SCF1WGWB12p, [92](#)
- PixelFormatInfoSelector_SCF1WGWB14, [92](#)
- PixelFormatInfoSelector_SCF1WGWB16, [92](#)
- PixelFormatInfoSelector_SCF1WGWB8, [92](#)
- PixelFormatInfoSelector_SCF1WGWR10, [92](#)
- PixelFormatInfoSelector_SCF1WGWR10p, [92](#)
- PixelFormatInfoSelector_SCF1WGWR12, [92](#)
- PixelFormatInfoSelector_SCF1WGWR12p, [92](#)
- PixelFormatInfoSelector_SCF1WGWR14, [92](#)
- PixelFormatInfoSelector_SCF1WGWR16, [92](#)
- PixelFormatInfoSelector_SCF1WGWR8, [92](#)
- PixelFormatInfoSelector_SCF1WRWG10, [92](#)
- PixelFormatInfoSelector_SCF1WRWG10p, [92](#)
- PixelFormatInfoSelector_SCF1WRWG12, [92](#)
- PixelFormatInfoSelector_SCF1WRWG12p, [92](#)
- PixelFormatInfoSelector_SCF1WRWG14, [92](#)
- PixelFormatInfoSelector_SCF1WRWG16, [92](#)
- PixelFormatInfoSelector_SCF1WRWG8, [92](#)
- PixelFormatInfoSelector_YCbCr10_CbYCr, [92](#)
- PixelFormatInfoSelector_YCbCr10p_CbYCr, [92](#)
- PixelFormatInfoSelector_YCbCr12_CbYCr, [92](#)
- PixelFormatInfoSelector_YCbCr12p_CbYCr, [92](#)
- PixelFormatInfoSelector_YCbCr411_8, [93](#)
- PixelFormatInfoSelector_YCbCr411_8_CbYYCr↔
YY, [93](#)
- PixelFormatInfoSelector_YCbCr422_10, [93](#)
- PixelFormatInfoSelector_YCbCr422_10_CbYCrY,
[93](#)
- PixelFormatInfoSelector_YCbCr422_10p, [93](#)
- PixelFormatInfoSelector_YCbCr422_10p_CbY↔
CrY, [93](#)
- PixelFormatInfoSelector_YCbCr422_12, [93](#)

- PixelFormatInfoSelector_YCbCr422_12_CbYCrY, 93
- PixelFormatInfoSelector_YCbCr422_12p, 93
- PixelFormatInfoSelector_YCbCr422_12p_CbY←CrY, 93
- PixelFormatInfoSelector_YCbCr422_8, 93
- PixelFormatInfoSelector_YCbCr422_8_CbYCrY, 93
- PixelFormatInfoSelector_YCbCr601_10_CbYCr, 93
- PixelFormatInfoSelector_YCbCr601_10p_CbYCr, 93
- PixelFormatInfoSelector_YCbCr601_12_CbYCr, 93
- PixelFormatInfoSelector_YCbCr601_12p_CbYCr, 93
- PixelFormatInfoSelector_YCbCr601_411_8_Cb←YYCrYY, 93
- PixelFormatInfoSelector_YCbCr601_422_10, 93
- PixelFormatInfoSelector_YCbCr601_422_10_←CbYCrY, 93
- PixelFormatInfoSelector_YCbCr601_422_10p, 93
- PixelFormatInfoSelector_YCbCr601_422_10p_←CbYCrY, 93
- PixelFormatInfoSelector_YCbCr601_422_12, 93
- PixelFormatInfoSelector_YCbCr601_422_12_←CbYCrY, 93
- PixelFormatInfoSelector_YCbCr601_422_12p, 93
- PixelFormatInfoSelector_YCbCr601_422_12p_←CbYCrY, 93
- PixelFormatInfoSelector_YCbCr601_422_8, 93
- PixelFormatInfoSelector_YCbCr601_422_8_Cb←YCrY, 93
- PixelFormatInfoSelector_YCbCr601_8_CbYCr, 93
- PixelFormatInfoSelector_YCbCr709_10_CbYCr, 93
- PixelFormatInfoSelector_YCbCr709_10p_CbYCr, 93
- PixelFormatInfoSelector_YCbCr709_12_CbYCr, 93
- PixelFormatInfoSelector_YCbCr709_12p_CbYCr, 93
- PixelFormatInfoSelector_YCbCr709_411_8_Cb←YYCrYY, 93
- PixelFormatInfoSelector_YCbCr709_422_10, 93
- PixelFormatInfoSelector_YCbCr709_422_10_←CbYCrY, 93
- PixelFormatInfoSelector_YCbCr709_422_10p, 93
- PixelFormatInfoSelector_YCbCr709_422_10p_←CbYCrY, 93
- PixelFormatInfoSelector_YCbCr709_422_12, 93
- PixelFormatInfoSelector_YCbCr709_422_12_←CbYCrY, 93
- PixelFormatInfoSelector_YCbCr709_422_12p, 93
- PixelFormatInfoSelector_YCbCr709_422_12p_←CbYCrY, 93
- PixelFormatInfoSelector_YCbCr709_422_8, 93
- PixelFormatInfoSelector_YCbCr709_422_8_Cb←YCrY, 93
- PixelFormatInfoSelector_YCbCr709_8_CbYCr, 93
- PixelFormatInfoSelector_YCbCr8, 92
- PixelFormatInfoSelector_YCbCr8_CbYCr, 92
- PixelFormatInfoSelector_YUV411_8_UYYVYY, 94
- PixelFormatInfoSelector_YUV422_8, 94
- PixelFormatInfoSelector_YUV422_8_UYVY, 94
- PixelFormatInfoSelector_YUV8_UYV, 93
- PixelSize_Bpp1, 94
- PixelSize_Bpp10, 94
- PixelSize_Bpp12, 94
- PixelSize_Bpp14, 94
- PixelSize_Bpp16, 94
- PixelSize_Bpp2, 94
- PixelSize_Bpp20, 94
- PixelSize_Bpp24, 94
- PixelSize_Bpp30, 94
- PixelSize_Bpp32, 94
- PixelSize_Bpp36, 94
- PixelSize_Bpp4, 94
- PixelSize_Bpp48, 94
- PixelSize_Bpp64, 94
- PixelSize_Bpp8, 94
- PixelSize_Bpp96, 94
- RegionDestination_Stream0, 94
- RegionDestination_Stream1, 94
- RegionDestination_Stream2, 94
- RegionMode_Off, 95
- RegionMode_On, 95
- RegionSelector_All, 95
- RegionSelector_Region0, 95
- RegionSelector_Region1, 95
- RegionSelector_Region2, 95
- RgbTransformLightSource_Cloudy6500K, 95
- RgbTransformLightSource_CoolFluorescent4000K, 95
- RgbTransformLightSource_Custom, 95
- RgbTransformLightSource_Daylight5000K, 95
- RgbTransformLightSource_General, 95
- RgbTransformLightSource_Shade8000K, 95
- RgbTransformLightSource_Tungsten2800K, 95
- RgbTransformLightSource_WarmFluorescent3000K, 95
- Scan3dCoordinateReferenceSelector_RotationX, 96
- Scan3dCoordinateReferenceSelector_RotationY, 96
- Scan3dCoordinateReferenceSelector_RotationZ, 96
- Scan3dCoordinateReferenceSelector_TranslationX, 96
- Scan3dCoordinateReferenceSelector_TranslationY, 96
- Scan3dCoordinateReferenceSelector_TranslationZ, 96
- Scan3dCoordinateSelector_CoordinateA, 96
- Scan3dCoordinateSelector_CoordinateB, 96
- Scan3dCoordinateSelector_CoordinateC, 96

- Scan3dCoordinateSystem_Cartesian, 96
- Scan3dCoordinateSystem_Cylindrical, 96
- Scan3dCoordinateSystem_Spherical, 96
- Scan3dCoordinateSystemReference_Anchor, 96
- Scan3dCoordinateSystemReference_Transformed, 96
- Scan3dCoordinateTransformSelector_RotationX, 97
- Scan3dCoordinateTransformSelector_RotationY, 97
- Scan3dCoordinateTransformSelector_RotationZ, 97
- Scan3dCoordinateTransformSelector_TranslationX, 97
- Scan3dCoordinateTransformSelector_TranslationY, 97
- Scan3dCoordinateTransformSelector_TranslationZ, 97
- Scan3dDistanceUnit_Inch, 97
- Scan3dDistanceUnit_Millimeter, 97
- Scan3dOutputMode_CalibratedABC_Grid, 97
- Scan3dOutputMode_CalibratedABC_PointCloud, 97
- Scan3dOutputMode_CalibratedAC_Linescan, 97
- Scan3dOutputMode_CalibratedAC, 97
- Scan3dOutputMode_CalibratedC_Linescan, 97
- Scan3dOutputMode_CalibratedC, 97
- Scan3dOutputMode_DisparityC_Linescan, 98
- Scan3dOutputMode_DisparityC, 98
- Scan3dOutputMode_RectifiedC_Linescan, 97
- Scan3dOutputMode_RectifiedC, 97
- Scan3dOutputMode_UncalibratedC, 97
- SensorDigitizationTaps_Eight, 98
- SensorDigitizationTaps_Four, 98
- SensorDigitizationTaps_One, 98
- SensorDigitizationTaps_Ten, 98
- SensorDigitizationTaps_Three, 98
- SensorDigitizationTaps_Two, 98
- SensorShutterMode_Global, 98
- SensorShutterMode_GlobalReset, 98
- SensorShutterMode_Rolling, 98
- SensorTaps_Eight, 98
- SensorTaps_Four, 98
- SensorTaps_One, 98
- SensorTaps_Ten, 98
- SensorTaps_Three, 98
- SensorTaps_Two, 98
- SequencerConfigurationMode_Off, 99
- SequencerConfigurationMode_On, 99
- SequencerConfigurationValid_No, 99
- SequencerConfigurationValid_Yes, 99
- SequencerMode_Off, 99
- SequencerMode_On, 99
- SequencerSetValid_No, 99
- SequencerSetValid_Yes, 99
- SequencerTriggerActivation_AnyEdge, 99
- SequencerTriggerActivation_FallingEdge, 99
- SequencerTriggerActivation_LevelHigh, 99
- SequencerTriggerActivation_LevelLow, 99
- SequencerTriggerActivation_RisingEdge, 99
- SequencerTriggerSource_FrameStart, 100
- SequencerTriggerSource_Off, 100
- SerialPortBaudRate_Baud115200, 100
- SerialPortBaudRate_Baud1200, 100
- SerialPortBaudRate_Baud14400, 100
- SerialPortBaudRate_Baud19200, 100
- SerialPortBaudRate_Baud230400, 100
- SerialPortBaudRate_Baud2400, 100
- SerialPortBaudRate_Baud300, 100
- SerialPortBaudRate_Baud38400, 100
- SerialPortBaudRate_Baud460800, 100
- SerialPortBaudRate_Baud4800, 100
- SerialPortBaudRate_Baud57600, 100
- SerialPortBaudRate_Baud600, 100
- SerialPortBaudRate_Baud921600, 100
- SerialPortBaudRate_Baud9600, 100
- SerialPortParity_Even, 100
- SerialPortParity_Mark, 100
- SerialPortParity_None, 100
- SerialPortParity_Odd, 100
- SerialPortParity_Space, 100
- SerialPortSelector_SerialPort0, 101
- SerialPortSource_Line0, 101
- SerialPortSource_Line1, 101
- SerialPortSource_Line2, 101
- SerialPortSource_Line3, 101
- SerialPortSource_Off, 101
- SerialPortStopBits_Bits1, 101
- SerialPortStopBits_Bits1AndAHalf, 101
- SerialPortStopBits_Bits2, 101
- SoftwareSignalSelector_SoftwareSignal0, 101
- SoftwareSignalSelector_SoftwareSignal1, 101
- SoftwareSignalSelector_SoftwareSignal2, 101
- SourceSelector_All, 102
- SourceSelector_Source0, 102
- SourceSelector_Source1, 102
- SourceSelector_Source2, 102
- spinAcquisitionModeEnums, 43
- spinAcquisitionStatusSelectorEnums, 43
- spinActionUnconditionalModeEnums, 44
- spinAdcBitDepthEnums, 44
- spinAutoAlgorithmSelectorEnums, 44
- spinAutoExposureControlPriorityEnums, 44
- spinAutoExposureLightingModeEnums, 45
- spinAutoExposureMeteringModeEnums, 45
- spinAutoExposureTargetGreyValueAutoEnums, 45
- spinBalanceRatioSelectorEnums, 46
- spinBalanceWhiteAutoEnums, 46
- spinBalanceWhiteAutoProfileEnums, 46
- spinBinningHorizontalModeEnums, 46
- spinBinningSelectorEnums, 47
- spinBinningVerticalModeEnums, 47
- spinBlackLevelAutoBalanceEnums, 47
- spinBlackLevelAutoEnums, 47
- spinBlackLevelSelectorEnums, 48
- spinChunkBlackLevelSelectorEnums, 48

- spinChunkCounterSelectorEnums, 48
- spinChunkEncoderSelectorEnums, 48
- spinChunkEncoderStatusEnums, 49
- spinChunkExposureTimeSelectorEnums, 49
- spinChunkGainSelectorEnums, 49
- spinChunkImageComponentEnums, 50
- spinChunkPixelFormatEnums, 50
- spinChunkRegionIDEnums, 50
- spinChunkScan3dCoordinateReferenceSelector↔
Enums, 51
- spinChunkScan3dCoordinateSelectorEnums, 51
- spinChunkScan3dCoordinateSystemEnums, 51
- spinChunkScan3dCoordinateSystemReference↔
Enums, 51
- spinChunkScan3dCoordinateTransformSelector↔
Enums, 52
- spinChunkScan3dDistanceUnitEnums, 52
- spinChunkScan3dOutputModeEnums, 52
- spinChunkSelectorEnums, 53
- spinChunkSourceIDEnums, 54
- spinChunkTimerSelectorEnums, 54
- spinChunkTransferStreamIDEnums, 54
- spinCIConfigurationEnums, 54
- spinCITimeSlotsCountEnums, 55
- spinColorTransformationSelectorEnums, 55
- spinColorTransformationValueSelectorEnums, 55
- spinCounterEventActivationEnums, 56
- spinCounterEventSourceEnums, 56
- spinCounterResetActivationEnums, 57
- spinCounterResetSourceEnums, 57
- spinCounterSelectorEnums, 57
- spinCounterStatusEnums, 58
- spinCounterTriggerActivationEnums, 58
- spinCounterTriggerSourceEnums, 58
- spinCxpConnectionTestModeEnums, 59
- spinCxpLinkConfigurationEnums, 59
- spinCxpLinkConfigurationPreferredEnums, 60
- spinCxpLinkConfigurationStatusEnums, 61
- spinCxpPoCxpStatusEnums, 62
- spinDecimationHorizontalModeEnums, 62
- spinDecimationSelectorEnums, 62
- spinDecimationVerticalModeEnums, 63
- spinDefectCorrectionModeEnums, 63
- spinDeinterlacingEnums, 63
- spinDeviceCharacterSetEnums, 63
- spinDeviceClockSelectorEnums, 64
- spinDeviceConnectionStatusEnums, 64
- spinDeviceIndicatorModeEnums, 64
- spinDeviceLinkHeartbeatModeEnums, 64
- spinDeviceLinkThroughputLimitModeEnums, 65
- spinDevicePowerSupplySelectorEnums, 65
- spinDeviceRegistersEndiannessEnums, 65
- spinDeviceScanTypeEnums, 65
- spinDeviceSerialPortBaudRateEnums, 65
- spinDeviceSerialPortSelectorEnums, 66
- spinDeviceStreamChannelEndiannessEnums, 66
- spinDeviceStreamChannelTypeEnums, 66
- spinDeviceTLTypeEnums, 68
- spinDeviceTapGeometryEnums, 66
- spinDeviceTemperatureSelectorEnums, 68
- spinDeviceTypeEnums, 68
- spinEncoderModeEnums, 68
- spinEncoderOutputModeEnums, 69
- spinEncoderResetActivationEnums, 69
- spinEncoderResetSourceEnums, 69
- spinEncoderSelectorEnums, 70
- spinEncoderSourceAEnums, 71
- spinEncoderSourceBEnums, 71
- spinEncoderStatusEnums, 71
- spinEventNotificationEnums, 71
- spinEventSelectorEnums, 72
- spinExposureActiveModeEnums, 72
- spinExposureAutoEnums, 72
- spinExposureModeEnums, 72
- spinExposureTimeModeEnums, 73
- spinExposureTimeSelectorEnums, 73
- spinFileOpenModeEnums, 73
- spinFileOperationSelectorEnums, 74
- spinFileOperationStatusEnums, 74
- spinFileSelectorEnums, 74
- spinGainAutoBalanceEnums, 74
- spinGainAutoEnums, 75
- spinGainSelectorEnums, 75
- spinGevCCPEnums, 75
- spinGevCurrentPhysicalLinkConfigurationEnums,
75
- spinGevGVCPExtendedStatusCodesSelector↔
Enums, 76
- spinGevGVSPExtendedIDModeEnums, 76
- spinGevIEEE1588ClockAccuracyEnums, 76
- spinGevIEEE1588ModeEnums, 76
- spinGevIEEE1588StatusEnums, 77
- spinGevIPConfigurationStatusEnums, 77
- spinGevPhysicalLinkConfigurationEnums, 77
- spinGevSupportedOptionSelectorEnums, 77
- spinImageComponentSelectorEnums, 78
- spinImageCompressionJPEGFormatOption↔
Enums, 79
- spinImageCompressionModeEnums, 79
- spinImageCompressionRateOptionEnums, 79
- spinLUTSelectorEnums, 82
- spinLineFormatEnums, 79
- spinLineInputFilterSelectorEnums, 80
- spinLineModeEnums, 80
- spinLineSelectorEnums, 80
- spinLineSourceEnums, 80
- spinLogicBlockLUTInputActivationEnums, 81
- spinLogicBlockLUTInputSelectorEnums, 81
- spinLogicBlockLUTInputSourceEnums, 81
- spinLogicBlockLUTSelectorEnums, 82
- spinLogicBlockSelectorEnums, 82
- spinPixelColorFilterEnums, 83
- spinPixelFormatEnums, 83
- spinPixelFormatInfoSelectorEnums, 88
- spinPixelSizeEnums, 94
- spinRegionDestinationEnums, 94

- spinRegionModeEnums, 94
- spinRegionSelectorEnums, 95
- spinRgbTransformLightSourceEnums, 95
- spinScan3dCoordinateReferenceSelectorEnums, 95
- spinScan3dCoordinateSelectorEnums, 96
- spinScan3dCoordinateSystemEnums, 96
- spinScan3dCoordinateSystemReferenceEnums, 96
- spinScan3dCoordinateTransformSelectorEnums, 96
- spinScan3dDistanceUnitEnums, 97
- spinScan3dOutputModeEnums, 97
- spinSensorDigitizationTapsEnums, 98
- spinSensorShutterModeEnums, 98
- spinSensorTapsEnums, 98
- spinSequencerConfigurationModeEnums, 98
- spinSequencerConfigurationValidEnums, 99
- spinSequencerModeEnums, 99
- spinSequencerSetValidEnums, 99
- spinSequencerTriggerActivationEnums, 99
- spinSequencerTriggerSourceEnums, 99
- spinSerialPortBaudRateEnums, 100
- spinSerialPortParityEnums, 100
- spinSerialPortSelectorEnums, 100
- spinSerialPortSourceEnums, 101
- spinSerialPortStopBitsEnums, 101
- spinSoftwareSignalSelectorEnums, 101
- spinSourceSelectorEnums, 101
- spinTestPatternEnums, 102
- spinTestPatternGeneratorSelectorEnums, 102
- spinTimerSelectorEnums, 102
- spinTimerStatusEnums, 102
- spinTimerTriggerActivationEnums, 103
- spinTimerTriggerSourceEnums, 103
- spinTransferComponentSelectorEnums, 104
- spinTransferControlModeEnums, 105
- spinTransferOperationModeEnums, 105
- spinTransferQueueModeEnums, 105
- spinTransferSelectorEnums, 105
- spinTransferStatusSelectorEnums, 106
- spinTransferTriggerActivationEnums, 106
- spinTransferTriggerModeEnums, 106
- spinTransferTriggerSelectorEnums, 107
- spinTransferTriggerSourceEnums, 107
- spinTriggerActivationEnums, 108
- spinTriggerModeEnums, 108
- spinTriggerOverlapEnums, 108
- spinTriggerSelectorEnums, 109
- spinTriggerSourceEnums, 109
- spinUserOutputSelectorEnums, 109
- spinUserSetDefaultEnums, 110
- spinUserSetSelectorEnums, 110
- spinWhiteClipSelectorEnums, 110
- TestPattern_Increment, 102
- TestPattern_Off, 102
- TestPattern_SensorTestPattern, 102
- TestPatternGeneratorSelector_PipelineStart, 102
- TestPatternGeneratorSelector_Sensor, 102
- TimerSelector_Timer0, 102
- TimerSelector_Timer1, 102
- TimerSelector_Timer2, 102
- TimerStatus_TimerActive, 103
- TimerStatus_TimerCompleted, 103
- TimerStatus_TimerIdle, 103
- TimerStatus_TimerTriggerWait, 103
- TimerTriggerActivation_AnyEdge, 103
- TimerTriggerActivation_FallingEdge, 103
- TimerTriggerActivation_LevelHigh, 103
- TimerTriggerActivation_LevelLow, 103
- TimerTriggerActivation_RisingEdge, 103
- TimerTriggerSource_AcquisitionEnd, 103
- TimerTriggerSource_AcquisitionStart, 103
- TimerTriggerSource_AcquisitionTrigger, 103
- TimerTriggerSource_Action0, 104
- TimerTriggerSource_Action1, 104
- TimerTriggerSource_Action2, 104
- TimerTriggerSource_Counter0End, 104
- TimerTriggerSource_Counter0Start, 104
- TimerTriggerSource_Counter1End, 104
- TimerTriggerSource_Counter1Start, 104
- TimerTriggerSource_Counter2End, 104
- TimerTriggerSource_Counter2Start, 104
- TimerTriggerSource_Encoder0, 104
- TimerTriggerSource_Encoder1, 104
- TimerTriggerSource_Encoder2, 104
- TimerTriggerSource_ExposureEnd, 103
- TimerTriggerSource_ExposureStart, 103
- TimerTriggerSource_FrameBurstEnd, 103
- TimerTriggerSource_FrameBurstStart, 103
- TimerTriggerSource_FrameEnd, 103
- TimerTriggerSource_FrameStart, 103
- TimerTriggerSource_FrameTrigger, 103
- TimerTriggerSource_Line0, 103
- TimerTriggerSource_Line1, 104
- TimerTriggerSource_Line2, 104
- TimerTriggerSource_LineEnd, 103
- TimerTriggerSource_LineStart, 103
- TimerTriggerSource_LineTrigger, 103
- TimerTriggerSource_LinkTrigger0, 104
- TimerTriggerSource_LinkTrigger1, 104
- TimerTriggerSource_LinkTrigger2, 104
- TimerTriggerSource_Off, 103
- TimerTriggerSource_SoftwareSignal0, 104
- TimerTriggerSource_SoftwareSignal1, 104
- TimerTriggerSource_SoftwareSignal2, 104
- TimerTriggerSource_Timer0End, 104
- TimerTriggerSource_Timer0Start, 104
- TimerTriggerSource_Timer1End, 104
- TimerTriggerSource_Timer1Start, 104
- TimerTriggerSource_Timer2End, 104
- TimerTriggerSource_Timer2Start, 104
- TimerTriggerSource_UserOutput0, 104
- TimerTriggerSource_UserOutput1, 104
- TimerTriggerSource_UserOutput2, 104
- TransferComponentSelector_All, 105

- TransferComponentSelector_Blue, 105
- TransferComponentSelector_Green, 105
- TransferComponentSelector_Red, 105
- TransferControlMode_Automatic, 105
- TransferControlMode_Basic, 105
- TransferControlMode_UserControlled, 105
- TransferOperationMode_Continuous, 105
- TransferOperationMode_MultiBlock, 105
- TransferQueueMode_FirstInFirstOut, 105
- TransferSelector_All, 106
- TransferSelector_Stream0, 106
- TransferSelector_Stream1, 106
- TransferSelector_Stream2, 106
- TransferStatusSelector_Paused, 106
- TransferStatusSelector_QueueOverflow, 106
- TransferStatusSelector_Stopped, 106
- TransferStatusSelector_Stopping, 106
- TransferStatusSelector_Streaming, 106
- TransferTriggerActivation_AnyEdge, 106
- TransferTriggerActivation_FallingEdge, 106
- TransferTriggerActivation_LevelHigh, 106
- TransferTriggerActivation_LevelLow, 106
- TransferTriggerActivation_RisingEdge, 106
- TransferTriggerMode_Off, 107
- TransferTriggerMode_On, 107
- TransferTriggerSelector_TransferAbort, 107
- TransferTriggerSelector_TransferActive, 107
- TransferTriggerSelector_TransferBurstStart, 107
- TransferTriggerSelector_TransferBurstStop, 107
- TransferTriggerSelector_TransferPause, 107
- TransferTriggerSelector_TransferResume, 107
- TransferTriggerSelector_TransferStart, 107
- TransferTriggerSelector_TransferStop, 107
- TransferTriggerSource_Action0, 108
- TransferTriggerSource_Action1, 108
- TransferTriggerSource_Action2, 108
- TransferTriggerSource_Counter0End, 107
- TransferTriggerSource_Counter0Start, 107
- TransferTriggerSource_Counter1End, 107
- TransferTriggerSource_Counter1Start, 107
- TransferTriggerSource_Counter2End, 108
- TransferTriggerSource_Counter2Start, 107
- TransferTriggerSource_Line0, 107
- TransferTriggerSource_Line1, 107
- TransferTriggerSource_Line2, 107
- TransferTriggerSource_SoftwareSignal0, 108
- TransferTriggerSource_SoftwareSignal1, 108
- TransferTriggerSource_SoftwareSignal2, 108
- TransferTriggerSource_Timer0End, 108
- TransferTriggerSource_Timer0Start, 108
- TransferTriggerSource_Timer1End, 108
- TransferTriggerSource_Timer1Start, 108
- TransferTriggerSource_Timer2End, 108
- TransferTriggerSource_Timer2Start, 108
- TriggerActivation_AnyEdge, 108
- TriggerActivation_FallingEdge, 108
- TriggerActivation_LevelHigh, 108
- TriggerActivation_LevelLow, 108
- TriggerActivation_RisingEdge, 108
- TriggerMode_Off, 108
- TriggerMode_On, 108
- TriggerOverlap_Off, 109
- TriggerOverlap_PreviousFrame, 109
- TriggerOverlap_ReadOut, 109
- TriggerSelector_AcquisitionStart, 109
- TriggerSelector_FrameBurstStart, 109
- TriggerSelector_FrameStart, 109
- TriggerSource_Action0, 109
- TriggerSource_Counter0End, 109
- TriggerSource_Counter0Start, 109
- TriggerSource_Counter1End, 109
- TriggerSource_Counter1Start, 109
- TriggerSource_Line0, 109
- TriggerSource_Line1, 109
- TriggerSource_Line2, 109
- TriggerSource_Line3, 109
- TriggerSource_LogicBlock0, 109
- TriggerSource_LogicBlock1, 109
- TriggerSource_Software, 109
- TriggerSource_UserOutput0, 109
- TriggerSource_UserOutput1, 109
- TriggerSource_UserOutput2, 109
- TriggerSource_UserOutput3, 109
- UNKNOWN_PIXELFORMAT, 88
- UserOutputSelector_UserOutput0, 110
- UserOutputSelector_UserOutput1, 110
- UserOutputSelector_UserOutput2, 110
- UserOutputSelector_UserOutput3, 110
- UserSetDefault_Default, 110
- UserSetDefault_UserSet0, 110
- UserSetDefault_UserSet1, 110
- UserSetSelector_Default, 110
- UserSetSelector_UserSet0, 110
- UserSetSelector_UserSet1, 110
- WhiteClipSelector_All, 110
- WhiteClipSelector_Blue, 110
- WhiteClipSelector_Green, 110
- WhiteClipSelector_Red, 110
- WhiteClipSelector_Tap1, 110
- WhiteClipSelector_Tap2, 110
- WhiteClipSelector_U, 110
- WhiteClipSelector_V, 110
- WhiteClipSelector_Y, 110
- CameraList Access, 133
 - spinCameraListAppend, 133
 - spinCameraListClear, 134
 - spinCameraListCreateEmpty, 134
 - spinCameraListDestroy, 134
 - spinCameraListGet, 135
 - spinCameraListGetBySerial, 135
 - spinCameraListGetSize, 136
 - spinCameraListRemove, 136
 - spinCameraListRemoveBySerial, 136
- CategoryNode
 - Spinnaker C GenICam Enumerations, 267
- Chunk data access, 200

- spinImageChunkDataGetFloatValue, 200
- spinImageChunkDataGetIntValue, 200
- Chunk Data Structures, 111
- ChunkBlackLevel
 - quickSpin, 296
- ChunkBlackLevelSelector
 - quickSpin, 296
- ChunkBlackLevelSelector_All
 - Camera Enumerations, 48
- ChunkCRC
 - quickSpin, 297
- ChunkCounterSelector
 - quickSpin, 297
- ChunkCounterSelector_Counter0
 - Camera Enumerations, 48
- ChunkCounterSelector_Counter1
 - Camera Enumerations, 48
- ChunkCounterSelector_Counter2
 - Camera Enumerations, 48
- ChunkCounterValue
 - quickSpin, 297
- ChunkEnable
 - quickSpin, 297
- ChunkEncoderSelector
 - quickSpin, 297
- ChunkEncoderSelector_Encoder0
 - Camera Enumerations, 49
- ChunkEncoderSelector_Encoder1
 - Camera Enumerations, 49
- ChunkEncoderSelector_Encoder2
 - Camera Enumerations, 49
- ChunkEncoderStatus
 - quickSpin, 297
- ChunkEncoderStatus_EncoderDown
 - Camera Enumerations, 49
- ChunkEncoderStatus_EncoderIdle
 - Camera Enumerations, 49
- ChunkEncoderStatus_EncoderStatic
 - Camera Enumerations, 49
- ChunkEncoderStatus_EncoderUp
 - Camera Enumerations, 49
- ChunkEncoderValue
 - quickSpin, 297
- ChunkExposureEndLineStatusAll
 - quickSpin, 297
- ChunkExposureTime
 - quickSpin, 297
- ChunkExposureTimeSelector
 - quickSpin, 297
- ChunkExposureTimeSelector_Blue
 - Camera Enumerations, 49
- ChunkExposureTimeSelector_Common
 - Camera Enumerations, 49
- ChunkExposureTimeSelector_Cyan
 - Camera Enumerations, 49
- ChunkExposureTimeSelector_Green
 - Camera Enumerations, 49
- ChunkExposureTimeSelector_Infrared
- Camera Enumerations, 49
- ChunkExposureTimeSelector_Magenta
 - Camera Enumerations, 49
- ChunkExposureTimeSelector_Red
 - Camera Enumerations, 49
- ChunkExposureTimeSelector_Stage1
 - Camera Enumerations, 49
- ChunkExposureTimeSelector_Stage2
 - Camera Enumerations, 49
- ChunkExposureTimeSelector_Ultraviolet
 - Camera Enumerations, 49
- ChunkExposureTimeSelector_Yellow
 - Camera Enumerations, 49
- ChunkFrameID
 - quickSpin, 297
- ChunkGain
 - quickSpin, 297
- ChunkGainSelector
 - quickSpin, 297
- ChunkGainSelector_All
 - Camera Enumerations, 50
- ChunkGainSelector_Blue
 - Camera Enumerations, 50
- ChunkGainSelector_Green
 - Camera Enumerations, 50
- ChunkGainSelector_Red
 - Camera Enumerations, 50
- ChunkHeight
 - quickSpin, 297
- ChunkImage
 - quickSpin, 297
- ChunkImageComponent
 - quickSpin, 297
- ChunkImageComponent_Color
 - Camera Enumerations, 50
- ChunkImageComponent_Confidence
 - Camera Enumerations, 50
- ChunkImageComponent_Disparity
 - Camera Enumerations, 50
- ChunkImageComponent_Infrared
 - Camera Enumerations, 50
- ChunkImageComponent_Intensity
 - Camera Enumerations, 50
- ChunkImageComponent_Range
 - Camera Enumerations, 50
- ChunkImageComponent_Scatter
 - Camera Enumerations, 50
- ChunkImageComponent_Ultraviolet
 - Camera Enumerations, 50
- ChunkInferenceConfidence
 - quickSpin, 297
- ChunkInferenceResult
 - quickSpin, 297
- ChunkLinePitch
 - quickSpin, 297
- ChunkLineStatusAll
 - quickSpin, 297
- ChunkModeActive

- quickSpin, [297](#)
- ChunkOffsetX
 - quickSpin, [297](#)
- ChunkOffsetY
 - quickSpin, [297](#)
- ChunkPartSelector
 - quickSpin, [298](#)
- ChunkPixelDynamicRangeMax
 - quickSpin, [298](#)
- ChunkPixelDynamicRangeMin
 - quickSpin, [298](#)
- ChunkPixelFormat
 - quickSpin, [298](#)
- ChunkPixelFormat_BayerBG8
 - Camera Enumerations, [50](#)
- ChunkPixelFormat_BayerGB8
 - Camera Enumerations, [50](#)
- ChunkPixelFormat_BayerGR8
 - Camera Enumerations, [50](#)
- ChunkPixelFormat_BayerRG8
 - Camera Enumerations, [50](#)
- ChunkPixelFormat_Mono12Packed
 - Camera Enumerations, [50](#)
- ChunkPixelFormat_Mono16
 - Camera Enumerations, [50](#)
- ChunkPixelFormat_Mono8
 - Camera Enumerations, [50](#)
- ChunkPixelFormat_RGB8Packed
 - Camera Enumerations, [50](#)
- ChunkPixelFormat_YCbCr601_422_8_CbYCrY
 - Camera Enumerations, [50](#)
- ChunkPixelFormat_YUV422Packed
 - Camera Enumerations, [50](#)
- ChunkRegionID_Region0
 - Camera Enumerations, [51](#)
- ChunkRegionID_Region1
 - Camera Enumerations, [51](#)
- ChunkRegionID_Region2
 - Camera Enumerations, [51](#)
- ChunkRegionID
 - quickSpin, [298](#)
- ChunkScan3dAxisMax
 - quickSpin, [298](#)
- ChunkScan3dAxisMin
 - quickSpin, [298](#)
- ChunkScan3dCoordinateOffset
 - quickSpin, [298](#)
- ChunkScan3dCoordinateReferenceSelector
 - quickSpin, [298](#)
- ChunkScan3dCoordinateReferenceSelector_RotationX
 - Camera Enumerations, [51](#)
- ChunkScan3dCoordinateReferenceSelector_RotationY
 - Camera Enumerations, [51](#)
- ChunkScan3dCoordinateReferenceSelector_RotationZ
 - Camera Enumerations, [51](#)
- ChunkScan3dCoordinateReferenceSelector_TranslationX
 - Camera Enumerations, [51](#)
- ChunkScan3dCoordinateReferenceSelector_TranslationY
 - Camera Enumerations, [51](#)
- Camera Enumerations, [51](#)
- ChunkScan3dCoordinateReferenceSelector_TranslationZ
 - Camera Enumerations, [51](#)
- ChunkScan3dCoordinateReferenceValue
 - quickSpin, [298](#)
- ChunkScan3dCoordinateScale
 - quickSpin, [298](#)
- ChunkScan3dCoordinateSelector
 - quickSpin, [298](#)
- ChunkScan3dCoordinateSelector_CoordinateA
 - Camera Enumerations, [51](#)
- ChunkScan3dCoordinateSelector_CoordinateB
 - Camera Enumerations, [51](#)
- ChunkScan3dCoordinateSelector_CoordinateC
 - Camera Enumerations, [51](#)
- ChunkScan3dCoordinateSystem
 - quickSpin, [298](#)
- ChunkScan3dCoordinateSystem_Cartesian
 - Camera Enumerations, [51](#)
- ChunkScan3dCoordinateSystem_Cylindrical
 - Camera Enumerations, [51](#)
- ChunkScan3dCoordinateSystem_Spherical
 - Camera Enumerations, [51](#)
- ChunkScan3dCoordinateSystemReference
 - quickSpin, [298](#)
- ChunkScan3dCoordinateSystemReference_Anchor
 - Camera Enumerations, [52](#)
- ChunkScan3dCoordinateSystemReference_Transformed
 - Camera Enumerations, [52](#)
- ChunkScan3dCoordinateTransformSelector
 - quickSpin, [298](#)
- ChunkScan3dCoordinateTransformSelector_RotationX
 - Camera Enumerations, [52](#)
- ChunkScan3dCoordinateTransformSelector_RotationY
 - Camera Enumerations, [52](#)
- ChunkScan3dCoordinateTransformSelector_RotationZ
 - Camera Enumerations, [52](#)
- ChunkScan3dCoordinateTransformSelector_TranslationX
 - Camera Enumerations, [52](#)
- ChunkScan3dCoordinateTransformSelector_TranslationY
 - Camera Enumerations, [52](#)
- ChunkScan3dCoordinateTransformSelector_TranslationZ
 - Camera Enumerations, [52](#)
- ChunkScan3dDistanceUnit
 - quickSpin, [298](#)
- ChunkScan3dDistanceUnit_Inch
 - Camera Enumerations, [52](#)
- ChunkScan3dDistanceUnit_Millimeter
 - Camera Enumerations, [52](#)
- ChunkScan3dInvalidDataFlag
 - quickSpin, [298](#)
- ChunkScan3dInvalidDataValue
 - quickSpin, [298](#)
- ChunkScan3dOutputMode
 - quickSpin, [298](#)
- ChunkScan3dOutputMode_CalibratedABC_Grid
 - Camera Enumerations, [53](#)
- ChunkScan3dOutputMode_CalibratedABC_PointCloud

- Camera Enumerations, [53](#)
- ChunkScan3dOutputMode_CalibratedAC_Linescan
 - Camera Enumerations, [53](#)
- ChunkScan3dOutputMode_CalibratedAC
 - Camera Enumerations, [53](#)
- ChunkScan3dOutputMode_CalibratedC_Linescan
 - Camera Enumerations, [53](#)
- ChunkScan3dOutputMode_CalibratedC
 - Camera Enumerations, [53](#)
- ChunkScan3dOutputMode_DisparityC_Linescan
 - Camera Enumerations, [53](#)
- ChunkScan3dOutputMode_DisparityC
 - Camera Enumerations, [53](#)
- ChunkScan3dOutputMode_RectifiedC_Linescan
 - Camera Enumerations, [53](#)
- ChunkScan3dOutputMode_RectifiedC
 - Camera Enumerations, [53](#)
- ChunkScan3dOutputMode_UncalibratedC
 - Camera Enumerations, [53](#)
- ChunkScan3dTransformValue
 - [quickSpin](#), [298](#)
- ChunkScanLineSelector
 - [quickSpin](#), [298](#)
- ChunkSelector
 - [quickSpin](#), [298](#)
- ChunkSelector_BlackLevel
 - Camera Enumerations, [54](#)
- ChunkSelector_CRC
 - Camera Enumerations, [53](#)
- ChunkSelector_ExposureEndLineStatusAll
 - Camera Enumerations, [54](#)
- ChunkSelector_ExposureTime
 - Camera Enumerations, [53](#)
- ChunkSelector_FrameID
 - Camera Enumerations, [53](#)
- ChunkSelector_Gain
 - Camera Enumerations, [54](#)
- ChunkSelector_Height
 - Camera Enumerations, [53](#)
- ChunkSelector_Image
 - Camera Enumerations, [53](#)
- ChunkSelector_OffsetX
 - Camera Enumerations, [53](#)
- ChunkSelector_OffsetY
 - Camera Enumerations, [53](#)
- ChunkSelector_PixelFormat
 - Camera Enumerations, [54](#)
- ChunkSelector_SequencerSetActive
 - Camera Enumerations, [54](#)
- ChunkSelector_SerialData
 - Camera Enumerations, [54](#)
- ChunkSelector_Timestamp
 - Camera Enumerations, [54](#)
- ChunkSelector_Width
 - Camera Enumerations, [53](#)
- ChunkSequencerSetActive
 - [quickSpin](#), [298](#)
- ChunkSerialData
 - [quickSpin](#), [299](#)
- ChunkSerialDataLength
 - [quickSpin](#), [299](#)
- ChunkSerialReceiveOverflow
 - [quickSpin](#), [299](#)
- ChunkSourceID_Source0
 - Camera Enumerations, [54](#)
- ChunkSourceID_Source1
 - Camera Enumerations, [54](#)
- ChunkSourceID_Source2
 - Camera Enumerations, [54](#)
- ChunkSourceID
 - [quickSpin](#), [299](#)
- ChunkStreamChannelID
 - [quickSpin](#), [299](#)
- ChunkTimerSelector
 - [quickSpin](#), [299](#)
- ChunkTimerSelector_Timer0
 - Camera Enumerations, [54](#)
- ChunkTimerSelector_Timer1
 - Camera Enumerations, [54](#)
- ChunkTimerSelector_Timer2
 - Camera Enumerations, [54](#)
- ChunkTimerValue
 - [quickSpin](#), [299](#)
- ChunkTimestamp
 - [quickSpin](#), [299](#)
- ChunkTimestampLatchValue
 - [quickSpin](#), [299](#)
- ChunkTransferBlockID
 - [quickSpin](#), [299](#)
- ChunkTransferQueueCurrentBlockCount
 - [quickSpin](#), [299](#)
- ChunkTransferStreamID_Stream0
 - Camera Enumerations, [54](#)
- ChunkTransferStreamID_Stream1
 - Camera Enumerations, [54](#)
- ChunkTransferStreamID_Stream2
 - Camera Enumerations, [54](#)
- ChunkTransferStreamID_Stream3
 - Camera Enumerations, [54](#)
- ChunkTransferStreamID
 - [quickSpin](#), [299](#)
- ChunkWidth
 - [quickSpin](#), [299](#)
- CL
 - Spinnaker C GenICam Enumerations, [268](#)
- ClConfiguration
 - [quickSpin](#), [299](#)
- ClConfiguration_Base
 - Camera Enumerations, [55](#)
- ClConfiguration_DualBase
 - Camera Enumerations, [55](#)
- ClConfiguration_EightyBit
 - Camera Enumerations, [55](#)
- ClConfiguration_Full
 - Camera Enumerations, [55](#)
- ClConfiguration_Medium

- Camera Enumerations, [55](#)
- CITimeSlotsCount
 - quickSpin, [299](#)
- CITimeSlotsCount_One
 - Camera Enumerations, [55](#)
- CITimeSlotsCount_Three
 - Camera Enumerations, [55](#)
- CITimeSlotsCount_Two
 - Camera Enumerations, [55](#)
- ColorTransformationEnable
 - quickSpin, [299](#)
- ColorTransformationSelector
 - quickSpin, [299](#)
- ColorTransformationSelector_RGBtoRGB
 - Camera Enumerations, [55](#)
- ColorTransformationSelector_RGBtoYUV
 - Camera Enumerations, [55](#)
- ColorTransformationValue
 - quickSpin, [299](#)
- ColorTransformationValueSelector
 - quickSpin, [299](#)
- ColorTransformationValueSelector_Gain00
 - Camera Enumerations, [56](#)
- ColorTransformationValueSelector_Gain01
 - Camera Enumerations, [56](#)
- ColorTransformationValueSelector_Gain02
 - Camera Enumerations, [56](#)
- ColorTransformationValueSelector_Gain10
 - Camera Enumerations, [56](#)
- ColorTransformationValueSelector_Gain11
 - Camera Enumerations, [56](#)
- ColorTransformationValueSelector_Gain12
 - Camera Enumerations, [56](#)
- ColorTransformationValueSelector_Gain20
 - Camera Enumerations, [56](#)
- ColorTransformationValueSelector_Gain21
 - Camera Enumerations, [56](#)
- ColorTransformationValueSelector_Gain22
 - Camera Enumerations, [56](#)
- ColorTransformationValueSelector_Offset0
 - Camera Enumerations, [56](#)
- ColorTransformationValueSelector_Offset1
 - Camera Enumerations, [56](#)
- ColorTransformationValueSelector_Offset2
 - Camera Enumerations, [56](#)
- CommandNode
 - Spinnaker C GenICam Enumerations, [267](#)
- compression
 - spinTIFFOption, [341](#)
- compressionLevel
 - spinPNGOption, [339](#)
- CompressionRatio
 - quickSpin, [299](#)
- CounterDelay
 - quickSpin, [299](#)
- CounterDuration
 - quickSpin, [299](#)
- CounterEventActivation
 - quickSpin, [299](#)
- CounterEventActivation_AnyEdge
 - Camera Enumerations, [56](#)
- CounterEventActivation_FallingEdge
 - Camera Enumerations, [56](#)
- CounterEventActivation_LevelHigh
 - Camera Enumerations, [56](#)
- CounterEventActivation_LevelLow
 - Camera Enumerations, [56](#)
- CounterEventActivation_RisingEdge
 - Camera Enumerations, [56](#)
- CounterEventSource
 - quickSpin, [300](#)
- CounterEventSource_Counter0End
 - Camera Enumerations, [57](#)
- CounterEventSource_Counter0Start
 - Camera Enumerations, [57](#)
- CounterEventSource_Counter1End
 - Camera Enumerations, [57](#)
- CounterEventSource_Counter1Start
 - Camera Enumerations, [57](#)
- CounterEventSource_ExposureEnd
 - Camera Enumerations, [57](#)
- CounterEventSource_ExposureStart
 - Camera Enumerations, [57](#)
- CounterEventSource_FrameTriggerWait
 - Camera Enumerations, [57](#)
- CounterEventSource_Line0
 - Camera Enumerations, [56](#)
- CounterEventSource_Line1
 - Camera Enumerations, [56](#)
- CounterEventSource_Line2
 - Camera Enumerations, [56](#)
- CounterEventSource_Line3
 - Camera Enumerations, [56](#)
- CounterEventSource_LogicBlock0
 - Camera Enumerations, [57](#)
- CounterEventSource_LogicBlock1
 - Camera Enumerations, [57](#)
- CounterEventSource_MHzTick
 - Camera Enumerations, [56](#)
- CounterEventSource_Off
 - Camera Enumerations, [56](#)
- CounterEventSource_UserOutput0
 - Camera Enumerations, [56](#)
- CounterEventSource_UserOutput1
 - Camera Enumerations, [56](#)
- CounterEventSource_UserOutput2
 - Camera Enumerations, [57](#)
- CounterEventSource_UserOutput3
 - Camera Enumerations, [57](#)
- CounterReset
 - quickSpin, [300](#)
- CounterResetActivation
 - quickSpin, [300](#)
- CounterResetActivation_AnyEdge
 - Camera Enumerations, [57](#)
- CounterResetActivation_FallingEdge

- Camera Enumerations, [57](#)
- CounterResetActivation_LevelHigh
 - Camera Enumerations, [57](#)
- CounterResetActivation_LevelLow
 - Camera Enumerations, [57](#)
- CounterResetActivation_RisingEdge
 - Camera Enumerations, [57](#)
- CounterResetSource
 - quickSpin, [300](#)
- CounterResetSource_Counter0End
 - Camera Enumerations, [57](#)
- CounterResetSource_Counter0Start
 - Camera Enumerations, [57](#)
- CounterResetSource_Counter1End
 - Camera Enumerations, [57](#)
- CounterResetSource_Counter1Start
 - Camera Enumerations, [57](#)
- CounterResetSource_ExposureEnd
 - Camera Enumerations, [57](#)
- CounterResetSource_ExposureStart
 - Camera Enumerations, [57](#)
- CounterResetSource_FrameTriggerWait
 - Camera Enumerations, [57](#)
- CounterResetSource_Line0
 - Camera Enumerations, [57](#)
- CounterResetSource_Line1
 - Camera Enumerations, [57](#)
- CounterResetSource_Line2
 - Camera Enumerations, [57](#)
- CounterResetSource_Line3
 - Camera Enumerations, [57](#)
- CounterResetSource_LogicBlock0
 - Camera Enumerations, [57](#)
- CounterResetSource_LogicBlock1
 - Camera Enumerations, [57](#)
- CounterResetSource_Off
 - Camera Enumerations, [57](#)
- CounterResetSource_UserOutput0
 - Camera Enumerations, [57](#)
- CounterResetSource_UserOutput1
 - Camera Enumerations, [57](#)
- CounterResetSource_UserOutput2
 - Camera Enumerations, [57](#)
- CounterResetSource_UserOutput3
 - Camera Enumerations, [57](#)
- CounterSelector
 - quickSpin, [300](#)
- CounterSelector_Counter0
 - Camera Enumerations, [58](#)
- CounterSelector_Counter1
 - Camera Enumerations, [58](#)
- CounterStatus
 - quickSpin, [300](#)
- CounterStatus_CounterActive
 - Camera Enumerations, [58](#)
- CounterStatus_CounterCompleted
 - Camera Enumerations, [58](#)
- CounterStatus_CounterIdle
 - Camera Enumerations, [58](#)
- CounterStatus_CounterOverflow
 - Camera Enumerations, [58](#)
- CounterStatus_CounterTriggerWait
 - Camera Enumerations, [58](#)
- CounterTriggerActivation
 - quickSpin, [300](#)
- CounterTriggerActivation_AnyEdge
 - Camera Enumerations, [58](#)
- CounterTriggerActivation_FallingEdge
 - Camera Enumerations, [58](#)
- CounterTriggerActivation_LevelHigh
 - Camera Enumerations, [58](#)
- CounterTriggerActivation_LevelLow
 - Camera Enumerations, [58](#)
- CounterTriggerActivation_RisingEdge
 - Camera Enumerations, [58](#)
- CounterTriggerSource
 - quickSpin, [300](#)
- CounterTriggerSource_Counter0End
 - Camera Enumerations, [59](#)
- CounterTriggerSource_Counter0Start
 - Camera Enumerations, [59](#)
- CounterTriggerSource_Counter1End
 - Camera Enumerations, [59](#)
- CounterTriggerSource_Counter1Start
 - Camera Enumerations, [59](#)
- CounterTriggerSource_ExposureEnd
 - Camera Enumerations, [59](#)
- CounterTriggerSource_ExposureStart
 - Camera Enumerations, [59](#)
- CounterTriggerSource_FrameTriggerWait
 - Camera Enumerations, [59](#)
- CounterTriggerSource_Line0
 - Camera Enumerations, [59](#)
- CounterTriggerSource_Line1
 - Camera Enumerations, [59](#)
- CounterTriggerSource_Line2
 - Camera Enumerations, [59](#)
- CounterTriggerSource_Line3
 - Camera Enumerations, [59](#)
- CounterTriggerSource_LogicBlock0
 - Camera Enumerations, [59](#)
- CounterTriggerSource_LogicBlock1
 - Camera Enumerations, [59](#)
- CounterTriggerSource_Off
 - Camera Enumerations, [59](#)
- CounterTriggerSource_UserOutput0
 - Camera Enumerations, [59](#)
- CounterTriggerSource_UserOutput1
 - Camera Enumerations, [59](#)
- CounterTriggerSource_UserOutput2
 - Camera Enumerations, [59](#)
- CounterTriggerSource_UserOutput3
 - Camera Enumerations, [59](#)
- CounterValue
 - quickSpin, [300](#)
- CounterValueAtReset

- quickSpin, [300](#)
- ctAllDependingNodes
 - Spinnaker C GenICam Enumerations, [266](#)
- ctAllTerminalNodes
 - Spinnaker C GenICam Enumerations, [266](#)
- ctDependingChildren
 - Spinnaker C GenICam Enumerations, [266](#)
- ctInvalidators
 - Spinnaker C GenICam Enumerations, [266](#)
- ctReadingChildren
 - Spinnaker C GenICam Enumerations, [266](#)
- ctWritingChildren
 - Spinnaker C GenICam Enumerations, [266](#)
- Custom
 - Spinnaker C GenICam Enumerations, [267](#)
- CxpConnectionSelector
 - quickSpin, [300](#)
- CxpConnectionTestErrorCount
 - quickSpin, [300](#)
- CxpConnectionTestMode
 - quickSpin, [300](#)
- CxpConnectionTestMode_Mode1
 - Camera Enumerations, [59](#)
- CxpConnectionTestMode_Off
 - Camera Enumerations, [59](#)
- CxpConnectionTestPacketCount
 - quickSpin, [300](#)
- CxpLinkConfiguration
 - quickSpin, [300](#)
- CxpLinkConfiguration_Auto
 - Camera Enumerations, [59](#)
- CxpLinkConfiguration_CXP1_X1
 - Camera Enumerations, [59](#)
- CxpLinkConfiguration_CXP1_X2
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP1_X3
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP1_X4
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP1_X5
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP1_X6
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP2_X1
 - Camera Enumerations, [59](#)
- CxpLinkConfiguration_CXP2_X2
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP2_X3
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP2_X4
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP2_X5
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP2_X6
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP3_X1
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP3_X2
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP3_X3
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP3_X4
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP3_X5
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP3_X6
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP5_X1
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP5_X2
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP5_X3
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP5_X4
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP5_X5
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP5_X6
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP6_X1
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP6_X2
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP6_X3
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP6_X4
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP6_X5
 - Camera Enumerations, [60](#)
- CxpLinkConfiguration_CXP6_X6
 - Camera Enumerations, [60](#)
- CxpLinkConfigurationPreferred
 - quickSpin, [300](#)
- CxpLinkConfigurationPreferred_CXP1_X1
 - Camera Enumerations, [60](#)
- CxpLinkConfigurationPreferred_CXP1_X2
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP1_X3
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP1_X4
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP1_X5
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP1_X6
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP2_X1
 - Camera Enumerations, [60](#)
- CxpLinkConfigurationPreferred_CXP2_X2
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP2_X3
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP2_X4
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP2_X5
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP2_X6
 - Camera Enumerations, [61](#)

- Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP3_X1
 - Camera Enumerations, [60](#)
- CxpLinkConfigurationPreferred_CXP3_X2
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP3_X3
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP3_X4
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP3_X5
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP3_X6
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP5_X1
 - Camera Enumerations, [60](#)
- CxpLinkConfigurationPreferred_CXP5_X2
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP5_X3
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP5_X4
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP5_X5
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP5_X6
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP6_X1
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP6_X2
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP6_X3
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP6_X4
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP6_X5
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationPreferred_CXP6_X6
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationStatus
 - quickSpin, [300](#)
- CxpLinkConfigurationStatus_CXP1_X1
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationStatus_CXP1_X2
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationStatus_CXP1_X3
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP1_X4
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP1_X5
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP1_X6
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP2_X1
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationStatus_CXP2_X2
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationStatus_CXP2_X3
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP2_X4
 - Camera Enumerations, [62](#)
- Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP2_X5
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP2_X6
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP3_X1
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationStatus_CXP3_X2
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP3_X3
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP3_X4
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP3_X5
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP3_X6
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP5_X1
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationStatus_CXP5_X2
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP5_X3
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP5_X4
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP5_X5
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP5_X6
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP6_X1
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationStatus_CXP6_X2
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP6_X3
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP6_X4
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP6_X5
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_CXP6_X6
 - Camera Enumerations, [62](#)
- CxpLinkConfigurationStatus_None
 - Camera Enumerations, [61](#)
- CxpLinkConfigurationStatus_Pending
 - Camera Enumerations, [61](#)
- CxpPoCxpAuto
 - quickSpin, [300](#)
- CxpPoCxpStatus
 - quickSpin, [300](#)
- CxpPoCxpStatus_Auto
 - Camera Enumerations, [62](#)
- CxpPoCxpStatus_Off
 - Camera Enumerations, [62](#)
- CxpPoCxpStatus_Tripped
 - Camera Enumerations, [62](#)
- CxpPoCxpTripReset
 - quickSpin, [300](#)
- CxpPoCxpTurnOff

- quickSpin, [300](#)
- DEFAULT
 - Spinnaker C Enumerations, [209](#)
- DEFLATE
 - Spinnaker C Structures, [215](#)
- DIRECTIONAL_FILTER
 - Spinnaker C Enumerations, [209](#)
- DecimationHorizontal
 - quickSpin, [300](#)
- DecimationHorizontalMode
 - quickSpin, [300](#)
- DecimationHorizontalMode_Discard
 - Camera Enumerations, [63](#)
- DecimationSelector
 - quickSpin, [301](#)
- DecimationSelector_All
 - Camera Enumerations, [63](#)
- DecimationSelector_Sensor
 - Camera Enumerations, [63](#)
- DecimationVertical
 - quickSpin, [301](#)
- DecimationVerticalMode
 - quickSpin, [301](#)
- DecimationVerticalMode_Discard
 - Camera Enumerations, [63](#)
- Decreasing
 - Spinnaker C GenICam Enumerations, [268](#)
- DefectCorrectStaticEnable
 - quickSpin, [301](#)
- DefectCorrectionMode
 - quickSpin, [301](#)
- DefectCorrectionMode_Average
 - Camera Enumerations, [63](#)
- DefectCorrectionMode_Highlight
 - Camera Enumerations, [63](#)
- DefectCorrectionMode_Zero
 - Camera Enumerations, [63](#)
- DefectTableApply
 - quickSpin, [301](#)
- DefectTableCoordinateX
 - quickSpin, [301](#)
- DefectTableCoordinateY
 - quickSpin, [301](#)
- DefectTableFactoryRestore
 - quickSpin, [301](#)
- DefectTableIndex
 - quickSpin, [301](#)
- DefectTablePixelCount
 - quickSpin, [301](#)
- DefectTableSave
 - quickSpin, [301](#)
- Deinterlacing
 - quickSpin, [301](#)
- Deinterlacing_LineDuplication
 - Camera Enumerations, [63](#)
- Deinterlacing_Off
 - Camera Enumerations, [63](#)
- Deinterlacing_Weave
 - Camera Enumerations, [63](#)
- Device Event Data Access, [195](#)
 - spinDeviceEventGetId, [195](#)
 - spinDeviceEventGetName, [195](#)
 - spinDeviceEventGetPayloadData, [196](#)
 - spinDeviceEventGetPayloadDataSize, [196](#)
- DeviceAccessStatus
 - quickSpinTLDevice, [324](#)
 - quickSpinTLInterface, [326](#)
- DeviceAccessStatus_NoAccess
 - Transport Layer Enumerations, [273](#)
- DeviceAccessStatus_ReadOnly
 - Transport Layer Enumerations, [273](#)
- DeviceAccessStatus_ReadWrite
 - Transport Layer Enumerations, [273](#)
- DeviceAccessStatus_Unknown
 - Transport Layer Enumerations, [273](#)
- DeviceAddress
 - actionCommandResult, [281](#)
- DeviceCharacterSet
 - quickSpin, [301](#)
- DeviceCharacterSet_ASCII
 - Camera Enumerations, [64](#)
- DeviceCharacterSet_UTF8
 - Camera Enumerations, [64](#)
- DeviceClockFrequency
 - quickSpin, [301](#)
- DeviceClockSelector
 - quickSpin, [301](#)
- DeviceClockSelector_CameraLink
 - Camera Enumerations, [64](#)
- DeviceClockSelector_Sensor
 - Camera Enumerations, [64](#)
- DeviceClockSelector_SensorDigitization
 - Camera Enumerations, [64](#)
- DeviceConnectionSelector
 - quickSpin, [301](#)
- DeviceConnectionSpeed
 - quickSpin, [301](#)
- DeviceConnectionStatus
 - quickSpin, [301](#)
- DeviceConnectionStatus_Active
 - Camera Enumerations, [64](#)
- DeviceConnectionStatus_Inactive
 - Camera Enumerations, [64](#)
- DeviceCount
 - quickSpinTLInterface, [326](#)
- DeviceCurrentSpeed
 - quickSpinTLDevice, [324](#)
- DeviceCurrentSpeed_FullSpeed
 - Transport Layer Enumerations, [274](#)
- DeviceCurrentSpeed_HighSpeed
 - Transport Layer Enumerations, [274](#)
- DeviceCurrentSpeed_LowSpeed
 - Transport Layer Enumerations, [274](#)
- DeviceCurrentSpeed_SuperSpeed
 - Transport Layer Enumerations, [274](#)
- DeviceCurrentSpeed_UnknownSpeed

- Transport Layer Enumerations, 274
- DeviceDisplayName
 - quickSpinTLDevice, 324
- DeviceDriverVersion
 - quickSpinTLDevice, 324
- DeviceEndiannessMechanism
 - quickSpinTLDevice, 324
- DeviceEndiannessMechanism_Legacy
 - Transport Layer Enumerations, 274
- DeviceEndiannessMechanism_Standard
 - Transport Layer Enumerations, 274
- DeviceEventChannelCount
 - quickSpin, 301
- DeviceFamilyName
 - quickSpin, 301
- DeviceFeaturePersistenceEnd
 - quickSpin, 301
- DeviceFeaturePersistenceStart
 - quickSpin, 301
- DeviceFirmwareVersion
 - quickSpin, 302
- DeviceGenCPVersionMajor
 - quickSpin, 302
- DeviceGenCPVersionMinor
 - quickSpin, 302
- DeviceID
 - quickSpin, 302
 - quickSpinTLDevice, 324
 - quickSpinTLInterface, 326
- DeviceIndicatorMode
 - quickSpin, 302
- DeviceIndicatorMode_Active
 - Camera Enumerations, 64
- DeviceIndicatorMode_ErrorStatus
 - Camera Enumerations, 64
- DeviceIndicatorMode_Inactive
 - Camera Enumerations, 64
- DeviceInstanceId
 - quickSpinTLDevice, 324
- DeviceIsUpdater
 - quickSpinTLDevice, 324
- DeviceLinkBandwidthReserve
 - quickSpin, 302
- DeviceLinkCommandTimeout
 - quickSpin, 302
- DeviceLinkConnectionCount
 - quickSpin, 302
- DeviceLinkCurrentThroughput
 - quickSpin, 302
- DeviceLinkHeartbeatMode
 - quickSpin, 302
- DeviceLinkHeartbeatMode_Off
 - Camera Enumerations, 65
- DeviceLinkHeartbeatMode_On
 - Camera Enumerations, 65
- DeviceLinkHeartbeatTimeout
 - quickSpin, 302
- DeviceLinkSelector
 - quickSpin, 302
- DeviceLinkSpeed
 - quickSpin, 302
- DeviceLinkThroughputLimit
 - quickSpinTLDevice, 324
- DeviceLinkThroughputLimitMode
 - quickSpin, 302
- DeviceLinkThroughputLimitMode_Off
 - Camera Enumerations, 65
- DeviceLinkThroughputLimitMode_On
 - Camera Enumerations, 65
- DeviceManifestEntrySelector
 - quickSpin, 302
- DeviceManifestPrimaryURL
 - quickSpin, 302
- DeviceManifestSchemaMajorVersion
 - quickSpin, 302
- DeviceManifestSchemaMinorVersion
 - quickSpin, 302
- DeviceManifestSecondaryURL
 - quickSpin, 302
- DeviceManifestXMLMajorVersion
 - quickSpin, 302
- DeviceManifestXMLMinorVersion
 - quickSpin, 302
- DeviceManifestXMLSubMinorVersion
 - quickSpin, 302
- DeviceManufacturerInfo
 - quickSpin, 303
- DeviceMaxThroughput
 - quickSpin, 303
- DeviceModelName
 - quickSpin, 303
 - quickSpinTLDevice, 324
 - quickSpinTLInterface, 327
- DeviceMulticastMonitorMode
 - quickSpinTLDevice, 324
- DevicePowerSupplySelector
 - quickSpin, 303
- DevicePowerSupplySelector_External
 - Camera Enumerations, 65
- DeviceRegistersCheck
 - quickSpin, 303
- DeviceRegistersEndianness
 - quickSpin, 303
- DeviceRegistersEndianness_Big
 - Camera Enumerations, 65
- DeviceRegistersEndianness_Little
 - Camera Enumerations, 65
- DeviceRegistersStreamingEnd
 - quickSpin, 303
- DeviceRegistersStreamingStart
 - quickSpin, 303
- DeviceRegistersValid
 - quickSpin, 303
- DeviceReset
 - quickSpin, 303

- DeviceSFNCVersionMajor
 - quickSpin, [303](#)
- DeviceSFNCVersionMinor
 - quickSpin, [303](#)
- DeviceSFNCVersionSubMinor
 - quickSpin, [303](#)
- DeviceScanType
 - quickSpin, [303](#)
- DeviceScanType_Areascan
 - Camera Enumerations, [65](#)
- DeviceSelector
 - quickSpinTLInterface, [327](#)
- DeviceSerialNumber
 - quickSpin, [303](#)
 - quickSpinTLDevice, [324](#)
- DeviceSerialPortBaudRate
 - quickSpin, [303](#)
- DeviceSerialPortBaudRate_Baud115200
 - Camera Enumerations, [66](#)
- DeviceSerialPortBaudRate_Baud19200
 - Camera Enumerations, [66](#)
- DeviceSerialPortBaudRate_Baud230400
 - Camera Enumerations, [66](#)
- DeviceSerialPortBaudRate_Baud38400
 - Camera Enumerations, [66](#)
- DeviceSerialPortBaudRate_Baud460800
 - Camera Enumerations, [66](#)
- DeviceSerialPortBaudRate_Baud57600
 - Camera Enumerations, [66](#)
- DeviceSerialPortBaudRate_Baud921600
 - Camera Enumerations, [66](#)
- DeviceSerialPortBaudRate_Baud9600
 - Camera Enumerations, [66](#)
- DeviceSerialPortSelector
 - quickSpin, [303](#)
- DeviceSerialPortSelector_CameraLink
 - Camera Enumerations, [66](#)
- DeviceStreamChannelCount
 - quickSpin, [303](#)
- DeviceStreamChannelEndianness
 - quickSpin, [303](#)
- DeviceStreamChannelEndianness_Big
 - Camera Enumerations, [66](#)
- DeviceStreamChannelEndianness_Little
 - Camera Enumerations, [66](#)
- DeviceStreamChannelLink
 - quickSpin, [303](#)
- DeviceStreamChannelPacketSize
 - quickSpin, [303](#)
- DeviceStreamChannelSelector
 - quickSpin, [303](#)
- DeviceStreamChannelType
 - quickSpin, [303](#)
- DeviceStreamChannelType_Receiver
 - Camera Enumerations, [66](#)
- DeviceStreamChannelType_Transmitter
 - Camera Enumerations, [66](#)
- DeviceTLType
 - quickSpin, [304](#)
- DeviceTLType_CameraLink
 - Camera Enumerations, [68](#)
- DeviceTLType_CameraLinkHS
 - Camera Enumerations, [68](#)
- DeviceTLType_CoaXPRESS
 - Camera Enumerations, [68](#)
- DeviceTLType_Custom
 - Camera Enumerations, [68](#)
- DeviceTLType_GigEVision
 - Camera Enumerations, [68](#)
- DeviceTLType_USB3Vision
 - Camera Enumerations, [68](#)
- DeviceTLVersionMajor
 - quickSpin, [304](#)
- DeviceTLVersionMinor
 - quickSpin, [304](#)
- DeviceTLVersionSubMinor
 - quickSpin, [304](#)
- DeviceTapGeometry
 - quickSpin, [304](#)
- DeviceTapGeometry_Geometry_10X_1Y
 - Camera Enumerations, [68](#)
- DeviceTapGeometry_Geometry_10X
 - Camera Enumerations, [68](#)
- DeviceTapGeometry_Geometry_1X10
 - Camera Enumerations, [68](#)
- DeviceTapGeometry_Geometry_1X10_1Y
 - Camera Enumerations, [68](#)
- DeviceTapGeometry_Geometry_1X2
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_1X2_1Y2
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_1X2_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_1X2_2YE
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_1X3
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_1X3_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_1X4
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_1X4_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_1X8
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_1X8_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_1X_1Y2
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_1X_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_1X_2YE
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_1X
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2X2

- Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2X2_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2X2E_1YGeometry↔_2X2M_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2X2E_2YE
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2X2E
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2X2M
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2X_1Y2Geometry_2↔_XE_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2X_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2X_2YE
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2XE_1Y2
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2XE_2YE
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2XM_1Y2
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2XM_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2XM_2YE
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2XE
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2XM
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_2X
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_3X_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_3X
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_4X2
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_4X2_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_4X2E_1Y
 - Camera Enumerations, [68](#)
- DeviceTapGeometry_Geometry_4X2E
 - Camera Enumerations, [68](#)
- DeviceTapGeometry_Geometry_4X_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_4X
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_8X_1Y
 - Camera Enumerations, [67](#)
- DeviceTapGeometry_Geometry_8X
 - Camera Enumerations, [67](#)
- DeviceTemperature
 - quickSpin, [304](#)
- DeviceTemperatureSelector
 - quickSpin, [304](#)
- DeviceTemperatureSelector_Sensor
 - Camera Enumerations, [68](#)
- DeviceType
 - quickSpin, [304](#)
 - quickSpinTLDevice, [324](#)
- DeviceType_CLHS
 - Transport Layer Enumerations, [274](#)
- DeviceType_CXP
 - Transport Layer Enumerations, [274](#)
- DeviceType_CL
 - Transport Layer Enumerations, [274](#)
- DeviceType_Custom
 - Transport Layer Enumerations, [274](#)
- DeviceType_ETHERNET
 - Transport Layer Enumerations, [274](#)
- DeviceType_GEV
 - Transport Layer Enumerations, [274](#)
- DeviceType_IIDC
 - Transport Layer Enumerations, [274](#)
- DeviceType_Mixed
 - Transport Layer Enumerations, [274](#)
- DeviceType_PCI
 - Transport Layer Enumerations, [274](#)
- DeviceType_Peripheral
 - Camera Enumerations, [68](#)
- DeviceType_Receiver
 - Camera Enumerations, [68](#)
- DeviceType_Transceiver
 - Camera Enumerations, [68](#)
- DeviceType_Transmitter
 - Camera Enumerations, [68](#)
- DeviceType_U3V
 - Transport Layer Enumerations, [274](#)
- DeviceType_UVC
 - Transport Layer Enumerations, [274](#)
- DeviceU3VProtocol
 - quickSpinTLDevice, [324](#)
- DeviceUnlock
 - quickSpinTLInterface, [327](#)
- DeviceUpdateList
 - quickSpinTLInterface, [327](#)
- DeviceUptime
 - quickSpin, [304](#)
- DeviceUserID
 - quickSpin, [304](#)
 - quickSpinTLDevice, [324](#)
- DeviceVendorName
 - quickSpin, [304](#)
 - quickSpinTLDevice, [324](#)
 - quickSpinTLInterface, [327](#)
- DeviceVersion
 - quickSpin, [304](#)
 - quickSpinTLDevice, [325](#)
- doc/Doxygen/spindocs/C/Licensing.dox, [343](#)
- doc/Doxygen/spindocs/C/MainPage.dox, [343](#)
- EDGE_SENSING
 - Spinnaker C Enumerations, [209](#)

- EncoderDivider
 - quickSpin, [304](#)
- EncoderMode
 - quickSpin, [304](#)
- EncoderMode_FourPhase
 - Camera Enumerations, [69](#)
- EncoderMode_HighResolution
 - Camera Enumerations, [69](#)
- EncoderOutputMode
 - quickSpin, [304](#)
- EncoderOutputMode_DirectionDown
 - Camera Enumerations, [69](#)
- EncoderOutputMode_DirectionUp
 - Camera Enumerations, [69](#)
- EncoderOutputMode_Motion
 - Camera Enumerations, [69](#)
- EncoderOutputMode_Off
 - Camera Enumerations, [69](#)
- EncoderOutputMode_PositionDown
 - Camera Enumerations, [69](#)
- EncoderOutputMode_PositionUp
 - Camera Enumerations, [69](#)
- EncoderReset
 - quickSpin, [304](#)
- EncoderResetActivation
 - quickSpin, [304](#)
- EncoderResetActivation_AnyEdge
 - Camera Enumerations, [69](#)
- EncoderResetActivation_FallingEdge
 - Camera Enumerations, [69](#)
- EncoderResetActivation_LevelHigh
 - Camera Enumerations, [69](#)
- EncoderResetActivation_LevelLow
 - Camera Enumerations, [69](#)
- EncoderResetActivation_RisingEdge
 - Camera Enumerations, [69](#)
- EncoderResetSource
 - quickSpin, [304](#)
- EncoderResetSource_AcquisitionEnd
 - Camera Enumerations, [70](#)
- EncoderResetSource_AcquisitionStart
 - Camera Enumerations, [70](#)
- EncoderResetSource_AcquisitionTrigger
 - Camera Enumerations, [70](#)
- EncoderResetSource_Action0
 - Camera Enumerations, [70](#)
- EncoderResetSource_Action1
 - Camera Enumerations, [70](#)
- EncoderResetSource_Action2
 - Camera Enumerations, [70](#)
- EncoderResetSource_Counter0End
 - Camera Enumerations, [70](#)
- EncoderResetSource_Counter0Start
 - Camera Enumerations, [70](#)
- EncoderResetSource_Counter1End
 - Camera Enumerations, [70](#)
- EncoderResetSource_Counter1Start
 - Camera Enumerations, [70](#)
- EncoderResetSource_Counter2End
 - Camera Enumerations, [70](#)
- EncoderResetSource_Counter2Start
 - Camera Enumerations, [70](#)
- EncoderResetSource_ExposureEnd
 - Camera Enumerations, [70](#)
- EncoderResetSource_ExposureStart
 - Camera Enumerations, [70](#)
- EncoderResetSource_FrameEnd
 - Camera Enumerations, [70](#)
- EncoderResetSource_FrameStart
 - Camera Enumerations, [70](#)
- EncoderResetSource_FrameTrigger
 - Camera Enumerations, [70](#)
- EncoderResetSource_Line0
 - Camera Enumerations, [70](#)
- EncoderResetSource_Line1
 - Camera Enumerations, [70](#)
- EncoderResetSource_Line2
 - Camera Enumerations, [70](#)
- EncoderResetSource_LinkTrigger0
 - Camera Enumerations, [70](#)
- EncoderResetSource_LinkTrigger1
 - Camera Enumerations, [70](#)
- EncoderResetSource_LinkTrigger2
 - Camera Enumerations, [70](#)
- EncoderResetSource_Off
 - Camera Enumerations, [70](#)
- EncoderResetSource_SoftwareSignal0
 - Camera Enumerations, [70](#)
- EncoderResetSource_SoftwareSignal1
 - Camera Enumerations, [70](#)
- EncoderResetSource_SoftwareSignal2
 - Camera Enumerations, [70](#)
- EncoderResetSource_Timer0End
 - Camera Enumerations, [70](#)
- EncoderResetSource_Timer0Start
 - Camera Enumerations, [70](#)
- EncoderResetSource_Timer1End
 - Camera Enumerations, [70](#)
- EncoderResetSource_Timer1Start
 - Camera Enumerations, [70](#)
- EncoderResetSource_Timer2End
 - Camera Enumerations, [70](#)
- EncoderResetSource_Timer2Start
 - Camera Enumerations, [70](#)
- EncoderResetSource_UserOutput0
 - Camera Enumerations, [70](#)
- EncoderResetSource_UserOutput1
 - Camera Enumerations, [70](#)
- EncoderResetSource_UserOutput2
 - Camera Enumerations, [70](#)
- EncoderSelector
 - quickSpin, [304](#)
- EncoderSelector_Encoder0
 - Camera Enumerations, [71](#)
- EncoderSelector_Encoder1
 - Camera Enumerations, [71](#)

- EncoderSelector_Encoder2
 - Camera Enumerations, [71](#)
- EncoderSourceA_Line0
 - Camera Enumerations, [71](#)
- EncoderSourceA_Line1
 - Camera Enumerations, [71](#)
- EncoderSourceA_Line2
 - Camera Enumerations, [71](#)
- EncoderSourceA_Off
 - Camera Enumerations, [71](#)
- EncoderSourceB_Line0
 - Camera Enumerations, [71](#)
- EncoderSourceB_Line1
 - Camera Enumerations, [71](#)
- EncoderSourceB_Line2
 - Camera Enumerations, [71](#)
- EncoderSourceB_Off
 - Camera Enumerations, [71](#)
- EncoderSourceA
 - quickSpin, [304](#)
- EncoderSourceB
 - quickSpin, [304](#)
- EncoderStatus
 - quickSpin, [304](#)
- EncoderStatus_EncoderDown
 - Camera Enumerations, [71](#)
- EncoderStatus_EncoderIdle
 - Camera Enumerations, [71](#)
- EncoderStatus_EncoderStatic
 - Camera Enumerations, [71](#)
- EncoderStatus_EncoderUp
 - Camera Enumerations, [71](#)
- EncoderTimeout
 - quickSpin, [304](#)
- EncoderValue
 - quickSpin, [305](#)
- EncoderValueAtReset
 - quickSpin, [305](#)
- EnumEntryNode
 - Spinnaker C GenICam Enumerations, [267](#)
- EnumerationCount
 - quickSpin, [305](#)
- EnumerationNode
 - Spinnaker C GenICam Enumerations, [267](#)
- Error Handling, [116](#)
 - spinErrorGetLast, [116](#)
 - spinErrorGetLastBuildDate, [117](#)
 - spinErrorGetLastBuildTime, [117](#)
 - spinErrorGetLastFileName, [117](#)
 - spinErrorGetLastFullMessage, [118](#)
 - spinErrorGetLastFunctionName, [118](#)
 - spinErrorGetLastLineNumber, [119](#)
 - spinErrorGetLastMessage, [119](#)
- Event Access, [178](#)
 - spinArrivalEventCreate, [178](#)
 - spinArrivalEventDestroy, [179](#)
 - spinDeviceEventCreate, [179](#)
 - spinDeviceEventDestroy, [180](#)
 - spinImageEventCreate, [180](#)
 - spinImageEventDestroy, [180](#)
 - spinInterfaceEventCreate, [181](#)
 - spinInterfaceEventDestroy, [181](#)
 - spinLogEventCreate, [181](#)
 - spinLogEventDestroy, [182](#)
 - spinRemovalEventCreate, [182](#)
 - spinRemovalEventDestroy, [183](#)
- EventAcquisitionEnd
 - quickSpin, [305](#)
- EventAcquisitionEndFrameID
 - quickSpin, [305](#)
- EventAcquisitionEndTimestamp
 - quickSpin, [305](#)
- EventAcquisitionError
 - quickSpin, [305](#)
- EventAcquisitionErrorFrameID
 - quickSpin, [305](#)
- EventAcquisitionErrorTimestamp
 - quickSpin, [305](#)
- EventAcquisitionStart
 - quickSpin, [305](#)
- EventAcquisitionStartFrameID
 - quickSpin, [305](#)
- EventAcquisitionStartTimestamp
 - quickSpin, [305](#)
- EventAcquisitionTransferEnd
 - quickSpin, [305](#)
- EventAcquisitionTransferEndFrameID
 - quickSpin, [305](#)
- EventAcquisitionTransferEndTimestamp
 - quickSpin, [305](#)
- EventAcquisitionTransferStart
 - quickSpin, [305](#)
- EventAcquisitionTransferStartFrameID
 - quickSpin, [305](#)
- EventAcquisitionTransferStartTimestamp
 - quickSpin, [305](#)
- EventAcquisitionTrigger
 - quickSpin, [305](#)
- EventAcquisitionTriggerFrameID
 - quickSpin, [305](#)
- EventAcquisitionTriggerTimestamp
 - quickSpin, [305](#)
- EventActionLate
 - quickSpin, [305](#)
- EventActionLateFrameID
 - quickSpin, [305](#)
- EventActionLateTimestamp
 - quickSpin, [306](#)
- EventCounter0End
 - quickSpin, [306](#)
- EventCounter0EndFrameID
 - quickSpin, [306](#)
- EventCounter0EndTimestamp
 - quickSpin, [306](#)
- EventCounter0Start
 - quickSpin, [306](#)

EventCounter0StartFrameID
 [quickSpin, 306](#)

EventCounter0StartTimestamp
 [quickSpin, 306](#)

EventCounter1End
 [quickSpin, 306](#)

EventCounter1EndFrameID
 [quickSpin, 306](#)

EventCounter1EndTimestamp
 [quickSpin, 306](#)

EventCounter1Start
 [quickSpin, 306](#)

EventCounter1StartFrameID
 [quickSpin, 306](#)

EventCounter1StartTimestamp
 [quickSpin, 306](#)

EventEncoder0Restarted
 [quickSpin, 306](#)

EventEncoder0RestartedFrameID
 [quickSpin, 306](#)

EventEncoder0RestartedTimestamp
 [quickSpin, 306](#)

EventEncoder0Stopped
 [quickSpin, 306](#)

EventEncoder0StoppedFrameID
 [quickSpin, 306](#)

EventEncoder0StoppedTimestamp
 [quickSpin, 306](#)

EventEncoder1Restarted
 [quickSpin, 306](#)

EventEncoder1RestartedFrameID
 [quickSpin, 306](#)

EventEncoder1RestartedTimestamp
 [quickSpin, 306](#)

EventEncoder1Stopped
 [quickSpin, 306](#)

EventEncoder1StoppedFrameID
 [quickSpin, 307](#)

EventEncoder1StoppedTimestamp
 [quickSpin, 307](#)

EventError
 [quickSpin, 307](#)

EventErrorCode
 [quickSpin, 307](#)

EventErrorFrameID
 [quickSpin, 307](#)

EventErrorTimestamp
 [quickSpin, 307](#)

EventExposureEnd
 [quickSpin, 307](#)

EventExposureEndFrameID
 [quickSpin, 307](#)

EventExposureEndTimestamp
 [quickSpin, 307](#)

EventExposureStart
 [quickSpin, 307](#)

EventExposureStartFrameID
 [quickSpin, 307](#)

EventExposureStartTimestamp
 [quickSpin, 307](#)

EventFrameBurstEnd
 [quickSpin, 307](#)

EventFrameBurstEndFrameID
 [quickSpin, 307](#)

EventFrameBurstEndTimestamp
 [quickSpin, 307](#)

EventFrameBurstStart
 [quickSpin, 307](#)

EventFrameBurstStartFrameID
 [quickSpin, 307](#)

EventFrameBurstStartTimestamp
 [quickSpin, 307](#)

EventFrameEnd
 [quickSpin, 307](#)

EventFrameEndFrameID
 [quickSpin, 307](#)

EventFrameEndTimestamp
 [quickSpin, 307](#)

EventFrameStart
 [quickSpin, 307](#)

EventFrameStartFrameID
 [quickSpin, 307](#)

EventFrameStartTimestamp
 [quickSpin, 308](#)

EventFrameTransferEnd
 [quickSpin, 308](#)

EventFrameTransferEndFrameID
 [quickSpin, 308](#)

EventFrameTransferEndTimestamp
 [quickSpin, 308](#)

EventFrameTransferStart
 [quickSpin, 308](#)

EventFrameTransferStartFrameID
 [quickSpin, 308](#)

EventFrameTransferStartTimestamp
 [quickSpin, 308](#)

EventFrameTrigger
 [quickSpin, 308](#)

EventFrameTriggerFrameID
 [quickSpin, 308](#)

EventFrameTriggerTimestamp
 [quickSpin, 308](#)

EventLine0AnyEdge
 [quickSpin, 308](#)

EventLine0AnyEdgeFrameID
 [quickSpin, 308](#)

EventLine0AnyEdgeTimestamp
 [quickSpin, 308](#)

EventLine0FallingEdge
 [quickSpin, 308](#)

EventLine0FallingEdgeFrameID
 [quickSpin, 308](#)

EventLine0FallingEdgeTimestamp
 [quickSpin, 308](#)

EventLine0RisingEdge
 [quickSpin, 308](#)

- EventLine0RisingEdgeFrameID
 - quickSpin, [308](#)
- EventLine0RisingEdgeTimestamp
 - quickSpin, [308](#)
- EventLine1AnyEdge
 - quickSpin, [308](#)
- EventLine1AnyEdgeFrameID
 - quickSpin, [308](#)
- EventLine1AnyEdgeTimestamp
 - quickSpin, [308](#)
- EventLine1FallingEdge
 - quickSpin, [308](#)
- EventLine1FallingEdgeFrameID
 - quickSpin, [309](#)
- EventLine1FallingEdgeTimestamp
 - quickSpin, [309](#)
- EventLine1RisingEdge
 - quickSpin, [309](#)
- EventLine1RisingEdgeFrameID
 - quickSpin, [309](#)
- EventLine1RisingEdgeTimestamp
 - quickSpin, [309](#)
- EventLinkSpeedChange
 - quickSpin, [309](#)
- EventLinkSpeedChangeFrameID
 - quickSpin, [309](#)
- EventLinkSpeedChangeTimestamp
 - quickSpin, [309](#)
- EventLinkTrigger0
 - quickSpin, [309](#)
- EventLinkTrigger0FrameID
 - quickSpin, [309](#)
- EventLinkTrigger0Timestamp
 - quickSpin, [309](#)
- EventLinkTrigger1
 - quickSpin, [309](#)
- EventLinkTrigger1FrameID
 - quickSpin, [309](#)
- EventLinkTrigger1Timestamp
 - quickSpin, [309](#)
- EventNotification
 - quickSpin, [309](#)
- EventNotification_Off
 - Camera Enumerations, [72](#)
- EventNotification_On
 - Camera Enumerations, [72](#)
- EventSelector
 - quickSpin, [309](#)
- EventSelector_Error
 - Camera Enumerations, [72](#)
- EventSelector_ExposureEnd
 - Camera Enumerations, [72](#)
- EventSelector_SerialPortReceive
 - Camera Enumerations, [72](#)
- EventSequencerSetChange
 - quickSpin, [309](#)
- EventSequencerSetChangeFrameID
 - quickSpin, [309](#)
- EventSequencerSetChangeTimestamp
 - quickSpin, [309](#)
- EventSerialData
 - quickSpin, [309](#)
- EventSerialDataLength
 - quickSpin, [309](#)
- EventSerialPortReceive
 - quickSpin, [309](#)
- EventSerialPortReceiveTimestamp
 - quickSpin, [309](#)
- EventSerialReceiveOverflow
 - quickSpin, [310](#)
- EventStream0TransferBlockEnd
 - quickSpin, [310](#)
- EventStream0TransferBlockEndFrameID
 - quickSpin, [310](#)
- EventStream0TransferBlockEndTimestamp
 - quickSpin, [310](#)
- EventStream0TransferBlockStart
 - quickSpin, [310](#)
- EventStream0TransferBlockStartFrameID
 - quickSpin, [310](#)
- EventStream0TransferBlockStartTimestamp
 - quickSpin, [310](#)
- EventStream0TransferBlockTrigger
 - quickSpin, [310](#)
- EventStream0TransferBlockTriggerFrameID
 - quickSpin, [310](#)
- EventStream0TransferBlockTriggerTimestamp
 - quickSpin, [310](#)
- EventStream0TransferBurstEnd
 - quickSpin, [310](#)
- EventStream0TransferBurstEndFrameID
 - quickSpin, [310](#)
- EventStream0TransferBurstEndTimestamp
 - quickSpin, [310](#)
- EventStream0TransferBurstStart
 - quickSpin, [310](#)
- EventStream0TransferBurstStartFrameID
 - quickSpin, [310](#)
- EventStream0TransferBurstStartTimestamp
 - quickSpin, [310](#)
- EventStream0TransferEnd
 - quickSpin, [310](#)
- EventStream0TransferEndFrameID
 - quickSpin, [310](#)
- EventStream0TransferEndTimestamp
 - quickSpin, [310](#)
- EventStream0TransferOverflow
 - quickSpin, [310](#)
- EventStream0TransferOverflowFrameID
 - quickSpin, [310](#)
- EventStream0TransferOverflowTimestamp
 - quickSpin, [310](#)
- EventStream0TransferPause
 - quickSpin, [310](#)
- EventStream0TransferPauseFrameID
 - quickSpin, [311](#)

- EventStream0TransferPauseTimestamp
 - quickSpin, [311](#)
- EventStream0TransferResume
 - quickSpin, [311](#)
- EventStream0TransferResumeFrameID
 - quickSpin, [311](#)
- EventStream0TransferResumeTimestamp
 - quickSpin, [311](#)
- EventStream0TransferStart
 - quickSpin, [311](#)
- EventStream0TransferStartFrameID
 - quickSpin, [311](#)
- EventStream0TransferStartTimestamp
 - quickSpin, [311](#)
- EventTest
 - quickSpin, [311](#)
- EventTestTimestamp
 - quickSpin, [311](#)
- EventTimer0End
 - quickSpin, [311](#)
- EventTimer0EndFrameID
 - quickSpin, [311](#)
- EventTimer0EndTimestamp
 - quickSpin, [311](#)
- EventTimer0Start
 - quickSpin, [311](#)
- EventTimer0StartFrameID
 - quickSpin, [311](#)
- EventTimer0StartTimestamp
 - quickSpin, [311](#)
- EventTimer1End
 - quickSpin, [311](#)
- EventTimer1EndFrameID
 - quickSpin, [311](#)
- EventTimer1EndTimestamp
 - quickSpin, [311](#)
- EventTimer1Start
 - quickSpin, [311](#)
- EventTimer1StartFrameID
 - quickSpin, [311](#)
- EventTimer1StartTimestamp
 - quickSpin, [311](#)
- Expert
 - Spinnaker C GenICam Enumerations, [268](#)
- ExposureActiveMode
 - quickSpin, [311](#)
- ExposureActiveMode_AllPixels
 - Camera Enumerations, [72](#)
- ExposureActiveMode_AnyPixels
 - Camera Enumerations, [72](#)
- ExposureActiveMode_Line1
 - Camera Enumerations, [72](#)
- ExposureAuto
 - quickSpin, [312](#)
- ExposureAuto_Continuous
 - Camera Enumerations, [72](#)
- ExposureAuto_Off
 - Camera Enumerations, [72](#)
- ExposureAuto_Once
 - Camera Enumerations, [72](#)
- ExposureMode
 - quickSpin, [312](#)
- ExposureMode_Timed
 - Camera Enumerations, [73](#)
- ExposureMode_TriggerWidth
 - Camera Enumerations, [73](#)
- ExposureTime
 - quickSpin, [312](#)
- ExposureTimeMode
 - quickSpin, [312](#)
- ExposureTimeMode_Common
 - Camera Enumerations, [73](#)
- ExposureTimeMode_Individual
 - Camera Enumerations, [73](#)
- ExposureTimeSelector
 - quickSpin, [312](#)
- ExposureTimeSelector_Blue
 - Camera Enumerations, [73](#)
- ExposureTimeSelector_Common
 - Camera Enumerations, [73](#)
- ExposureTimeSelector_Cyan
 - Camera Enumerations, [73](#)
- ExposureTimeSelector_Green
 - Camera Enumerations, [73](#)
- ExposureTimeSelector_Infrared
 - Camera Enumerations, [73](#)
- ExposureTimeSelector_Magenta
 - Camera Enumerations, [73](#)
- ExposureTimeSelector_Red
 - Camera Enumerations, [73](#)
- ExposureTimeSelector_Stage1
 - Camera Enumerations, [73](#)
- ExposureTimeSelector_Stage2
 - Camera Enumerations, [73](#)
- ExposureTimeSelector_Ultraviolet
 - Camera Enumerations, [73](#)
- ExposureTimeSelector_Yellow
 - Camera Enumerations, [73](#)
- FROM_FILE_EXT
 - Spinnaker C Enumerations, [211](#)
- FactoryReset
 - quickSpin, [312](#)
- False
 - Spinnaker C Definitions, [12](#)
- FileAccessBuffer
 - quickSpin, [312](#)
- FileAccessLength
 - quickSpin, [312](#)
- FileAccessOffset
 - quickSpin, [312](#)
- FileOpenMode
 - quickSpin, [312](#)
- FileOpenMode_Read
 - Camera Enumerations, [74](#)
- FileOpenMode_ReadWrite
 - Camera Enumerations, [74](#)

- FileOpenMode_Write
 - Camera Enumerations, [74](#)
- FileOperationExecute
 - quickSpin, [312](#)
- FileOperationResult
 - quickSpin, [312](#)
- FileOperationSelector
 - quickSpin, [312](#)
- FileOperationSelector_Close
 - Camera Enumerations, [74](#)
- FileOperationSelector_Delete
 - Camera Enumerations, [74](#)
- FileOperationSelector_Open
 - Camera Enumerations, [74](#)
- FileOperationSelector_Read
 - Camera Enumerations, [74](#)
- FileOperationSelector_Write
 - Camera Enumerations, [74](#)
- FileOperationStatus
 - quickSpin, [312](#)
- FileOperationStatus_Failure
 - Camera Enumerations, [74](#)
- FileOperationStatus_Overflow
 - Camera Enumerations, [74](#)
- FileOperationStatus_Success
 - Camera Enumerations, [74](#)
- FileSelector
 - quickSpin, [312](#)
- FileSelector_SerialPort0
 - Camera Enumerations, [74](#)
- FileSelector_UserFile1
 - Camera Enumerations, [74](#)
- FileSelector_UserSet0
 - Camera Enumerations, [74](#)
- FileSelector_UserSet1
 - Camera Enumerations, [74](#)
- FileSelector_UserSetDefault
 - Camera Enumerations, [74](#)
- FileSize
 - quickSpin, [312](#)
- fixedIncrement
 - Spinnaker C GenICam Enumerations, [265](#)
- FloatNode
 - Spinnaker C GenICam Enumerations, [267](#)
- fnAutomatic
 - Spinnaker C GenICam Enumerations, [265](#)
- fnFixed
 - Spinnaker C GenICam Enumerations, [265](#)
- fnScientific
 - Spinnaker C GenICam Enumerations, [265](#)
- frameRate
 - spinAVIOption, [330](#)
 - spinH264Option, [334](#)
 - spinMJPEGOption, [338](#)
- GENICAM_ERR_ACCESS
 - Spinnaker C Enumerations, [210](#)
- GENICAM_ERR_BAD_ALLOCATION
 - Spinnaker C Enumerations, [210](#)
- GENICAM_ERR_DYNAMIC_CAST
 - Spinnaker C Enumerations, [210](#)
- GENICAM_ERR_GENERIC
 - Spinnaker C Enumerations, [210](#)
- GENICAM_ERR_INVALID_ARGUMENT
 - Spinnaker C Enumerations, [210](#)
- GENICAM_ERR_LOGICAL
 - Spinnaker C Enumerations, [210](#)
- GENICAM_ERR_OUT_OF_RANGE
 - Spinnaker C Enumerations, [210](#)
- GENICAM_ERR_PROPERTY
 - Spinnaker C Enumerations, [210](#)
- GENICAM_ERR_RUN_TIME
 - Spinnaker C Enumerations, [210](#)
- GENICAM_ERR_TIMEOUT
 - Spinnaker C Enumerations, [210](#)
- GEV
 - Spinnaker C GenICam Enumerations, [268](#)
- GREEN
 - Spinnaker C Enumerations, [213](#)
- GREY
 - Spinnaker C Enumerations, [213](#)
- GUIXMLLocation
 - quickSpinTLDevice, [325](#)
- GUIXMLLocation_Device
 - Transport Layer Enumerations, [275](#)
- GUIXMLLocation_Host
 - Transport Layer Enumerations, [275](#)
- GUIXMLPath
 - quickSpinTLDevice, [325](#)
- Gain
 - quickSpin, [312](#)
- GainAuto
 - quickSpin, [312](#)
- GainAuto_Continuous
 - Camera Enumerations, [75](#)
- GainAuto_Off
 - Camera Enumerations, [75](#)
- GainAuto_Once
 - Camera Enumerations, [75](#)
- GainAutoBalance
 - quickSpin, [312](#)
- GainAutoBalance_Continuous
 - Camera Enumerations, [75](#)
- GainAutoBalance_Off
 - Camera Enumerations, [75](#)
- GainAutoBalance_Once
 - Camera Enumerations, [75](#)
- GainSelector
 - quickSpin, [312](#)
- GainSelector_All
 - Camera Enumerations, [75](#)
- Gamma
 - quickSpin, [312](#)
- GammaEnable
 - quickSpin, [312](#)
- GenICamXMLLocation
 - quickSpinTLDevice, [325](#)

- GenICamXMLLocation_Device
 - Transport Layer Enumerations, [275](#)
- GenICamXMLLocation_Host
 - Transport Layer Enumerations, [275](#)
- GenICamXMLPath
 - quickSpinTLDevice, [325](#)
- GevActionDeviceKey
 - quickSpinTLInterface, [327](#)
- GevActionGroupKey
 - quickSpinTLInterface, [327](#)
- GevActionGroupMask
 - quickSpinTLInterface, [327](#)
- GevActionTime
 - quickSpinTLInterface, [327](#)
- GevActiveLinkCount
 - quickSpin, [312](#)
- GevCCP_ControlAccess
 - Camera Enumerations, [75](#)
- GevCCP_EnumEntry_GevCCP_ControlAccess
 - Transport Layer Enumerations, [275](#)
- GevCCP_EnumEntry_GevCCP_ExclusiveAccess
 - Transport Layer Enumerations, [275](#)
- GevCCP_EnumEntry_GevCCP_OpenAccess
 - Transport Layer Enumerations, [275](#)
- GevCCP_ExclusiveAccess
 - Camera Enumerations, [75](#)
- GevCCP_OpenAccess
 - Camera Enumerations, [75](#)
- GevCCP
 - quickSpin, [313](#)
 - quickSpinTLDevice, [325](#)
- GevCurrentDefaultGateway
 - quickSpin, [313](#)
- GevCurrentIPAddress
 - quickSpin, [313](#)
- GevCurrentIPConfigurationDHCP
 - quickSpin, [313](#)
- GevCurrentIPConfigurationLLA
 - quickSpin, [313](#)
- GevCurrentIPConfigurationPersistentIP
 - quickSpin, [313](#)
- GevCurrentPhysicalLinkConfiguration
 - quickSpin, [313](#)
- GevCurrentPhysicalLinkConfiguration_DynamicLAG
 - Camera Enumerations, [76](#)
- GevCurrentPhysicalLinkConfiguration_MultiLink
 - Camera Enumerations, [76](#)
- GevCurrentPhysicalLinkConfiguration_SingleLink
 - Camera Enumerations, [76](#)
- GevCurrentPhysicalLinkConfiguration_StaticLAG
 - Camera Enumerations, [76](#)
- GevCurrentSubnetMask
 - quickSpin, [313](#)
- GevDeviceDiscoverMaximumPacketSize
 - quickSpinTLDevice, [325](#)
- GevDeviceGateway
 - quickSpinTLDevice, [325](#)
- GevDeviceIPAddress
 - quickSpinTLDevice, [325](#)
 - quickSpinTLInterface, [327](#)
- GevDevicesWrongSubnet
 - quickSpinTLDevice, [325](#)
- GevDeviceMACAddress
 - quickSpinTLDevice, [325](#)
 - quickSpinTLInterface, [327](#)
- GevDeviceMaximumPacketSize
 - quickSpinTLDevice, [325](#)
- GevDeviceMaximumRetryCount
 - quickSpinTLDevice, [325](#)
- GevDeviceModelsBigEndian
 - quickSpinTLDevice, [325](#)
- GevDevicePort
 - quickSpinTLDevice, [325](#)
- GevDeviceReadAndWriteTimeout
 - quickSpinTLDevice, [325](#)
- GevDeviceSubnetMask
 - quickSpinTLDevice, [325](#)
 - quickSpinTLInterface, [327](#)
- GevDiscoveryAckDelay
 - quickSpin, [313](#)
- GevFailedPacketCount
 - quickSpinTLStream, [329](#)
- GevFirstURL
 - quickSpin, [313](#)
- GevGVCPExtendedStatusCodes
 - quickSpin, [313](#)
- GevGVCPExtendedStatusCodesSelector
 - quickSpin, [313](#)
- GevGVCPExtendedStatusCodesSelector_Version1_1
 - Camera Enumerations, [76](#)
- GevGVCPExtendedStatusCodesSelector_Version2_0
 - Camera Enumerations, [76](#)
- GevGVCPHeartbeatDisable
 - quickSpin, [313](#)
- GevGVCPPendingAck
 - quickSpin, [313](#)
- GevGVCPPendingTimeout
 - quickSpin, [313](#)
- GevGVSPExtendedIDMode
 - quickSpin, [313](#)
- GevGVSPExtendedIDMode_Off
 - Camera Enumerations, [76](#)
- GevGVSPExtendedIDMode_On
 - Camera Enumerations, [76](#)
- GevHeartbeatTimeout
 - quickSpin, [313](#)
- GevIEEE1588
 - quickSpin, [313](#)
- GevIEEE1588ClockAccuracy
 - quickSpin, [313](#)
- GevIEEE1588ClockAccuracy_Unknown
 - Camera Enumerations, [76](#)
- GevIEEE1588Mode
 - quickSpin, [313](#)
- GevIEEE1588Mode_Auto
 - Camera Enumerations, [77](#)

- GevIEEE1588Mode_SlaveOnly
 - Camera Enumerations, [77](#)
- GevIEEE1588Status
 - quickSpin, [313](#)
- GevIEEE1588Status_Disabled
 - Camera Enumerations, [77](#)
- GevIEEE1588Status_Faulty
 - Camera Enumerations, [77](#)
- GevIEEE1588Status_Initializing
 - Camera Enumerations, [77](#)
- GevIEEE1588Status_Listening
 - Camera Enumerations, [77](#)
- GevIEEE1588Status_Master
 - Camera Enumerations, [77](#)
- GevIEEE1588Status_Passive
 - Camera Enumerations, [77](#)
- GevIEEE1588Status_PreMaster
 - Camera Enumerations, [77](#)
- GevIEEE1588Status_Slave
 - Camera Enumerations, [77](#)
- GevIEEE1588Status_Uncalibrated
 - Camera Enumerations, [77](#)
- GevIPConfigurationStatus
 - quickSpin, [313](#)
- GevIPConfigurationStatus_DHCP
 - Camera Enumerations, [77](#)
- GevIPConfigurationStatus_ForceIP
 - Camera Enumerations, [77](#)
- GevIPConfigurationStatus_LLA
 - Camera Enumerations, [77](#)
- GevIPConfigurationStatus_None
 - Camera Enumerations, [77](#)
- GevIPConfigurationStatus_PersistentIP
 - Camera Enumerations, [77](#)
- GevInterfaceGateway
 - quickSpinTLInterface, [327](#)
- GevInterfaceIPAddress
 - quickSpinTLInterface, [327](#)
- GevInterfaceMACAddress
 - quickSpinTLInterface, [327](#)
- GevInterfaceSelector
 - quickSpin, [313](#)
- GevInterfaceSubnetMask
 - quickSpinTLInterface, [327](#)
- GevMACAddress
 - quickSpin, [314](#)
- GevMCDA
 - quickSpin, [314](#)
- GevMCPHostPort
 - quickSpin, [314](#)
- GevMCRC
 - quickSpin, [314](#)
- GevMCSP
 - quickSpin, [314](#)
- GevMCTT
 - quickSpin, [314](#)
- GevMaximumNumberResendBuffers
 - quickSpinTLStream, [329](#)
- GevMaximumNumberResendRequests
 - quickSpinTLStream, [329](#)
- GevNumberOfInterfaces
 - quickSpin, [314](#)
- GevPAUSEFrameReception
 - quickSpin, [314](#)
- GevPAUSEFrameTransmission
 - quickSpin, [314](#)
- GevPacketResendMode
 - quickSpinTLStream, [329](#)
- GevPacketResendTimeout
 - quickSpinTLStream, [329](#)
- GevPersistentDefaultGateway
 - quickSpin, [314](#)
- GevPersistentIPAddress
 - quickSpin, [314](#)
- GevPersistentSubnetMask
 - quickSpin, [314](#)
- GevPhysicalLinkConfiguration
 - quickSpin, [314](#)
- GevPhysicalLinkConfiguration_DynamicLAG
 - Camera Enumerations, [77](#)
- GevPhysicalLinkConfiguration_MultiLink
 - Camera Enumerations, [77](#)
- GevPhysicalLinkConfiguration_SingleLink
 - Camera Enumerations, [77](#)
- GevPhysicalLinkConfiguration_StaticLAG
 - Camera Enumerations, [77](#)
- GevPrimaryApplicationIPAddress
 - quickSpin, [314](#)
- GevPrimaryApplicationSocket
 - quickSpin, [314](#)
- GevPrimaryApplicationSwitchoverKey
 - quickSpin, [314](#)
- GevResendPacketCount
 - quickSpinTLStream, [329](#)
- GevResendRequestCount
 - quickSpinTLStream, [329](#)
- GevSCCFGAllInTransmission
 - quickSpin, [314](#)
- GevSCCFGExtendedChunkData
 - quickSpin, [314](#)
- GevSCCFGPacketResendDestination
 - quickSpin, [314](#)
- GevSCCFGUnconditionalStreaming
 - quickSpin, [314](#)
- GevSCDA
 - quickSpin, [314](#)
- GevSCPDirection
 - quickSpin, [314](#)
- GevSCPHostPort
 - quickSpin, [315](#)
- GevSCPInterfaceIndex
 - quickSpin, [315](#)
- GevSCPSBigEndian
 - quickSpin, [315](#)
- GevSCPSDoNotFragment
 - quickSpin, [315](#)

- GevSCPSFireTestPacket
 - quickSpin, [315](#)
- GevSCPSPacketSize
 - quickSpin, [315](#)
- GevSCPD
 - quickSpin, [314](#)
- GevSCSP
 - quickSpin, [315](#)
- GevSCZoneConfigurationLock
 - quickSpin, [315](#)
- GevSCZoneCount
 - quickSpin, [315](#)
- GevSCZoneDirectionAll
 - quickSpin, [315](#)
- GevSecondURL
 - quickSpin, [315](#)
- GevStreamChannelSelector
 - quickSpin, [315](#)
- GevSupportedOption
 - quickSpin, [315](#)
- GevSupportedOptionSelector
 - quickSpin, [315](#)
- GevSupportedOptionSelector_Action
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_CCPApplicationSocket
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_CommandsConcatenation
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_DiscoveryAckDelay
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_DiscoveryAckDelay↔
 - Writable
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_Event
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_EventData
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_ExtendedStatusCodes
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_HeartbeatDisable
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_IPConfigurationDHCP
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_IPConfigurationLLA
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_IPConfiguration↔
 - PersistentIP
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_LinkSpeed
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_ManifestTable
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_MessageChannel↔
 - SourceSocket
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_PacketResend
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_PendingAck
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_SerialNumber
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_StreamChannel↔
 - SourceSocket
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_TestData
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_UserDefinedName
 - Camera Enumerations, [78](#)
- GevSupportedOptionSelector_WriteMem
 - Camera Enumerations, [78](#)
- GevTimestampTickFrequency
 - quickSpin, [315](#)
- GevTotalPacketCount
 - quickSpinTLStream, [329](#)
- GevVersionMajor
 - quickSpinTLDevice, [325](#)
- GevVersionMinor
 - quickSpinTLDevice, [325](#)
- GuiXmlManifestAddress
 - quickSpin, [315](#)
- Guru
 - Spinnaker C GenICam Enumerations, [268](#)
- HQ_LINEAR
 - Spinnaker C Enumerations, [209](#)
- HUE
 - Spinnaker C Enumerations, [213](#)
- Height
 - quickSpin, [315](#)
- height
 - spinH264Option, [334](#)
- HeightMax
 - quickSpin, [315](#)
- HexNumber
 - Spinnaker C GenICam Enumerations, [267](#)
- HostAdapterDriverVersion
 - quickSpinTLInterface, [327](#)
- HostAdapterName
 - quickSpinTLInterface, [327](#)
- HostAdapterVendor
 - quickSpinTLInterface, [327](#)
- IBoolean Access, [251](#)
 - spinBooleanGetValue, [251](#)
 - spinBooleanSetValue, [252](#)
- ICategory Access, [255](#)
 - spinCategoryGetFeatureByIndex, [255](#)
 - spinCategoryGetNumFeatures, [256](#)
- ICommand Access, [253](#)
 - spinCommandExecute, [253](#)
 - spinCommandIsDone, [253](#)
- IEnumEntry Access, [249](#)
 - spinEnumerationEntryGetEnumValue, [249](#)
 - spinEnumerationEntryGetIntValue, [250](#)
 - spinEnumerationEntryGetSymbolic, [250](#)
- IEnumeration Access, [245](#)
 - spinEnumerationGetCurrentEntry, [245](#)

- spinEnumerationGetEntryByIndex, [246](#)
- spinEnumerationGetEntryByName, [246](#)
- spinEnumerationGetNumEntries, [246](#)
- spinEnumerationSetEnumValue, [247](#)
- spinEnumerationSetIntValue, [247](#)
- IFloat Access, [241](#)
 - spinFloatGetMax, [241](#)
 - spinFloatGetMin, [242](#)
 - spinFloatGetRepresentation, [242](#)
 - spinFloatGetUnit, [242](#)
 - spinFloatGetValue, [243](#)
 - spinFloatGetValueEx, [243](#)
 - spinFloatSetValue, [244](#)
 - spinFloatSetValueEx, [244](#)
- IIDC
 - Spinnaker C GenICam Enumerations, [268](#)
- Integer Access, [237](#)
 - spinIntegerGetInc, [237](#)
 - spinIntegerGetMax, [238](#)
 - spinIntegerGetMin, [238](#)
 - spinIntegerGetRepresentation, [238](#)
 - spinIntegerGetValue, [239](#)
 - spinIntegerGetValueEx, [239](#)
 - spinIntegerSetValue, [240](#)
 - spinIntegerSetValueEx, [240](#)
- IMAGE_CHUNK_DATA_INVALID
 - Spinnaker C Enumerations, [211](#)
- IMAGE_CRC_CHECK_FAILED
 - Spinnaker C Enumerations, [211](#)
- IMAGE_DATA_INCOMPLETE
 - Spinnaker C Enumerations, [211](#)
- IMAGE_DATA_OVERFLOW
 - Spinnaker C Enumerations, [211](#)
- IMAGE_FILE_FORMAT_FORCE_32BITS
 - Spinnaker C Enumerations, [211](#)
- IMAGE_INFO_INCONSISTENT
 - Spinnaker C Enumerations, [211](#)
- IMAGE_LEADER_BUFFER_SIZE_INCONSISTENT
 - Spinnaker C Enumerations, [211](#)
- IMAGE_MISSING_LEADER
 - Spinnaker C Enumerations, [211](#)
- IMAGE_MISSING_PACKETS
 - Spinnaker C Enumerations, [211](#)
- IMAGE_MISSING_TRAILER
 - Spinnaker C Enumerations, [211](#)
- IMAGE_NO_ERROR
 - Spinnaker C Enumerations, [211](#)
- IMAGE_NO_SYSTEM_RESOURCES
 - Spinnaker C Enumerations, [211](#)
- IMAGE_PACKETID_INCONSISTENT
 - Spinnaker C Enumerations, [211](#)
- IMAGE_TRAILER_BUFFER_SIZE_INCONSISTENT
 - Spinnaker C Enumerations, [211](#)
- IMAGE_UNKNOWN_ERROR
 - Spinnaker C Enumerations, [211](#)
- IPV4Address
 - Spinnaker C GenICam Enumerations, [267](#)
- IPP
 - Spinnaker C Enumerations, [209](#)
- IRegister Access, [257](#)
 - spinRegisterGet, [257](#)
 - spinRegisterGetAddress, [258](#)
 - spinRegisterGetEx, [258](#)
 - spinRegisterGetLength, [259](#)
 - spinRegisterSet, [259](#)
 - spinRegisterSetEx, [259](#)
 - spinRegisterSetReference, [260](#)
- IValue Access, [231](#)
 - spinNodeFromString, [231](#)
 - spinNodeFromStringEx, [232](#)
 - spinNodeToString, [232](#)
 - spinNodeToStringEx, [232](#)
- idFrom
 - Spinnaker C GenICam Enumerations, [266](#)
- idNone
 - Spinnaker C GenICam Enumerations, [266](#)
- idTo
 - Spinnaker C GenICam Enumerations, [266](#)
- Image Access, [155](#)
 - spinImageCalculateStatistics, [157](#)
 - spinImageCheckCRC, [158](#)
 - spinImageConvert, [158](#)
 - spinImageConvertEx, [158](#)
 - spinImageCreate, [159](#)
 - spinImageCreateEmpty, [159](#)
 - spinImageCreateEx, [159](#)
 - spinImageDeepCopy, [160](#)
 - spinImageDestroy, [160](#)
 - spinImageGetBitsPerPixel, [161](#)
 - spinImageGetBufferSize, [161](#)
 - spinImageGetChunkLayoutID, [161](#)
 - spinImageGetColorProcessing, [162](#)
 - spinImageGetData, [162](#)
 - spinImageGetDefaultColorProcessing, [163](#)
 - spinImageGetFrameID, [163](#)
 - spinImageGetHeight, [163](#)
 - spinImageGetID, [164](#)
 - spinImageGetOffsetX, [164](#)
 - spinImageGetOffsetY, [164](#)
 - spinImageGetPaddingX, [165](#)
 - spinImageGetPaddingY, [165](#)
 - spinImageGetPayloadType, [165](#)
 - spinImageGetPixelFormat, [166](#)
 - spinImageGetPixelFormatName, [166](#)
 - spinImageGetPrivateData, [167](#)
 - spinImageGetSize, [167](#)
 - spinImageGetStatus, [167](#)
 - spinImageGetStatusDescription, [168](#)
 - spinImageGetStride, [168](#)
 - spinImageGetTLPayloadType, [169](#)
 - spinImageGetTLPixelFormat, [169](#)
 - spinImageGetTLPixelFormatNamespace, [170](#)
 - spinImageGetTimeStamp, [169](#)
 - spinImageGetValidPayloadSize, [170](#)
 - spinImageGetWidth, [171](#)
 - spinImageHasCRC, [171](#)

- spinImageIsIncomplete, [171](#)
- spinImageRelease, [172](#)
- spinImageReset, [172](#)
- spinImageResetEx, [173](#)
- spinImageSave, [173](#)
- spinImageSaveBmp, [173](#)
- spinImageSaveFromExt, [174](#)
- spinImageSaveJpeg, [174](#)
- spinImageSaveJpg2, [175](#)
- spinImageSavePgm, [175](#)
- spinImageSavePng, [176](#)
- spinImageSavePpm, [176](#)
- spinImageSaveTiff, [176](#)
- spinImageSetDefaultColorProcessing, [177](#)
- ImageComponentEnable
 - quickSpin, [315](#)
- ImageComponentSelector
 - quickSpin, [315](#)
- ImageComponentSelector_Color
 - Camera Enumerations, [78](#)
- ImageComponentSelector_Confidence
 - Camera Enumerations, [78](#)
- ImageComponentSelector_Disparity
 - Camera Enumerations, [78](#)
- ImageComponentSelector_Infrared
 - Camera Enumerations, [78](#)
- ImageComponentSelector_Intensity
 - Camera Enumerations, [78](#)
- ImageComponentSelector_Range
 - Camera Enumerations, [78](#)
- ImageComponentSelector_Scatter
 - Camera Enumerations, [79](#)
- ImageComponentSelector_Ultraviolet
 - Camera Enumerations, [78](#)
- ImageCompressionBitrate
 - quickSpin, [315](#)
- ImageCompressionJPEGFormatOption
 - quickSpin, [315](#)
- ImageCompressionJPEGFormatOption_Baseline↵
 - Optimized
 - Camera Enumerations, [79](#)
- ImageCompressionJPEGFormatOption_Baseline↵
 - Standard
 - Camera Enumerations, [79](#)
- ImageCompressionJPEGFormatOption_Lossless
 - Camera Enumerations, [79](#)
- ImageCompressionJPEGFormatOption_Progressive
 - Camera Enumerations, [79](#)
- ImageCompressionMode
 - quickSpin, [315](#)
- ImageCompressionMode_Lossless
 - Camera Enumerations, [79](#)
- ImageCompressionMode_Off
 - Camera Enumerations, [79](#)
- ImageCompressionQuality
 - quickSpin, [316](#)
- ImageCompressionRateOption
 - quickSpin, [316](#)
- ImageCompressionRateOption_FixBitrate
 - Camera Enumerations, [79](#)
- ImageCompressionRateOption_FixQuality
 - Camera Enumerations, [79](#)
- ImageStatistics Access, [184](#)
 - spinImageStatisticsCreate, [185](#)
 - spinImageStatisticsDestroy, [185](#)
 - spinImageStatisticsDisableAll, [185](#)
 - spinImageStatisticsEnableAll, [185](#)
 - spinImageStatisticsEnableGreyOnly, [186](#)
 - spinImageStatisticsEnableHslOnly, [186](#)
 - spinImageStatisticsEnableRgbOnly, [187](#)
 - spinImageStatisticsGetAll, [187](#)
 - spinImageStatisticsGetChannelStatus, [187](#)
 - spinImageStatisticsGetHistogram, [188](#)
 - spinImageStatisticsGetMean, [188](#)
 - spinImageStatisticsGetNumPixelValues, [189](#)
 - spinImageStatisticsGetPixelValueRange, [189](#)
 - spinImageStatisticsGetRange, [190](#)
 - spinImageStatisticsSetChannelStatus, [190](#)
- include/spinc/CameraDefsC.h, [343](#)
- include/spinc/ChunkDataDefC.h, [374](#)
- include/spinc/QuickSpinC.h, [375](#)
- include/spinc/QuickSpinDefsC.h, [375](#)
- include/spinc/SpinVideoC.h, [399](#)
- include/spinc/SpinnakerC.h, [376](#)
- include/spinc/SpinnakerDefsC.h, [385](#)
- include/spinc/SpinnakerGenApiC.h, [391](#)
- include/spinc/SpinnakerGenApiDefsC.h, [395](#)
- include/spinc/SpinnakerPlatformC.h, [398](#)
- include/spinc/TransportLayerDefsC.h, [399](#)
- include/spinc/TransportLayerDeviceC.h, [401](#)
- include/spinc/TransportLayerInterfaceC.h, [402](#)
- include/spinc/TransportLayerStreamC.h, [403](#)
- IncompatibleDeviceCount
 - quickSpinTLInterface, [327](#)
- IncompatibleDeviceID
 - quickSpinTLInterface, [327](#)
- IncompatibleDeviceModelName
 - quickSpinTLInterface, [327](#)
- IncompatibleDeviceSelector
 - quickSpinTLInterface, [327](#)
- IncompatibleDeviceVendorName
 - quickSpinTLInterface, [328](#)
- IncompatibleGevDeviceIPAddress
 - quickSpinTLInterface, [328](#)
- IncompatibleGevDeviceMACAddress
 - quickSpinTLInterface, [328](#)
- IncompatibleGevDeviceSubnetMask
 - quickSpinTLInterface, [328](#)
- Increasing
 - Spinnaker C GenICam Enumerations, [268](#)
- indexedColor_8bit
 - spinBMPOption, [331](#)
- IntegerNode
 - Spinnaker C GenICam Enumerations, [267](#)
- Interface Access, [138](#)
 - spinInterfaceGetCameras, [139](#)

- spinInterfaceGetCamerasEx, [139](#)
- spinInterfaceGetTLNodeMap, [140](#)
- spinInterfaceIsInUse, [140](#)
- spinInterfaceRegisterArrivalEvent, [140](#)
- spinInterfaceRegisterInterfaceEvent, [141](#)
- spinInterfaceRegisterRemovalEvent, [141](#)
- spinInterfaceRelease, [141](#)
- spinInterfaceSendActionCommand, [142](#)
- spinInterfaceUnregisterArrivalEvent, [142](#)
- spinInterfaceUnregisterInterfaceEvent, [143](#)
- spinInterfaceUnregisterRemovalEvent, [143](#)
- spinInterfaceUpdateCameras, [144](#)
- InterfaceDisplayName
 - quickSpinTLInterface, [328](#)
- InterfaceID
 - quickSpinTLInterface, [328](#)
- InterfaceList Access, [130](#)
 - spinInterfaceListClear, [130](#)
 - spinInterfaceListCreateEmpty, [131](#)
 - spinInterfaceListDestroy, [131](#)
 - spinInterfaceListGet, [131](#)
 - spinInterfaceListGetSize, [132](#)
- InterfaceType
 - quickSpinTLInterface, [328](#)
- interlaced
 - spinPNGOption, [339](#)
- intfIBase
 - Spinnaker C GenICam Enumerations, [266](#)
- intfIBoolean
 - Spinnaker C GenICam Enumerations, [266](#)
- intfICategory
 - Spinnaker C GenICam Enumerations, [266](#)
- intfICommand
 - Spinnaker C GenICam Enumerations, [266](#)
- intfIEnumEntry
 - Spinnaker C GenICam Enumerations, [266](#)
- intfIEnumeration
 - Spinnaker C GenICam Enumerations, [266](#)
- intfIFloat
 - Spinnaker C GenICam Enumerations, [266](#)
- intfIInteger
 - Spinnaker C GenICam Enumerations, [266](#)
- intfIPort
 - Spinnaker C GenICam Enumerations, [266](#)
- intfIRegister
 - Spinnaker C GenICam Enumerations, [266](#)
- intfIString
 - Spinnaker C GenICam Enumerations, [266](#)
- intfIValue
 - Spinnaker C GenICam Enumerations, [266](#)
- Invisible
 - Spinnaker C GenICam Enumerations, [268](#)
- IspEnable
 - quickSpin, [316](#)
- JPEG2000
 - Spinnaker C Enumerations, [211](#)
- JPEG
 - Spinnaker C Enumerations, [211](#)
- JPG
 - Spinnaker C Structures, [215](#)
- LIGHTNESS
 - Spinnaker C Enumerations, [213](#)
- LOG_LEVEL_ALERT
 - Spinnaker C Enumerations, [212](#)
- LOG_LEVEL_CRIT
 - Spinnaker C Enumerations, [212](#)
- LOG_LEVEL_DEBUG
 - Spinnaker C Enumerations, [212](#)
- LOG_LEVEL_ERROR
 - Spinnaker C Enumerations, [212](#)
- LOG_LEVEL_FATAL
 - Spinnaker C Enumerations, [212](#)
- LOG_LEVEL_INFO
 - Spinnaker C Enumerations, [212](#)
- LOG_LEVEL_NOTICE
 - Spinnaker C Enumerations, [212](#)
- LOG_LEVEL_NOTSET
 - Spinnaker C Enumerations, [212](#)
- LOG_LEVEL_OFF
 - Spinnaker C Enumerations, [212](#)
- LOG_LEVEL_WARN
 - Spinnaker C Enumerations, [212](#)
- LUTEnable
 - quickSpin, [317](#)
- LUTIndex
 - quickSpin, [317](#)
- LUTSelector
 - quickSpin, [317](#)
- LUTSelector_LUT1
 - Camera Enumerations, [83](#)
- LUTValue
 - quickSpin, [317](#)
- LUTValueAll
 - quickSpin, [317](#)
- LZW
 - Spinnaker C Structures, [215](#)
- LineFilterWidth
 - quickSpin, [316](#)
- LineFormat
 - quickSpin, [316](#)
- LineFormat_LVDS
 - Camera Enumerations, [80](#)
- LineFormat_NoConnect
 - Camera Enumerations, [80](#)
- LineFormat_OpenDrain
 - Camera Enumerations, [80](#)
- LineFormat_OptoCoupled
 - Camera Enumerations, [80](#)
- LineFormat_RS422
 - Camera Enumerations, [80](#)
- LineFormat_TTL
 - Camera Enumerations, [80](#)
- LineFormat_TriState
 - Camera Enumerations, [80](#)
- LineInputFilterSelector
 - quickSpin, [316](#)

- LineInputFilterSelector_Debounce
 - Camera Enumerations, [80](#)
- LineInputFilterSelector_Deglitch
 - Camera Enumerations, [80](#)
- LineInverter
 - quickSpin, [316](#)
- LineMode
 - quickSpin, [316](#)
- LineMode_Input
 - Camera Enumerations, [80](#)
- LineMode_Output
 - Camera Enumerations, [80](#)
- LinePitch
 - quickSpin, [316](#)
- LineSelector
 - quickSpin, [316](#)
- LineSelector_Line0
 - Camera Enumerations, [80](#)
- LineSelector_Line1
 - Camera Enumerations, [80](#)
- LineSelector_Line2
 - Camera Enumerations, [80](#)
- LineSelector_Line3
 - Camera Enumerations, [80](#)
- LineSource
 - quickSpin, [316](#)
- LineSource_AllPixel
 - Camera Enumerations, [81](#)
- LineSource_AnyPixel
 - Camera Enumerations, [81](#)
- LineSource_Counter0Active
 - Camera Enumerations, [81](#)
- LineSource_Counter1Active
 - Camera Enumerations, [81](#)
- LineSource_ExposureActive
 - Camera Enumerations, [81](#)
- LineSource_FrameTriggerWait
 - Camera Enumerations, [81](#)
- LineSource_Line0
 - Camera Enumerations, [81](#)
- LineSource_Line1
 - Camera Enumerations, [81](#)
- LineSource_Line2
 - Camera Enumerations, [81](#)
- LineSource_Line3
 - Camera Enumerations, [81](#)
- LineSource_LogicBlock0
 - Camera Enumerations, [81](#)
- LineSource_LogicBlock1
 - Camera Enumerations, [81](#)
- LineSource_Off
 - Camera Enumerations, [81](#)
- LineSource_PPSSignal
 - Camera Enumerations, [81](#)
- LineSource_SerialPort0
 - Camera Enumerations, [81](#)
- LineSource_UserOutput0
 - Camera Enumerations, [81](#)
- LineSource_UserOutput1
 - Camera Enumerations, [81](#)
- LineSource_UserOutput2
 - Camera Enumerations, [81](#)
- LineSource_UserOutput3
 - Camera Enumerations, [81](#)
- LineStatus
 - quickSpin, [316](#)
- LineStatusAll
 - quickSpin, [316](#)
- Linear
 - Spinnaker C GenICam Enumerations, [267](#)
- LinkErrorCount
 - quickSpin, [316](#)
- LinkUptime
 - quickSpin, [316](#)
- listIncrement
 - Spinnaker C GenICam Enumerations, [265](#)
- LittleEndian
 - Spinnaker C GenICam Enumerations, [265](#)
- Logarithmic
 - Spinnaker C GenICam Enumerations, [267](#)
- Logging Event Data Access, [191](#)
 - spinLogDataGetCategoryName, [191](#)
 - spinLogDataGetLogMessage, [192](#)
 - spinLogDataGetNDC, [192](#)
 - spinLogDataGetPriority, [192](#)
 - spinLogDataGetPriorityName, [193](#)
 - spinLogDataGetThreadName, [193](#)
 - spinLogDataGetTimestamp, [194](#)
- LogicBlockLUTInputActivation
 - quickSpin, [316](#)
- LogicBlockLUTInputActivation_AnyEdge
 - Camera Enumerations, [81](#)
- LogicBlockLUTInputActivation_FallingEdge
 - Camera Enumerations, [81](#)
- LogicBlockLUTInputActivation_LevelHigh
 - Camera Enumerations, [81](#)
- LogicBlockLUTInputActivation_LevelLow
 - Camera Enumerations, [81](#)
- LogicBlockLUTInputActivation_RisingEdge
 - Camera Enumerations, [81](#)
- LogicBlockLUTInputSelector
 - quickSpin, [316](#)
- LogicBlockLUTInputSelector_Input0
 - Camera Enumerations, [81](#)
- LogicBlockLUTInputSelector_Input1
 - Camera Enumerations, [81](#)
- LogicBlockLUTInputSelector_Input2
 - Camera Enumerations, [81](#)
- LogicBlockLUTInputSelector_Input3
 - Camera Enumerations, [81](#)
- LogicBlockLUTInputSource
 - quickSpin, [316](#)
- LogicBlockLUTInputSource_AcquisitionActive
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_Counter0End
 - Camera Enumerations, [82](#)

- LogicBlockLUTInputSource_Counter0Start
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_Counter1End
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_Counter1Start
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_ExposureEnd
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_ExposureStart
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_FrameTriggerWait
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_Line0
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_Line1
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_Line2
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_Line3
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_LogicBlock0
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_LogicBlock1
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_UserOutput0
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_UserOutput1
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_UserOutput2
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_UserOutput3
 - Camera Enumerations, [82](#)
- LogicBlockLUTInputSource_Zero
 - Camera Enumerations, [82](#)
- LogicBlockLUTOutputValue
 - quickSpin, [316](#)
- LogicBlockLUTOutputValueAll
 - quickSpin, [316](#)
- LogicBlockLUTRowIndex
 - quickSpin, [316](#)
- LogicBlockLUTSelector
 - quickSpin, [316](#)
- LogicBlockLUTSelector_Enable
 - Camera Enumerations, [82](#)
- LogicBlockLUTSelector_Value
 - Camera Enumerations, [82](#)
- LogicBlockSelector
 - quickSpin, [316](#)
- LogicBlockSelector_LogicBlock0
 - Camera Enumerations, [82](#)
- LogicBlockSelector_LogicBlock1
 - Camera Enumerations, [82](#)
- m_blackLevel
 - spinChunkData, [332](#)
- m_cRC
 - spinChunkData, [332](#)
- m_counterValue
 - spinChunkData, [332](#)
- m_encoderValue
 - spinChunkData, [332](#)
- m_exposureEndLineStatusAll
 - spinChunkData, [332](#)
- m_exposureTime
 - spinChunkData, [332](#)
- m_frameID
 - spinChunkData, [332](#)
- m_gain
 - spinChunkData, [332](#)
- m_height
 - spinChunkData, [332](#)
- m_image
 - spinChunkData, [332](#)
- m_inferenceConfidence
 - spinChunkData, [332](#)
- m_inferenceResult
 - spinChunkData, [332](#)
- m_linePitch
 - spinChunkData, [332](#)
- m_lineStatusAll
 - spinChunkData, [332](#)
- m_offsetX
 - spinChunkData, [332](#)
- m_offsetY
 - spinChunkData, [333](#)
- m_partSelector
 - spinChunkData, [333](#)
- m_pixelDynamicRangeMax
 - spinChunkData, [333](#)
- m_pixelDynamicRangeMin
 - spinChunkData, [333](#)
- m_scan3dAxisMax
 - spinChunkData, [333](#)
- m_scan3dAxisMin
 - spinChunkData, [333](#)
- m_scan3dCoordinateOffset
 - spinChunkData, [333](#)
- m_scan3dCoordinateReferenceValue
 - spinChunkData, [333](#)
- m_scan3dCoordinateScale
 - spinChunkData, [333](#)
- m_scan3dInvalidDataValue
 - spinChunkData, [333](#)
- m_scan3dTransformValue
 - spinChunkData, [333](#)
- m_scanLineSelector
 - spinChunkData, [333](#)
- m_sequencerSetActive
 - spinChunkData, [333](#)
- m_serialDataLength
 - spinChunkData, [333](#)
- m_streamChannelID
 - spinChunkData, [333](#)
- m_timerValue
 - spinChunkData, [333](#)
- m_timestamp
 - spinChunkData, [333](#)

- m_timestampLatchValue
 - spinChunkData, [333](#)
- m_transferBlockID
 - spinChunkData, [333](#)
- m_transferQueueCurrentBlockCount
 - spinChunkData, [333](#)
- m_width
 - spinChunkData, [333](#)
- MACAddress
 - Spinnaker C GenICam Enumerations, [267](#)
- major
 - spinLibraryVersion, [337](#)
- MaxDeviceResetTime
 - quickSpin, [317](#)
- minor
 - spinLibraryVersion, [337](#)
- NEAREST_NEIGHBOR
 - Spinnaker C Enumerations, [209](#)
- NO_COLOR_PROCESSING
 - Spinnaker C Enumerations, [209](#)
- NONE
 - Spinnaker C Structures, [215](#)
- NUM_ACQUISITIONMODE
 - Camera Enumerations, [43](#)
- NUM_ACQUISITIONSTATUSSELECTOR
 - Camera Enumerations, [44](#)
- NUM_ACTIONUNCONDITIONALMODE
 - Camera Enumerations, [44](#)
- NUM_ADCBITDEPTH
 - Camera Enumerations, [44](#)
- NUM_AUTOALGORITHMSELECTOR
 - Camera Enumerations, [44](#)
- NUM_AUTOEXPOSURECONTROLPRIORITY
 - Camera Enumerations, [45](#)
- NUM_AUTOEXPOSURELIGHTINGMODE
 - Camera Enumerations, [45](#)
- NUM_AUTOEXPOSUREMETERINGMODE
 - Camera Enumerations, [45](#)
- NUM_AUTOEXPOSURETARGETGREYVALUEAUTO
 - Camera Enumerations, [46](#)
- NUM_BALANCERATIOSELECTOR
 - Camera Enumerations, [46](#)
- NUM_BALANCEWHITEAUTOPROFILE
 - Camera Enumerations, [46](#)
- NUM_BALANCEWHITEAUTO
 - Camera Enumerations, [46](#)
- NUM_BINNINGHORIZONTALMODE
 - Camera Enumerations, [47](#)
- NUM_BINNINGSELECTOR
 - Camera Enumerations, [47](#)
- NUM_BINNINGVERTICALMODE
 - Camera Enumerations, [47](#)
- NUM_BLACKLEVELAUTOBALANCE
 - Camera Enumerations, [47](#)
- NUM_BLACKLEVELAUTO
 - Camera Enumerations, [48](#)
- NUM_BLACKLEVELSELECTOR
 - Camera Enumerations, [48](#)
- NUM_CHUNKBLACKLEVELSELECTOR
 - Camera Enumerations, [48](#)
- NUM_CHUNKCOUNTERSELECTOR
 - Camera Enumerations, [48](#)
- NUM_CHUNKENCODERSELECTOR
 - Camera Enumerations, [49](#)
- NUM_CHUNKENCODERSTATUS
 - Camera Enumerations, [49](#)
- NUM_CHUNKEXPOSURETIMESELECTOR
 - Camera Enumerations, [49](#)
- NUM_CHUNKGAINSELECTOR
 - Camera Enumerations, [50](#)
- NUM_CHUNKIMAGECOMPONENT
 - Camera Enumerations, [50](#)
- NUM_CHUNKPIXELFORMAT
 - Camera Enumerations, [50](#)
- NUM_CHUNKREGIONID
 - Camera Enumerations, [51](#)
- NUM_CHUNKSCAN3DCOORDINATEREFERENCE↔SELECTOR
 - Camera Enumerations, [51](#)
- NUM_CHUNKSCAN3DCOORDINATESELECTOR
 - Camera Enumerations, [51](#)
- NUM_CHUNKSCAN3DCOORDINATESYSTEMREF↔ERENCE
 - Camera Enumerations, [52](#)
- NUM_CHUNKSCAN3DCOORDINATESYSTEM
 - Camera Enumerations, [51](#)
- NUM_CHUNKSCAN3DCOORDINATETTRANSFORM↔SELECTOR
 - Camera Enumerations, [52](#)
- NUM_CHUNKSCAN3DDISTANCEUNIT
 - Camera Enumerations, [52](#)
- NUM_CHUNKSCAN3DOUTPUTMODE
 - Camera Enumerations, [53](#)
- NUM_CHUNKSELECTOR
 - Camera Enumerations, [54](#)
- NUM_CHUNKSOURCEID
 - Camera Enumerations, [54](#)
- NUM_CHUNKTIMERSELECTOR
 - Camera Enumerations, [54](#)
- NUM_CHUNKTRANSFERSTREAMID
 - Camera Enumerations, [54](#)
- NUM_CLCONFIGURATION
 - Camera Enumerations, [55](#)
- NUM_CLTIMESLOTSCOUNT
 - Camera Enumerations, [55](#)
- NUM_COLORTRANSFORMATIONSELECTOR
 - Camera Enumerations, [55](#)
- NUM_COLORTRANSFORMATIONVALUESELECTOR
 - Camera Enumerations, [56](#)
- NUM_COUNTEREVENTACTIVATION
 - Camera Enumerations, [56](#)
- NUM_COUNTEREVENTSOURCE
 - Camera Enumerations, [57](#)
- NUM_COUNTERRESETACTIVATION
 - Camera Enumerations, [57](#)
- NUM_COUNTERRESETSOURCE

- Camera Enumerations, [57](#)
- NUM_COUNTERSELECTOR
 - Camera Enumerations, [58](#)
- NUM_COUNTERSTATUS
 - Camera Enumerations, [58](#)
- NUM_COUNTERTRIGGERACTIVATION
 - Camera Enumerations, [58](#)
- NUM_COUNTERTRIGGERSOURCE
 - Camera Enumerations, [59](#)
- NUM_CXPCONNECTIONTESTMODE
 - Camera Enumerations, [59](#)
- NUM_CXPLINKCONFIGURATIONPREFERRED
 - Camera Enumerations, [61](#)
- NUM_CXPLINKCONFIGURATIONSTATUS
 - Camera Enumerations, [62](#)
- NUM_CXPLINKCONFIGURATION
 - Camera Enumerations, [60](#)
- NUM_CXPPOCXPSTATUS
 - Camera Enumerations, [62](#)
- NUM_DECIMATIONHORIZONTALMODE
 - Camera Enumerations, [62](#)
- NUM_DECIMATIONSELECTOR
 - Camera Enumerations, [63](#)
- NUM_DECIMATIONVERTICALMODE
 - Camera Enumerations, [63](#)
- NUM_DEFECTCORRECTIONMODE
 - Camera Enumerations, [63](#)
- NUM_DEINTERLACING
 - Camera Enumerations, [63](#)
- NUM_DEVICECHARACTERSET
 - Camera Enumerations, [64](#)
- NUM_DEVICECLOCKSELECTOR
 - Camera Enumerations, [64](#)
- NUM_DEVICECONNECTIONSTATUS
 - Camera Enumerations, [64](#)
- NUM_DEVICEINDICATORMODE
 - Camera Enumerations, [64](#)
- NUM_DEVICELINKHEARTBEATMODE
 - Camera Enumerations, [65](#)
- NUM_DEVICELINKTHROUGHPUTLIMITMODE
 - Camera Enumerations, [65](#)
- NUM_DEVICEPOWERSUPPLYSELECTOR
 - Camera Enumerations, [65](#)
- NUM_DEVICEREGISTERSENDIANNESS
 - Camera Enumerations, [65](#)
- NUM_DEVICESCANTYPE
 - Camera Enumerations, [65](#)
- NUM_DEVICESERIALPORTBAUDRATE
 - Camera Enumerations, [66](#)
- NUM_DEVICESERIALPORTSELECTOR
 - Camera Enumerations, [66](#)
- NUM_DEVICESTREAMCHANNELENDIANNESS
 - Camera Enumerations, [66](#)
- NUM_DEVICESTREAMCHANNELTYPE
 - Camera Enumerations, [66](#)
- NUM_DEVICETAPGEOMETRY
 - Camera Enumerations, [68](#)
- NUM_DEVICETEMPERATURESELECTOR
 - Camera Enumerations, [68](#)
- NUM_DEVICETLTYPE
 - Camera Enumerations, [68](#)
- NUM_DEVICETYPE
 - Camera Enumerations, [68](#)
- NUM_ENCODERMODE
 - Camera Enumerations, [69](#)
- NUM_ENCODEROUTPUTMODE
 - Camera Enumerations, [69](#)
- NUM_ENCODERRESETACTIVATION
 - Camera Enumerations, [69](#)
- NUM_ENCODERRESETSOURCE
 - Camera Enumerations, [70](#)
- NUM_ENCODERSELECTOR
 - Camera Enumerations, [71](#)
- NUM_ENCODERSOURCEA
 - Camera Enumerations, [71](#)
- NUM_ENCODERSOURCEB
 - Camera Enumerations, [71](#)
- NUM_ENCODERSTATUS
 - Camera Enumerations, [71](#)
- NUM_EVENTNOTIFICATION
 - Camera Enumerations, [72](#)
- NUM_EVENTSELECTOR
 - Camera Enumerations, [72](#)
- NUM_EXPOSUREACTIVEMODE
 - Camera Enumerations, [72](#)
- NUM_EXPOSUREAUTO
 - Camera Enumerations, [72](#)
- NUM_EXPOSUREMODE
 - Camera Enumerations, [73](#)
- NUM_EXPOSURETIMEMODE
 - Camera Enumerations, [73](#)
- NUM_EXPOSURETIMESELECTOR
 - Camera Enumerations, [73](#)
- NUM_FILEOPENMODE
 - Camera Enumerations, [74](#)
- NUM_FILEOPERATIONSELECTOR
 - Camera Enumerations, [74](#)
- NUM_FILEOPERATIONSTATUS
 - Camera Enumerations, [74](#)
- NUM_FILESELECTOR
 - Camera Enumerations, [74](#)
- NUM_GAINAUTOBALANCE
 - Camera Enumerations, [75](#)
- NUM_GAINAUTO
 - Camera Enumerations, [75](#)
- NUM_GAINSELECTOR
 - Camera Enumerations, [75](#)
- NUM_GEVCCP
 - Camera Enumerations, [75](#)
- NUM_GEVCURRENTPHYSICALLINKCONFIGURATION
 - Camera Enumerations, [76](#)
- NUM_GEVGVCPEXTENDEDSTATUSCODESSELECTOR
 - Camera Enumerations, [76](#)
- NUM_GEVGVSPEXTENDEDIDMODE

- Camera Enumerations, [76](#)
- NUM_GEVIEEE1588CLOCKACCURACY
 - Camera Enumerations, [76](#)
- NUM_GEVIEEE1588MODE
 - Camera Enumerations, [77](#)
- NUM_GEVIEEE1588STATUS
 - Camera Enumerations, [77](#)
- NUM_GEVIPCONFIGURATIONSTATUS
 - Camera Enumerations, [77](#)
- NUM_GEVPHYSICALLINKCONFIGURATION
 - Camera Enumerations, [77](#)
- NUM_GEVSUPPORTEDOPTIONSELECTOR
 - Camera Enumerations, [78](#)
- NUM_IMAGECOMPONENTSELECTOR
 - Camera Enumerations, [79](#)
- NUM_IMAGECOMPRESSIONJPEGFORMATOPTION
 - Camera Enumerations, [79](#)
- NUM_IMAGECOMPRESSIONMODE
 - Camera Enumerations, [79](#)
- NUM_IMAGECOMPRESSIONRATEOPTION
 - Camera Enumerations, [79](#)
- NUM_LINEFORMAT
 - Camera Enumerations, [80](#)
- NUM_LINEINPUTFILTERSELECTOR
 - Camera Enumerations, [80](#)
- NUM_LINEMODE
 - Camera Enumerations, [80](#)
- NUM_LINESELECTOR
 - Camera Enumerations, [80](#)
- NUM_LINESOURCE
 - Camera Enumerations, [81](#)
- NUM_LOGICBLOCKLUTINPUTACTIVATION
 - Camera Enumerations, [81](#)
- NUM_LOGICBLOCKLUTINPUTSELECTOR
 - Camera Enumerations, [81](#)
- NUM_LOGICBLOCKLUTINPUTSOURCE
 - Camera Enumerations, [82](#)
- NUM_LOGICBLOCKLUTSELECTOR
 - Camera Enumerations, [82](#)
- NUM_LOGICBLOCKSELECTOR
 - Camera Enumerations, [82](#)
- NUM_LUTSELECTOR
 - Camera Enumerations, [83](#)
- NUM_PIXELCOLORFILTER
 - Camera Enumerations, [83](#)
- NUM_PIXELFORMATINFOSELECTOR
 - Camera Enumerations, [94](#)
- NUM_PIXELFORMAT
 - Camera Enumerations, [88](#)
- NUM_PIXELSIZE
 - Camera Enumerations, [94](#)
- NUM_REGIONDESTINATION
 - Camera Enumerations, [94](#)
- NUM_REGIONMODE
 - Camera Enumerations, [95](#)
- NUM_REGIONSELECTOR
 - Camera Enumerations, [95](#)
- NUM_RGBTRANSFORMLIGHTSOURCE
 - Camera Enumerations, [95](#)
- NUM_SCAN3DCOORDINATEREFERENCESELECTOR
 - Camera Enumerations, [96](#)
- NUM_SCAN3DCOORDINATESELECTOR
 - Camera Enumerations, [96](#)
- NUM_SCAN3DCOORDINATESYSTEMREFERENCE
 - Camera Enumerations, [96](#)
- NUM_SCAN3DCOORDINATESYSTEM
 - Camera Enumerations, [96](#)
- NUM_SCAN3DCOORDINATETRANSFORMSELECTOR
 - Camera Enumerations, [97](#)
- NUM_SCAN3DDISTANCEUNIT
 - Camera Enumerations, [97](#)
- NUM_SCAN3DOUTPUTMODE
 - Camera Enumerations, [98](#)
- NUM_SENSORDIGITIZATIONTAPS
 - Camera Enumerations, [98](#)
- NUM_SENSORSHUTTERMODE
 - Camera Enumerations, [98](#)
- NUM_SENSORTAPS
 - Camera Enumerations, [98](#)
- NUM_SEQUENCERCONFIGURATIONMODE
 - Camera Enumerations, [99](#)
- NUM_SEQUENCERCONFIGURATIONVALID
 - Camera Enumerations, [99](#)
- NUM_SEQUENCERMODE
 - Camera Enumerations, [99](#)
- NUM_SEQUENCERSETVALID
 - Camera Enumerations, [99](#)
- NUM_SEQUENCERTRIGGERACTIVATION
 - Camera Enumerations, [99](#)
- NUM_SEQUENCERTRIGGERSOURCE
 - Camera Enumerations, [100](#)
- NUM_SERIALPORTBAUDRATE
 - Camera Enumerations, [100](#)
- NUM_SERIALPORTPARITY
 - Camera Enumerations, [100](#)
- NUM_SERIALPORTSELECTOR
 - Camera Enumerations, [101](#)
- NUM_SERIALPORTSOURCE
 - Camera Enumerations, [101](#)
- NUM_SERIALPORTSTOPBITS
 - Camera Enumerations, [101](#)
- NUM_SOFTWARESIGNALSELECTOR
 - Camera Enumerations, [101](#)
- NUM_SOURCESELECTOR
 - Camera Enumerations, [102](#)
- NUM_STATISTICS_CHANNELS
 - Spinnaker C Enumerations, [213](#)
- NUM_TESTPATTERNGENERATORSELECTOR
 - Camera Enumerations, [102](#)
- NUM_TESTPATTERN
 - Camera Enumerations, [102](#)
- NUM_TIMERSELECTOR
 - Camera Enumerations, [102](#)
- NUM_TIMERSTATUS

- Camera Enumerations, [103](#)
- NUM_TIMERTRIGGERACTIVATION
 - Camera Enumerations, [103](#)
- NUM_TIMERTRIGGERSOURCE
 - Camera Enumerations, [104](#)
- NUM_TRANSFERCOMPONENTSELECTOR
 - Camera Enumerations, [105](#)
- NUM_TRANSFERCONTROLMODE
 - Camera Enumerations, [105](#)
- NUM_TRANSFEROPERATIONMODE
 - Camera Enumerations, [105](#)
- NUM_TRANSFERQUEUEMODE
 - Camera Enumerations, [105](#)
- NUM_TRANSFERSELECTOR
 - Camera Enumerations, [106](#)
- NUM_TRANSFERSTATUSSELECTOR
 - Camera Enumerations, [106](#)
- NUM_TRANSFERTRIGGERACTIVATION
 - Camera Enumerations, [106](#)
- NUM_TRANSFERTRIGGERMODE
 - Camera Enumerations, [107](#)
- NUM_TRANSFERTRIGGERSELECTOR
 - Camera Enumerations, [107](#)
- NUM_TRANSFERTRIGGERSOURCE
 - Camera Enumerations, [108](#)
- NUM_TRIGGERACTIVATION
 - Camera Enumerations, [108](#)
- NUM_TRIGGERMODE
 - Camera Enumerations, [108](#)
- NUM_TRIGGEROVERLAP
 - Camera Enumerations, [109](#)
- NUM_TRIGGERSELECTOR
 - Camera Enumerations, [109](#)
- NUM_TRIGGERSOURCE
 - Camera Enumerations, [109](#)
- NUM_USEROUTPUTSELECTOR
 - Camera Enumerations, [110](#)
- NUM_USERSETDEFAULT
 - Camera Enumerations, [110](#)
- NUM_USERSETSELECTOR
 - Camera Enumerations, [110](#)
- NUM_WHITECLIPSELECTOR
 - Camera Enumerations, [110](#)
- NUMDEVICEACCESSSTATUS
 - Transport Layer Enumerations, [273](#)
- NUMDEVICECURRENTSPEED
 - Transport Layer Enumerations, [274](#)
- NUMDEVICEENDIANESSMECHANISM
 - Transport Layer Enumerations, [274](#)
- NUMDEVICETYPE
 - Transport Layer Enumerations, [274](#)
- NUMGENICAMXMLLOCATION
 - Transport Layer Enumerations, [275](#)
- NUMGEVCCP
 - Transport Layer Enumerations, [275](#)
- NUMGUIXMLLOCATION
 - Transport Layer Enumerations, [275](#)
- NUMPOESTATUS
 - Transport Layer Enumerations, [275](#)
- NUMSTREAMBUFFERCOUNTMODE
 - Transport Layer Enumerations, [276](#)
- NUMSTREAMBUFFERHANDLINGMODE
 - Transport Layer Enumerations, [276](#)
- NUMSTREAMDEFAULTBUFFERCOUNTMODE
 - Transport Layer Enumerations, [276](#)
- NUMSTREAMTYPE
 - Transport Layer Enumerations, [277](#)
- NA
 - Spinnaker C GenICam Enumerations, [264](#)
- NI
 - Spinnaker C GenICam Enumerations, [264](#)
- No
 - Spinnaker C GenICam Enumerations, [269](#)
- NoCache
 - Spinnaker C GenICam Enumerations, [265](#)
- noIncrement
 - Spinnaker C GenICam Enumerations, [265](#)
- Node Access, [221](#)
 - [spinNodeDeregisterCallback](#), [222](#)
 - [spinNodeGetAccessMode](#), [222](#)
 - [spinNodeGetCachingMode](#), [223](#)
 - [spinNodeGetDescription](#), [223](#)
 - [spinNodeGetDisplayName](#), [223](#)
 - [spinNodeGetImposedAccessMode](#), [224](#)
 - [spinNodeGetImposedVisibility](#), [224](#)
 - [spinNodeGetName](#), [225](#)
 - [spinNodeGetNameSpace](#), [225](#)
 - [spinNodeGetPollingTime](#), [225](#)
 - [spinNodeGetToolTip](#), [226](#)
 - [spinNodeGetType](#), [226](#)
 - [spinNodeGetVisibility](#), [227](#)
 - [spinNodeInvalidateNode](#), [227](#)
 - [spinNodeIsAvailable](#), [227](#)
 - [spinNodeIsEqual](#), [228](#)
 - [spinNodeIsImplemented](#), [228](#)
 - [spinNodeIsReadable](#), [229](#)
 - [spinNodeIsWritable](#), [229](#)
 - [spinNodeRegisterCallback](#), [229](#)
- Node Map Access, [218](#)
 - [spinNodeMapGetNode](#), [218](#)
 - [spinNodeMapGetNodeByIndex](#), [219](#)
 - [spinNodeMapGetNumNodes](#), [219](#)
 - [spinNodeMapPoll](#), [219](#)
- None
 - Spinnaker C GenICam Enumerations, [268](#)
- OffsetX
 - [quickSpin](#), [317](#)
- OffsetY
 - [quickSpin](#), [317](#)
- PACKBITS
 - Spinnaker C Structures, [215](#)
- PAYLOAD_TYPE_CHUNK_DATA
 - Spinnaker C Enumerations, [212](#)
- PAYLOAD_TYPE_CHUNK_ONLY
 - Spinnaker C Enumerations, [212](#)

- PAYLOAD_TYPE_CUSTOM_ID
 - Spinnaker C Enumerations, [212](#)
- PAYLOAD_TYPE_DEVICE_SPECIFIC
 - Spinnaker C Enumerations, [212](#)
- PAYLOAD_TYPE_EXTENDED_CHUNK
 - Spinnaker C Enumerations, [212](#)
- PAYLOAD_TYPE_FILE
 - Spinnaker C Enumerations, [212](#)
- PAYLOAD_TYPE_H264
 - Spinnaker C Enumerations, [212](#)
- PAYLOAD_TYPE_IMAGE
 - Spinnaker C Enumerations, [212](#)
- PAYLOAD_TYPE_JPEG2000
 - Spinnaker C Enumerations, [212](#)
- PAYLOAD_TYPE_JPEG
 - Spinnaker C Enumerations, [212](#)
- PAYLOAD_TYPE_MULTI_PART
 - Spinnaker C Enumerations, [212](#)
- PAYLOAD_TYPE_RAW_DATA
 - Spinnaker C Enumerations, [212](#)
- PAYLOAD_TYPE_UNKNOWN
 - Spinnaker C Enumerations, [212](#)
- PGM
 - Spinnaker C Enumerations, [211](#)
- PNG
 - Spinnaker C Enumerations, [211](#)
- POEStatus
 - quickSpinTLInterface, [328](#)
- POEStatus_NotSupported
 - Transport Layer Enumerations, [275](#)
- POEStatus_PowerOff
 - Transport Layer Enumerations, [275](#)
- POEStatus_PowerOn
 - Transport Layer Enumerations, [275](#)
- PPM
 - Spinnaker C Enumerations, [211](#)
- PacketResendRequestCount
 - quickSpin, [317](#)
- PayloadSize
 - quickSpin, [317](#)
- PixelColorFilter
 - quickSpin, [317](#)
- PixelColorFilter_BayerBG
 - Camera Enumerations, [83](#)
- PixelColorFilter_BayerGB
 - Camera Enumerations, [83](#)
- PixelColorFilter_BayerGR
 - Camera Enumerations, [83](#)
- PixelColorFilter_BayerRG
 - Camera Enumerations, [83](#)
- PixelColorFilter_None
 - Camera Enumerations, [83](#)
- PixelDynamicRangeMax
 - quickSpin, [317](#)
- PixelDynamicRangeMin
 - quickSpin, [317](#)
- PixelFormat
 - quickSpin, [317](#)
- PixelFormat_B10
 - Camera Enumerations, [85](#)
- PixelFormat_B12
 - Camera Enumerations, [85](#)
- PixelFormat_B12_Jpeg
 - Camera Enumerations, [88](#)
- PixelFormat_B16
 - Camera Enumerations, [85](#)
- PixelFormat_B8
 - Camera Enumerations, [85](#)
- PixelFormat_BGR10
 - Camera Enumerations, [85](#)
- PixelFormat_BGR10p
 - Camera Enumerations, [85](#)
- PixelFormat_BGR12
 - Camera Enumerations, [85](#)
- PixelFormat_BGR12p
 - Camera Enumerations, [85](#)
- PixelFormat_BGR14
 - Camera Enumerations, [85](#)
- PixelFormat_BGR16
 - Camera Enumerations, [85](#)
- PixelFormat_BGR565p
 - Camera Enumerations, [85](#)
- PixelFormat_BGR8
 - Camera Enumerations, [84](#)
- PixelFormat_BGRa10
 - Camera Enumerations, [85](#)
- PixelFormat_BGRa10p
 - Camera Enumerations, [85](#)
- PixelFormat_BGRa12
 - Camera Enumerations, [85](#)
- PixelFormat_BGRa12p
 - Camera Enumerations, [85](#)
- PixelFormat_BGRa14
 - Camera Enumerations, [85](#)
- PixelFormat_BGRa16
 - Camera Enumerations, [85](#)
- PixelFormat_BGRa8
 - Camera Enumerations, [84](#)
- PixelFormat_BayerBG10
 - Camera Enumerations, [84](#)
- PixelFormat_BayerBG10Packed
 - Camera Enumerations, [84](#)
- PixelFormat_BayerBG10p
 - Camera Enumerations, [84](#)
- PixelFormat_BayerBG12
 - Camera Enumerations, [84](#)
- PixelFormat_BayerBG12Packed
 - Camera Enumerations, [83](#)
- PixelFormat_BayerBG12p
 - Camera Enumerations, [84](#)
- PixelFormat_BayerBG16
 - Camera Enumerations, [83](#)
- PixelFormat_BayerBG8
 - Camera Enumerations, [83](#)
- PixelFormat_BayerGB10
 - Camera Enumerations, [84](#)

- PixelFormat_BayerGB10Packed
 - Camera Enumerations, [84](#)
- PixelFormat_BayerGB10p
 - Camera Enumerations, [84](#)
- PixelFormat_BayerGB12
 - Camera Enumerations, [84](#)
- PixelFormat_BayerGB12Packed
 - Camera Enumerations, [83](#)
- PixelFormat_BayerGB12p
 - Camera Enumerations, [84](#)
- PixelFormat_BayerGB16
 - Camera Enumerations, [83](#)
- PixelFormat_BayerGB8
 - Camera Enumerations, [83](#)
- PixelFormat_BayerGR10
 - Camera Enumerations, [84](#)
- PixelFormat_BayerGR10Packed
 - Camera Enumerations, [84](#)
- PixelFormat_BayerGR10p
 - Camera Enumerations, [84](#)
- PixelFormat_BayerGR12
 - Camera Enumerations, [84](#)
- PixelFormat_BayerGR12Packed
 - Camera Enumerations, [83](#)
- PixelFormat_BayerGR12p
 - Camera Enumerations, [84](#)
- PixelFormat_BayerGR16
 - Camera Enumerations, [83](#)
- PixelFormat_BayerGR8
 - Camera Enumerations, [83](#)
- PixelFormat_BayerRG10
 - Camera Enumerations, [84](#)
- PixelFormat_BayerRG10Packed
 - Camera Enumerations, [84](#)
- PixelFormat_BayerRG10p
 - Camera Enumerations, [84](#)
- PixelFormat_BayerRG12
 - Camera Enumerations, [84](#)
- PixelFormat_BayerRG12Packed
 - Camera Enumerations, [83](#)
- PixelFormat_BayerRG12p
 - Camera Enumerations, [84](#)
- PixelFormat_BayerRG16
 - Camera Enumerations, [83](#)
- PixelFormat_BayerRG8
 - Camera Enumerations, [83](#)
- PixelFormat_BayerRGPolarized10p
 - Camera Enumerations, [88](#)
- PixelFormat_BayerRGPolarized12p
 - Camera Enumerations, [88](#)
- PixelFormat_BayerRGPolarized16
 - Camera Enumerations, [88](#)
- PixelFormat_BayerRGPolarized8
 - Camera Enumerations, [88](#)
- PixelFormat_BiColorBGRG10
 - Camera Enumerations, [86](#)
- PixelFormat_BiColorBGRG10p
 - Camera Enumerations, [86](#)
- PixelFormat_BiColorBGRG12
 - Camera Enumerations, [86](#)
- PixelFormat_BiColorBGRG12p
 - Camera Enumerations, [86](#)
- PixelFormat_BiColorBGRG8
 - Camera Enumerations, [86](#)
- PixelFormat_BiColorRGBG10
 - Camera Enumerations, [86](#)
- PixelFormat_BiColorRGBG10p
 - Camera Enumerations, [86](#)
- PixelFormat_BiColorRGBG12
 - Camera Enumerations, [86](#)
- PixelFormat_BiColorRGBG12p
 - Camera Enumerations, [86](#)
- PixelFormat_BiColorRGBG8
 - Camera Enumerations, [86](#)
- PixelFormat_Confidence1
 - Camera Enumerations, [86](#)
- PixelFormat_Confidence16
 - Camera Enumerations, [86](#)
- PixelFormat_Confidence1p
 - Camera Enumerations, [86](#)
- PixelFormat_Confidence32f
 - Camera Enumerations, [86](#)
- PixelFormat_Confidence8
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_A10p
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_A12p
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_A16
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_A32f
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_A8
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_ABC10p
 - Camera Enumerations, [85](#)
- PixelFormat_Coord3D_ABC10p_Planar
 - Camera Enumerations, [85](#)
- PixelFormat_Coord3D_ABC12p
 - Camera Enumerations, [85](#)
- PixelFormat_Coord3D_ABC12p_Planar
 - Camera Enumerations, [85](#)
- PixelFormat_Coord3D_ABC16
 - Camera Enumerations, [85](#)
- PixelFormat_Coord3D_ABC16_Planar
 - Camera Enumerations, [85](#)
- PixelFormat_Coord3D_ABC32f
 - Camera Enumerations, [85](#)
- PixelFormat_Coord3D_ABC32f_Planar
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_ABC8
 - Camera Enumerations, [85](#)
- PixelFormat_Coord3D_ABC8_Planar
 - Camera Enumerations, [85](#)
- PixelFormat_Coord3D_AC10p
 - Camera Enumerations, [86](#)

- PixelFormat_Coord3D_AC10p_Planar
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_AC12p
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_AC12p_Planar
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_AC16
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_AC16_Planar
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_AC32f
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_AC32f_Planar
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_AC8
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_AC8_Planar
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_B10p
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_B12p
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_B16
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_B32f
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_B8
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_C10p
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_C12p
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_C16
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_C32f
 - Camera Enumerations, [86](#)
- PixelFormat_Coord3D_C8
 - Camera Enumerations, [86](#)
- PixelFormat_G10
 - Camera Enumerations, [85](#)
- PixelFormat_G12
 - Camera Enumerations, [85](#)
- PixelFormat_G16
 - Camera Enumerations, [85](#)
- PixelFormat_G8
 - Camera Enumerations, [85](#)
- PixelFormat_GB12_Jpeg
 - Camera Enumerations, [88](#)
- PixelFormat_GR12_Jpeg
 - Camera Enumerations, [88](#)
- PixelFormat_Mono10
 - Camera Enumerations, [84](#)
- PixelFormat_Mono10Packed
 - Camera Enumerations, [84](#)
- PixelFormat_Mono10p
 - Camera Enumerations, [84](#)
- PixelFormat_Mono12
 - Camera Enumerations, [84](#)
- PixelFormat_Mono12Packed
 - Camera Enumerations, [83](#)
- PixelFormat_Mono12p
 - Camera Enumerations, [84](#)
- PixelFormat_Mono14
 - Camera Enumerations, [84](#)
- PixelFormat_Mono16
 - Camera Enumerations, [83](#)
- PixelFormat_Mono1p
 - Camera Enumerations, [84](#)
- PixelFormat_Mono2p
 - Camera Enumerations, [84](#)
- PixelFormat_Mono4p
 - Camera Enumerations, [84](#)
- PixelFormat_Mono8
 - Camera Enumerations, [83](#)
- PixelFormat_Mono8s
 - Camera Enumerations, [84](#)
- PixelFormat_Polarized10p
 - Camera Enumerations, [88](#)
- PixelFormat_Polarized12p
 - Camera Enumerations, [88](#)
- PixelFormat_Polarized16
 - Camera Enumerations, [88](#)
- PixelFormat_Polarized8
 - Camera Enumerations, [88](#)
- PixelFormat_R10
 - Camera Enumerations, [85](#)
- PixelFormat_R12
 - Camera Enumerations, [85](#)
- PixelFormat_R12_Jpeg
 - Camera Enumerations, [88](#)
- PixelFormat_R16
 - Camera Enumerations, [85](#)
- PixelFormat_R8
 - Camera Enumerations, [85](#)
- PixelFormat_RGB10
 - Camera Enumerations, [85](#)
- PixelFormat_RGB10_Planar
 - Camera Enumerations, [85](#)
- PixelFormat_RGB10p
 - Camera Enumerations, [85](#)
- PixelFormat_RGB10p32
 - Camera Enumerations, [85](#)
- PixelFormat_RGB12
 - Camera Enumerations, [85](#)
- PixelFormat_RGB12_Planar
 - Camera Enumerations, [85](#)
- PixelFormat_RGB12p
 - Camera Enumerations, [85](#)
- PixelFormat_RGB14
 - Camera Enumerations, [85](#)
- PixelFormat_RGB16
 - Camera Enumerations, [85](#)
- PixelFormat_RGB16_Planar
 - Camera Enumerations, [85](#)
- PixelFormat_RGB565p
 - Camera Enumerations, [85](#)

- PixelFormat_RGB8
 - Camera Enumerations, [84](#)
- PixelFormat_RGB8_Planar
 - Camera Enumerations, [84](#)
- PixelFormat_RGB8Packed
 - Camera Enumerations, [83](#)
- PixelFormat_RGBa10
 - Camera Enumerations, [84](#)
- PixelFormat_RGBa10p
 - Camera Enumerations, [84](#)
- PixelFormat_RGBa12
 - Camera Enumerations, [84](#)
- PixelFormat_RGBa12p
 - Camera Enumerations, [84](#)
- PixelFormat_RGBa14
 - Camera Enumerations, [84](#)
- PixelFormat_RGBa16
 - Camera Enumerations, [84](#)
- PixelFormat_RGBa8
 - Camera Enumerations, [84](#)
- PixelFormat_Raw16
 - Camera Enumerations, [88](#)
- PixelFormat_Raw8
 - Camera Enumerations, [88](#)
- PixelFormat_SCF1WBWG10
 - Camera Enumerations, [86](#)
- PixelFormat_SCF1WBWG10p
 - Camera Enumerations, [86](#)
- PixelFormat_SCF1WBWG12
 - Camera Enumerations, [86](#)
- PixelFormat_SCF1WBWG12p
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WBWG14
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WBWG16
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WBWG8
 - Camera Enumerations, [86](#)
- PixelFormat_SCF1WGWB10
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WGWB10p
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WGWB12
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WGWB12p
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WGWB14
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WGWB16
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WGWB8
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WGWR10
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WGWR10p
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WGWR12
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WGWR12p
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WGWR14
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WGWR16
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WGWR8
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WRWG10
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WRWG10p
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WRWG12
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WRWG12p
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WRWG14
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WRWG16
 - Camera Enumerations, [87](#)
- PixelFormat_SCF1WRWG8
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr10_CbYCr
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr10p_CbYCr
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr12_CbYCr
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr12p_CbYCr
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr411_8
 - Camera Enumerations, [84](#)
- PixelFormat_YCbCr411_8_CbYYCrYY
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr422_10
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr422_10_CbYCrY
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr422_10p
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr422_10p_CbYCrY
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr422_12
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr422_12_CbYCrY
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr422_12p
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr422_12p_CbYCrY
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr422_8
 - Camera Enumerations, [84](#)
- PixelFormat_YCbCr422_8_CbYCrY
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr601_10_CbYCr
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr601_10p_CbYCr
 - Camera Enumerations, [87](#)

- PixelFormat_YCbCr601_12_CbYCr
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr601_12p_CbYCr
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr601_411_8_CbYYCrYY
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr601_422_10
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr601_422_10_CbYCrY
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr601_422_10p
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr601_422_10p_CbYCrY
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr601_422_12
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr601_422_12_CbYCrY
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr601_422_12p
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr601_422_12p_CbYCrY
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr601_422_8
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr601_422_8_CbYCrY
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr601_8_CbYCr
 - Camera Enumerations, [87](#)
- PixelFormat_YCbCr709_10_CbYCr
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_10p_CbYCr
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_12_CbYCr
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_12p_CbYCr
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_411_8_CbYYCrYY
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_422_10
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_422_10_CbYCrY
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_422_10p
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_422_10p_CbYCrY
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_422_12
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_422_12_CbYCrY
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_422_12p
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_422_12p_CbYCrY
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_422_8
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_422_8_CbYCrY
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr709_8_CbYCr
 - Camera Enumerations, [88](#)
- PixelFormat_YCbCr8
 - Camera Enumerations, [84](#)
- PixelFormat_YCbCr8_CbYCr
 - Camera Enumerations, [87](#)
- PixelFormat_YUV411_8_UYYVYY
 - Camera Enumerations, [88](#)
- PixelFormat_YUV411Packed
 - Camera Enumerations, [83](#)
- PixelFormat_YUV422_8
 - Camera Enumerations, [88](#)
- PixelFormat_YUV422_8_UYVY
 - Camera Enumerations, [88](#)
- PixelFormat_YUV422Packed
 - Camera Enumerations, [83](#)
- PixelFormat_YUV444Packed
 - Camera Enumerations, [84](#)
- PixelFormat_YUV8_UYV
 - Camera Enumerations, [88](#)
- PixelFormatInfoD
 - quickSpin, [317](#)
- PixelFormatInfoSelector
 - quickSpin, [317](#)
- PixelFormatInfoSelector_B10
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_B12
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_B16
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_B8
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGR10
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGR10p
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGR12
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGR12p
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGR14
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGR16
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGR565p
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGR8
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGRa10
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGRa10p
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGRa12
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGRa12p
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGRa14
 - Camera Enumerations, [90](#)

- PixelFormatInfoSelector_BGRa16
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BGRa8
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_BayerBG10
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerBG10p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerBG12
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerBG12p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerBG16
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerBG8
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerGB10
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerGB10p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerGB12
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerGB12p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerGB16
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerGB8
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerGR10
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerGR10p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerGR12
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerGR12p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerGR16
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerGR8
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerRG10
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerRG10p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerRG12
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerRG12p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerRG16
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerRG8
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_BayerRGPolarized10p
 - Camera Enumerations, [94](#)
- PixelFormatInfoSelector_BayerRGPolarized12p
 - Camera Enumerations, [94](#)
- PixelFormatInfoSelector_BayerRGPolarized16
 - Camera Enumerations, [94](#)
- PixelFormatInfoSelector_BayerRGPolarized8
 - Camera Enumerations, [94](#)
- PixelFormatInfoSelector_BiColorBGRG10
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_BiColorBGRG10p
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_BiColorBGRG12
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_BiColorBGRG12p
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_BiColorBGRG8
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_BiColorRGBG10
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_BiColorRGBG10p
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_BiColorRGBG12
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_BiColorRGBG12p
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_BiColorRGBG8
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Confidence1
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Confidence16
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Confidence1p
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Confidence32f
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Confidence8
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_A10p
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_A12p
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_A16
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_A32f
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_A8
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_ABC10p
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_Coord3D_ABC10p_Planar
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_Coord3D_ABC12p
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_ABC12p_Planar
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_ABC16
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_ABC16_Planar
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_ABC32f
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_ABC32f_Planar
 - Camera Enumerations, [91](#)

- PixelFormatInfoSelector_Coord3D_ABC8
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_Coord3D_ABC8_Planar
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_Coord3D_AC10p
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_AC10p_Planar
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_AC12p
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_AC12p_Planar
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_AC16
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_AC16_Planar
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_AC32f
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_AC32f_Planar
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_AC8
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_AC8_Planar
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_B10p
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_B12p
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_B16
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_B32f
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_B8
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_C10p
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_C12p
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_C16
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_C32f
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_Coord3D_C8
 - Camera Enumerations, [91](#)
- PixelFormatInfoSelector_G10
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_G12
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_G16
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_G8
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_Mono10
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_Mono10p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_Mono12
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_Mono12p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_Mono14
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_Mono16
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_Mono1p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_Mono2p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_Mono4p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_Mono8
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_Mono8s
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_Polarized10p
 - Camera Enumerations, [94](#)
- PixelFormatInfoSelector_Polarized12p
 - Camera Enumerations, [94](#)
- PixelFormatInfoSelector_Polarized16
 - Camera Enumerations, [94](#)
- PixelFormatInfoSelector_Polarized8
 - Camera Enumerations, [94](#)
- PixelFormatInfoSelector_R10
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_R12
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_R16
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_R8
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGB10
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGB10_Planar
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGB10p
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGB10p32
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGB12
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGB12_Planar
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGB12p
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGB14
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGB16
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGB16_Planar
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGB565p
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGB8
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGB8_Planar
 - Camera Enumerations, [90](#)

- PixelFormatInfoSelector_RGBA10
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_RGBA10p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_RGBA12
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_RGBA12p
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_RGBA14
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_RGBA16
 - Camera Enumerations, [90](#)
- PixelFormatInfoSelector_RGBA8
 - Camera Enumerations, [89](#)
- PixelFormatInfoSelector_SCF1WBWG10
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_SCF1WBWG10p
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_SCF1WBWG12
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_SCF1WBWG12p
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_SCF1WBWG14
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_SCF1WBWG16
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_SCF1WBWG8
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_YCbCr10_CbYCr
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_YCbCr10p_CbYCr
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_YCbCr12_CbYCr
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_YCbCr12p_CbYCr
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_YCbCr411_8
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr411_8_CbYYCrYY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr422_10
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr422_10_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr422_10p
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr422_10p_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr422_12
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr422_12_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr422_12p
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr422_12p_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr422_8
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr422_8_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_10_CbYCr
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_10p_CbYCr
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_12_CbYCr
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_12p_CbYCr
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_411_8_CbYYCrYY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_422_10
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_422_10_CbYCrY
 - Camera Enumerations, [93](#)

- PixelFormatInfoSelector_YCbCr601_422_10p
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_422_10p_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_422_12
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_422_12_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_422_12p
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_422_12p_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_422_8
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_422_8_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr601_8_CbYCr
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_10_CbYCr
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_10p_CbYCr
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_12_CbYCr
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_12p_CbYCr
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_411_8_CbYYCrYY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_422_10
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_422_10_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_422_10p
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_422_10p_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_422_12
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_422_12_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_422_12p
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_422_12p_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_422_8
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_422_8_CbYCrY
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr709_8_CbYCr
 - Camera Enumerations, [93](#)
- PixelFormatInfoSelector_YCbCr8
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_YCbCr8_CbYCr
 - Camera Enumerations, [92](#)
- PixelFormatInfoSelector_YUV411_8_UYYVYY
 - Camera Enumerations, [94](#)
- PixelFormatInfoSelector_YUV422_8
 - Camera Enumerations, [94](#)
- PixelFormatInfoSelector_YUV422_8_UYVY
 - Camera Enumerations, [94](#)
- PixelFormatInfoSelector_YUV8_UYV
 - Camera Enumerations, [93](#)
- PixelSize
 - quickSpin, [317](#)
- PixelSize_Bpp1
 - Camera Enumerations, [94](#)
- PixelSize_Bpp10
 - Camera Enumerations, [94](#)
- PixelSize_Bpp12
 - Camera Enumerations, [94](#)
- PixelSize_Bpp14
 - Camera Enumerations, [94](#)
- PixelSize_Bpp16
 - Camera Enumerations, [94](#)
- PixelSize_Bpp2
 - Camera Enumerations, [94](#)
- PixelSize_Bpp20
 - Camera Enumerations, [94](#)
- PixelSize_Bpp24
 - Camera Enumerations, [94](#)
- PixelSize_Bpp30
 - Camera Enumerations, [94](#)
- PixelSize_Bpp32
 - Camera Enumerations, [94](#)
- PixelSize_Bpp36
 - Camera Enumerations, [94](#)
- PixelSize_Bpp4
 - Camera Enumerations, [94](#)
- PixelSize_Bpp48
 - Camera Enumerations, [94](#)
- PixelSize_Bpp64
 - Camera Enumerations, [94](#)
- PixelSize_Bpp8
 - Camera Enumerations, [94](#)
- PixelSize_Bpp96
 - Camera Enumerations, [94](#)
- PortNode
 - Spinnaker C GenICam Enumerations, [267](#)
- PowerSupplyCurrent
 - quickSpin, [317](#)
- PowerSupplyVoltage
 - quickSpin, [317](#)
- progressive
 - spinJPEGOption, [335](#)
- PureNumber
 - Spinnaker C GenICam Enumerations, [267](#)
- quality
 - spinJPEGOption, [335](#)
 - spinJPG2Option, [336](#)
 - spinMJPGOption, [338](#)
- quickSpin, [282](#)
 - aPAUSEMACCtrlFramesReceived, [295](#)
 - aPAUSEMACCtrlFramesTransmitted, [295](#)
 - AasRoiEnable, [294](#)
 - AasRoiHeight, [294](#)
 - AasRoiOffsetX, [294](#)

- AasRoiOffsetY, [294](#)
- AasRoiWidth, [294](#)
- AcquisitionAbort, [294](#)
- AcquisitionArm, [294](#)
- AcquisitionBurstFrameCount, [294](#)
- AcquisitionFrameCount, [294](#)
- AcquisitionFrameRate, [294](#)
- AcquisitionFrameRateEnable, [294](#)
- AcquisitionLineRate, [294](#)
- AcquisitionMode, [294](#)
- AcquisitionResultingFrameRate, [294](#)
- AcquisitionStart, [294](#)
- AcquisitionStatus, [294](#)
- AcquisitionStatusSelector, [295](#)
- AcquisitionStop, [295](#)
- ActionDeviceKey, [295](#)
- ActionGroupKey, [295](#)
- ActionGroupMask, [295](#)
- ActionQueueSize, [295](#)
- ActionSelector, [295](#)
- ActionUnconditionalMode, [295](#)
- AdaptiveCompressionEnable, [295](#)
- AdcBitDepth, [295](#)
- AutoAlgorithmSelector, [295](#)
- AutoExposureControlLoopDamping, [295](#)
- AutoExposureControlPriority, [295](#)
- AutoExposureEVCompensation, [295](#)
- AutoExposureExposureTimeLowerLimit, [295](#)
- AutoExposureExposureTimeUpperLimit, [295](#)
- AutoExposureGainLowerLimit, [295](#)
- AutoExposureGainUpperLimit, [295](#)
- AutoExposureGreyValueLowerLimit, [295](#)
- AutoExposureGreyValueUpperLimit, [295](#)
- AutoExposureLightingMode, [295](#)
- AutoExposureMeteringMode, [296](#)
- AutoExposureTargetGreyValue, [296](#)
- AutoExposureTargetGreyValueAuto, [296](#)
- BalanceRatio, [296](#)
- BalanceRatioSelector, [296](#)
- BalanceWhiteAuto, [296](#)
- BalanceWhiteAutoDamping, [296](#)
- BalanceWhiteAutoLowerLimit, [296](#)
- BalanceWhiteAutoProfile, [296](#)
- BalanceWhiteAutoUpperLimit, [296](#)
- BinningHorizontal, [296](#)
- BinningHorizontalMode, [296](#)
- BinningSelector, [296](#)
- BinningVertical, [296](#)
- BinningVerticalMode, [296](#)
- BlackLevel, [296](#)
- BlackLevelAuto, [296](#)
- BlackLevelAutoBalance, [296](#)
- BlackLevelClampingEnable, [296](#)
- BlackLevelRaw, [296](#)
- BlackLevelSelector, [296](#)
- ChunkBlackLevel, [296](#)
- ChunkBlackLevelSelector, [296](#)
- ChunkCRC, [297](#)
- ChunkCounterSelector, [297](#)
- ChunkCounterValue, [297](#)
- ChunkEnable, [297](#)
- ChunkEncoderSelector, [297](#)
- ChunkEncoderStatus, [297](#)
- ChunkEncoderValue, [297](#)
- ChunkExposureEndLineStatusAll, [297](#)
- ChunkExposureTime, [297](#)
- ChunkExposureTimeSelector, [297](#)
- ChunkFrameID, [297](#)
- ChunkGain, [297](#)
- ChunkGainSelector, [297](#)
- ChunkHeight, [297](#)
- ChunkImage, [297](#)
- ChunkImageComponent, [297](#)
- ChunkInferenceConfidence, [297](#)
- ChunkInferenceResult, [297](#)
- ChunkLinePitch, [297](#)
- ChunkLineStatusAll, [297](#)
- ChunkModeActive, [297](#)
- ChunkOffsetX, [297](#)
- ChunkOffsetY, [297](#)
- ChunkPartSelector, [298](#)
- ChunkPixelDynamicRangeMax, [298](#)
- ChunkPixelDynamicRangeMin, [298](#)
- ChunkPixelFormat, [298](#)
- ChunkRegionID, [298](#)
- ChunkScan3dAxisMax, [298](#)
- ChunkScan3dAxisMin, [298](#)
- ChunkScan3dCoordinateOffset, [298](#)
- ChunkScan3dCoordinateReferenceSelector, [298](#)
- ChunkScan3dCoordinateReferenceValue, [298](#)
- ChunkScan3dCoordinateScale, [298](#)
- ChunkScan3dCoordinateSelector, [298](#)
- ChunkScan3dCoordinateSystem, [298](#)
- ChunkScan3dCoordinateSystemReference, [298](#)
- ChunkScan3dCoordinateTransformSelector, [298](#)
- ChunkScan3dDistanceUnit, [298](#)
- ChunkScan3dInvalidDataFlag, [298](#)
- ChunkScan3dInvalidDataValue, [298](#)
- ChunkScan3dOutputMode, [298](#)
- ChunkScan3dTransformValue, [298](#)
- ChunkScanLineSelector, [298](#)
- ChunkSelector, [298](#)
- ChunkSequencerSetActive, [298](#)
- ChunkSerialData, [299](#)
- ChunkSerialDataLength, [299](#)
- ChunkSerialReceiveOverflow, [299](#)
- ChunkSourceID, [299](#)
- ChunkStreamChannelID, [299](#)
- ChunkTimerSelector, [299](#)
- ChunkTimerValue, [299](#)
- ChunkTimestamp, [299](#)
- ChunkTimestampLatchValue, [299](#)
- ChunkTransferBlockID, [299](#)
- ChunkTransferQueueCurrentBlockCount, [299](#)
- ChunkTransferStreamID, [299](#)
- ChunkWidth, [299](#)

- CIConfiguration, [299](#)
- CITimeSlotsCount, [299](#)
- ColorTransformationEnable, [299](#)
- ColorTransformationSelector, [299](#)
- ColorTransformationValue, [299](#)
- ColorTransformationValueSelector, [299](#)
- CompressionRatio, [299](#)
- CounterDelay, [299](#)
- CounterDuration, [299](#)
- CounterEventActivation, [299](#)
- CounterEventSource, [300](#)
- CounterReset, [300](#)
- CounterResetActivation, [300](#)
- CounterResetSource, [300](#)
- CounterSelector, [300](#)
- CounterStatus, [300](#)
- CounterTriggerActivation, [300](#)
- CounterTriggerSource, [300](#)
- CounterValue, [300](#)
- CounterValueAtReset, [300](#)
- CxpConnectionSelector, [300](#)
- CxpConnectionTestErrorCount, [300](#)
- CxpConnectionTestMode, [300](#)
- CxpConnectionTestPacketCount, [300](#)
- CxpLinkConfiguration, [300](#)
- CxpLinkConfigurationPreferred, [300](#)
- CxpLinkConfigurationStatus, [300](#)
- CxpPoCxpAuto, [300](#)
- CxpPoCxpStatus, [300](#)
- CxpPoCxpTripReset, [300](#)
- CxpPoCxpTurnOff, [300](#)
- DecimationHorizontal, [300](#)
- DecimationHorizontalMode, [300](#)
- DecimationSelector, [301](#)
- DecimationVertical, [301](#)
- DecimationVerticalMode, [301](#)
- DefectCorrectStaticEnable, [301](#)
- DefectCorrectionMode, [301](#)
- DefectTableApply, [301](#)
- DefectTableCoordinateX, [301](#)
- DefectTableCoordinateY, [301](#)
- DefectTableFactoryRestore, [301](#)
- DefectTableIndex, [301](#)
- DefectTablePixelCount, [301](#)
- DefectTableSave, [301](#)
- Deinterlacing, [301](#)
- DeviceCharacterSet, [301](#)
- DeviceClockFrequency, [301](#)
- DeviceClockSelector, [301](#)
- DeviceConnectionSelector, [301](#)
- DeviceConnectionSpeed, [301](#)
- DeviceConnectionStatus, [301](#)
- DeviceEventChannelCount, [301](#)
- DeviceFamilyName, [301](#)
- DeviceFeaturePersistenceEnd, [301](#)
- DeviceFeaturePersistenceStart, [301](#)
- DeviceFirmwareVersion, [302](#)
- DeviceGenCPVersionMajor, [302](#)
- DeviceGenCPVersionMinor, [302](#)
- DeviceID, [302](#)
- DeviceIndicatorMode, [302](#)
- DeviceLinkBandwidthReserve, [302](#)
- DeviceLinkCommandTimeout, [302](#)
- DeviceLinkConnectionCount, [302](#)
- DeviceLinkCurrentThroughput, [302](#)
- DeviceLinkHeartbeatMode, [302](#)
- DeviceLinkHeartbeatTimeout, [302](#)
- DeviceLinkSelector, [302](#)
- DeviceLinkSpeed, [302](#)
- DeviceLinkThroughputLimit, [302](#)
- DeviceLinkThroughputLimitMode, [302](#)
- DeviceManifestEntrySelector, [302](#)
- DeviceManifestPrimaryURL, [302](#)
- DeviceManifestSchemaMajorVersion, [302](#)
- DeviceManifestSchemaMinorVersion, [302](#)
- DeviceManifestSecondaryURL, [302](#)
- DeviceManifestXMLMajorVersion, [302](#)
- DeviceManifestXMLMinorVersion, [302](#)
- DeviceManifestXMLSubMinorVersion, [302](#)
- DeviceManufacturerInfo, [303](#)
- DeviceMaxThroughput, [303](#)
- DeviceModelName, [303](#)
- DevicePowerSupplySelector, [303](#)
- DeviceRegistersCheck, [303](#)
- DeviceRegistersEndianness, [303](#)
- DeviceRegistersStreamingEnd, [303](#)
- DeviceRegistersStreamingStart, [303](#)
- DeviceRegistersValid, [303](#)
- DeviceReset, [303](#)
- DeviceSFNCVersionMajor, [303](#)
- DeviceSFNCVersionMinor, [303](#)
- DeviceSFNCVersionSubMinor, [303](#)
- DeviceScanType, [303](#)
- DeviceSerialNumber, [303](#)
- DeviceSerialPortBaudRate, [303](#)
- DeviceSerialPortSelector, [303](#)
- DeviceStreamChannelCount, [303](#)
- DeviceStreamChannelEndianness, [303](#)
- DeviceStreamChannelLink, [303](#)
- DeviceStreamChannelPacketSize, [303](#)
- DeviceStreamChannelSelector, [303](#)
- DeviceStreamChannelType, [303](#)
- DeviceTLType, [304](#)
- DeviceTLVersionMajor, [304](#)
- DeviceTLVersionMinor, [304](#)
- DeviceTLVersionSubMinor, [304](#)
- DeviceTapGeometry, [304](#)
- DeviceTemperature, [304](#)
- DeviceTemperatureSelector, [304](#)
- DeviceType, [304](#)
- DeviceUptime, [304](#)
- DeviceUserID, [304](#)
- DeviceVendorName, [304](#)
- DeviceVersion, [304](#)
- EncoderDivider, [304](#)
- EncoderMode, [304](#)

EncoderOutputMode, 304
EncoderReset, 304
EncoderResetActivation, 304
EncoderResetSource, 304
EncoderSelector, 304
EncoderSourceA, 304
EncoderSourceB, 304
EncoderStatus, 304
EncoderTimeout, 304
EncoderValue, 305
EncoderValueAtReset, 305
EnumerationCount, 305
EventAcquisitionEnd, 305
EventAcquisitionEndFrameID, 305
EventAcquisitionEndTimestamp, 305
EventAcquisitionError, 305
EventAcquisitionErrorFrameID, 305
EventAcquisitionErrorTimestamp, 305
EventAcquisitionStart, 305
EventAcquisitionStartFrameID, 305
EventAcquisitionStartTimestamp, 305
EventAcquisitionTransferEnd, 305
EventAcquisitionTransferEndFrameID, 305
EventAcquisitionTransferEndTimestamp, 305
EventAcquisitionTransferStart, 305
EventAcquisitionTransferStartFrameID, 305
EventAcquisitionTransferStartTimestamp, 305
EventAcquisitionTrigger, 305
EventAcquisitionTriggerFrameID, 305
EventAcquisitionTriggerTimestamp, 305
EventActionLate, 305
EventActionLateFrameID, 305
EventActionLateTimestamp, 306
EventCounter0End, 306
EventCounter0EndFrameID, 306
EventCounter0EndTimestamp, 306
EventCounter0Start, 306
EventCounter0StartFrameID, 306
EventCounter0StartTimestamp, 306
EventCounter1End, 306
EventCounter1EndFrameID, 306
EventCounter1EndTimestamp, 306
EventCounter1Start, 306
EventCounter1StartFrameID, 306
EventCounter1StartTimestamp, 306
EventEncoder0Restarted, 306
EventEncoder0RestartedFrameID, 306
EventEncoder0RestartedTimestamp, 306
EventEncoder0Stopped, 306
EventEncoder0StoppedFrameID, 306
EventEncoder0StoppedTimestamp, 306
EventEncoder1Restarted, 306
EventEncoder1RestartedFrameID, 306
EventEncoder1RestartedTimestamp, 306
EventEncoder1Stopped, 306
EventEncoder1StoppedFrameID, 307
EventEncoder1StoppedTimestamp, 307
EventError, 307
EventErrorCode, 307
EventErrorFrameID, 307
EventErrorTimestamp, 307
EventExposureEnd, 307
EventExposureEndFrameID, 307
EventExposureEndTimestamp, 307
EventExposureStart, 307
EventExposureStartFrameID, 307
EventExposureStartTimestamp, 307
EventFrameBurstEnd, 307
EventFrameBurstEndFrameID, 307
EventFrameBurstEndTimestamp, 307
EventFrameBurstStart, 307
EventFrameBurstStartFrameID, 307
EventFrameBurstStartTimestamp, 307
EventFrameEnd, 307
EventFrameEndFrameID, 307
EventFrameEndTimestamp, 307
EventFrameStart, 307
EventFrameStartFrameID, 307
EventFrameStartTimestamp, 308
EventFrameTransferEnd, 308
EventFrameTransferEndFrameID, 308
EventFrameTransferEndTimestamp, 308
EventFrameTransferStart, 308
EventFrameTransferStartFrameID, 308
EventFrameTransferStartTimestamp, 308
EventFrameTrigger, 308
EventFrameTriggerFrameID, 308
EventFrameTriggerTimestamp, 308
EventLine0AnyEdge, 308
EventLine0AnyEdgeFrameID, 308
EventLine0AnyEdgeTimestamp, 308
EventLine0FallingEdge, 308
EventLine0FallingEdgeFrameID, 308
EventLine0FallingEdgeTimestamp, 308
EventLine0RisingEdge, 308
EventLine0RisingEdgeFrameID, 308
EventLine0RisingEdgeTimestamp, 308
EventLine1AnyEdge, 308
EventLine1AnyEdgeFrameID, 308
EventLine1AnyEdgeTimestamp, 308
EventLine1FallingEdge, 308
EventLine1FallingEdgeFrameID, 309
EventLine1FallingEdgeTimestamp, 309
EventLine1RisingEdge, 309
EventLine1RisingEdgeFrameID, 309
EventLine1RisingEdgeTimestamp, 309
EventLinkSpeedChange, 309
EventLinkSpeedChangeFrameID, 309
EventLinkSpeedChangeTimestamp, 309
EventLinkTrigger0, 309
EventLinkTrigger0FrameID, 309
EventLinkTrigger0Timestamp, 309
EventLinkTrigger1, 309
EventLinkTrigger1FrameID, 309
EventLinkTrigger1Timestamp, 309
EventNotification, 309

- EventSelector, [309](#)
- EventSequencerSetChange, [309](#)
- EventSequencerSetChangeFrameID, [309](#)
- EventSequencerSetChangeTimestamp, [309](#)
- EventSerialData, [309](#)
- EventSerialDataLength, [309](#)
- EventSerialPortReceive, [309](#)
- EventSerialPortReceiveTimestamp, [309](#)
- EventSerialReceiveOverflow, [310](#)
- EventStream0TransferBlockEnd, [310](#)
- EventStream0TransferBlockEndFrameID, [310](#)
- EventStream0TransferBlockEndTimestamp, [310](#)
- EventStream0TransferBlockStart, [310](#)
- EventStream0TransferBlockStartFrameID, [310](#)
- EventStream0TransferBlockStartTimestamp, [310](#)
- EventStream0TransferBlockTrigger, [310](#)
- EventStream0TransferBlockTriggerFrameID, [310](#)
- EventStream0TransferBlockTriggerTimestamp, [310](#)
- EventStream0TransferBurstEnd, [310](#)
- EventStream0TransferBurstEndFrameID, [310](#)
- EventStream0TransferBurstEndTimestamp, [310](#)
- EventStream0TransferBurstStart, [310](#)
- EventStream0TransferBurstStartFrameID, [310](#)
- EventStream0TransferBurstStartTimestamp, [310](#)
- EventStream0TransferEnd, [310](#)
- EventStream0TransferEndFrameID, [310](#)
- EventStream0TransferEndTimestamp, [310](#)
- EventStream0TransferOverflow, [310](#)
- EventStream0TransferOverflowFrameID, [310](#)
- EventStream0TransferOverflowTimestamp, [310](#)
- EventStream0TransferPause, [310](#)
- EventStream0TransferPauseFrameID, [311](#)
- EventStream0TransferPauseTimestamp, [311](#)
- EventStream0TransferResume, [311](#)
- EventStream0TransferResumeFrameID, [311](#)
- EventStream0TransferResumeTimestamp, [311](#)
- EventStream0TransferStart, [311](#)
- EventStream0TransferStartFrameID, [311](#)
- EventStream0TransferStartTimestamp, [311](#)
- EventTest, [311](#)
- EventTestTimestamp, [311](#)
- EventTimer0End, [311](#)
- EventTimer0EndFrameID, [311](#)
- EventTimer0EndTimestamp, [311](#)
- EventTimer0Start, [311](#)
- EventTimer0StartFrameID, [311](#)
- EventTimer0StartTimestamp, [311](#)
- EventTimer1End, [311](#)
- EventTimer1EndFrameID, [311](#)
- EventTimer1EndTimestamp, [311](#)
- EventTimer1Start, [311](#)
- EventTimer1StartFrameID, [311](#)
- EventTimer1StartTimestamp, [311](#)
- ExposureActiveMode, [311](#)
- ExposureAuto, [312](#)
- ExposureMode, [312](#)
- ExposureTime, [312](#)
- ExposureTimeMode, [312](#)
- ExposureTimeSelector, [312](#)
- FactoryReset, [312](#)
- FileAccessBuffer, [312](#)
- FileAccessLength, [312](#)
- FileAccessOffset, [312](#)
- FileOpenMode, [312](#)
- FileOperationExecute, [312](#)
- FileOperationResult, [312](#)
- FileOperationSelector, [312](#)
- FileOperationStatus, [312](#)
- FileSelector, [312](#)
- FileSize, [312](#)
- Gain, [312](#)
- GainAuto, [312](#)
- GainAutoBalance, [312](#)
- GainSelector, [312](#)
- Gamma, [312](#)
- GammaEnable, [312](#)
- GevActiveLinkCount, [312](#)
- GevCCP, [313](#)
- GevCurrentDefaultGateway, [313](#)
- GevCurrentIPAddress, [313](#)
- GevCurrentIPConfigurationDHCP, [313](#)
- GevCurrentIPConfigurationLLA, [313](#)
- GevCurrentIPConfigurationPersistentIP, [313](#)
- GevCurrentPhysicalLinkConfiguration, [313](#)
- GevCurrentSubnetMask, [313](#)
- GevDiscoveryAckDelay, [313](#)
- GevFirstURL, [313](#)
- GevGVCPExtendedStatusCodes, [313](#)
- GevGVCPExtendedStatusCodesSelector, [313](#)
- GevGVCPHeartbeatDisable, [313](#)
- GevGVCPPendingAck, [313](#)
- GevGVCPPendingTimeout, [313](#)
- GevGVSPExtendedIDMode, [313](#)
- GevHeartbeatTimeout, [313](#)
- GevIEEE1588, [313](#)
- GevIEEE1588ClockAccuracy, [313](#)
- GevIEEE1588Mode, [313](#)
- GevIEEE1588Status, [313](#)
- GevIPConfigurationStatus, [313](#)
- GevInterfaceSelector, [313](#)
- GevMACAddress, [314](#)
- GevMCDA, [314](#)
- GevMCPHostPort, [314](#)
- GevMCRC, [314](#)
- GevMCSP, [314](#)
- GevMCTT, [314](#)
- GevNumberOfInterfaces, [314](#)
- GevPAUSEFrameReception, [314](#)
- GevPAUSEFrameTransmission, [314](#)
- GevPersistentDefaultGateway, [314](#)
- GevPersistentIPAddress, [314](#)
- GevPersistentSubnetMask, [314](#)
- GevPhysicalLinkConfiguration, [314](#)
- GevPrimaryApplicationIPAddress, [314](#)
- GevPrimaryApplicationSocket, [314](#)
- GevPrimaryApplicationSwitchoverKey, [314](#)

- GevSCCFGAllInTransmission, [314](#)
- GevSCCFGExtendedChunkData, [314](#)
- GevSCCFGPacketResendDestination, [314](#)
- GevSCCFGUnconditionalStreaming, [314](#)
- GevSCDA, [314](#)
- GevSCPDirection, [314](#)
- GevSCPHostPort, [315](#)
- GevSCPInterfaceIndex, [315](#)
- GevSCPSBigEndian, [315](#)
- GevSCPSDoNotFragment, [315](#)
- GevSCPSFireTestPacket, [315](#)
- GevSCPSPacketSize, [315](#)
- GevSCPD, [314](#)
- GevSCSP, [315](#)
- GevSCZoneConfigurationLock, [315](#)
- GevSCZoneCount, [315](#)
- GevSCZoneDirectionAll, [315](#)
- GevSecondURL, [315](#)
- GevStreamChannelSelector, [315](#)
- GevSupportedOption, [315](#)
- GevSupportedOptionSelector, [315](#)
- GevTimestampTickFrequency, [315](#)
- GuiXmlManifestAddress, [315](#)
- Height, [315](#)
- HeightMax, [315](#)
- ImageComponentEnable, [315](#)
- ImageComponentSelector, [315](#)
- ImageCompressionBitrate, [315](#)
- ImageCompressionJPEGFormatOption, [315](#)
- ImageCompressionMode, [315](#)
- ImageCompressionQuality, [316](#)
- ImageCompressionRateOption, [316](#)
- IspEnable, [316](#)
- LUTEnable, [317](#)
- LUTIndex, [317](#)
- LUTSelector, [317](#)
- LUTValue, [317](#)
- LUTValueAll, [317](#)
- LineFilterWidth, [316](#)
- LineFormat, [316](#)
- LineInputFilterSelector, [316](#)
- LineInverter, [316](#)
- LineMode, [316](#)
- LinePitch, [316](#)
- LineSelector, [316](#)
- LineSource, [316](#)
- LineStatus, [316](#)
- LineStatusAll, [316](#)
- LinkErrorCount, [316](#)
- LinkUptime, [316](#)
- LogicBlockLUTInputActivation, [316](#)
- LogicBlockLUTInputSelector, [316](#)
- LogicBlockLUTInputSource, [316](#)
- LogicBlockLUTOutputValue, [316](#)
- LogicBlockLUTOutputValueAll, [316](#)
- LogicBlockLUTRowIndex, [316](#)
- LogicBlockLUTSelector, [316](#)
- LogicBlockSelector, [316](#)
- MaxDeviceResetTime, [317](#)
- OffsetX, [317](#)
- OffsetY, [317](#)
- PacketResendRequestCount, [317](#)
- PayloadSize, [317](#)
- PixelColorFilter, [317](#)
- PixelDynamicRangeMax, [317](#)
- PixelDynamicRangeMin, [317](#)
- PixelFormat, [317](#)
- PixelFormatInfoID, [317](#)
- PixelFormatInfoSelector, [317](#)
- PixelSize, [317](#)
- PowerSupplyCurrent, [317](#)
- PowerSupplyVoltage, [317](#)
- RegionDestination, [317](#)
- RegionMode, [317](#)
- RegionSelector, [317](#)
- ReverseX, [317](#)
- ReverseY, [318](#)
- RgbTransformLightSource, [318](#)
- Saturation, [318](#)
- SaturationEnable, [318](#)
- Scan3dAxisMax, [318](#)
- Scan3dAxisMin, [318](#)
- Scan3dCoordinateOffset, [318](#)
- Scan3dCoordinateReferenceSelector, [318](#)
- Scan3dCoordinateReferenceValue, [318](#)
- Scan3dCoordinateScale, [318](#)
- Scan3dCoordinateSelector, [318](#)
- Scan3dCoordinateSystem, [318](#)
- Scan3dCoordinateSystemReference, [318](#)
- Scan3dCoordinateTransformSelector, [318](#)
- Scan3dDistanceUnit, [318](#)
- Scan3dInvalidDataFlag, [318](#)
- Scan3dInvalidDataValue, [318](#)
- Scan3dOutputMode, [318](#)
- Scan3dTransformValue, [318](#)
- SensorDescription, [318](#)
- SensorDigitizationTaps, [318](#)
- SensorHeight, [318](#)
- SensorShutterMode, [318](#)
- SensorTaps, [319](#)
- SensorWidth, [319](#)
- SequencerConfigurationMode, [319](#)
- SequencerConfigurationValid, [319](#)
- SequencerFeatureEnable, [319](#)
- SequencerMode, [319](#)
- SequencerPathSelector, [319](#)
- SequencerSetActive, [319](#)
- SequencerSetLoad, [319](#)
- SequencerSetNext, [319](#)
- SequencerSetSave, [319](#)
- SequencerSetSelector, [319](#)
- SequencerSetStart, [319](#)
- SequencerSetValid, [319](#)
- SequencerTriggerActivation, [319](#)
- SequencerTriggerSource, [319](#)
- SerialPortBaudRate, [319](#)

- SerialPortDataBits, [319](#)
- SerialPortParity, [319](#)
- SerialPortSelector, [319](#)
- SerialPortSource, [319](#)
- SerialPortStopBits, [319](#)
- SerialReceiveFramingErrorCount, [319](#)
- SerialReceiveParityErrorCount, [320](#)
- SerialReceiveQueueClear, [320](#)
- SerialReceiveQueueCurrentCharacterCount, [320](#)
- SerialReceiveQueueMaxCharacterCount, [320](#)
- SerialTransmitQueueCurrentCharacterCount, [320](#)
- SerialTransmitQueueMaxCharacterCount, [320](#)
- Sharpening, [320](#)
- SharpeningAuto, [320](#)
- SharpeningEnable, [320](#)
- SharpeningThreshold, [320](#)
- SoftwareSignalPulse, [320](#)
- SoftwareSignalSelector, [320](#)
- SourceCount, [320](#)
- SourceSelector, [320](#)
- TLParamsLocked, [321](#)
- Test0001, [320](#)
- TestEventGenerate, [320](#)
- TestPattern, [320](#)
- TestPatternGeneratorSelector, [320](#)
- TestPendingAck, [320](#)
- TimerDelay, [320](#)
- TimerDuration, [320](#)
- TimerReset, [320](#)
- TimerSelector, [320](#)
- TimerStatus, [321](#)
- TimerTriggerActivation, [321](#)
- TimerTriggerSource, [321](#)
- TimerValue, [321](#)
- Timestamp, [321](#)
- TimestampLatch, [321](#)
- TimestampLatchValue, [321](#)
- TimestampReset, [321](#)
- TransferAbort, [321](#)
- TransferBlockCount, [321](#)
- TransferBurstCount, [321](#)
- TransferComponentSelector, [321](#)
- TransferControlMode, [321](#)
- TransferOperationMode, [321](#)
- TransferPause, [321](#)
- TransferQueueCurrentBlockCount, [321](#)
- TransferQueueMaxBlockCount, [321](#)
- TransferQueueMode, [321](#)
- TransferQueueOverflowCount, [321](#)
- TransferResume, [321](#)
- TransferSelector, [321](#)
- TransferStart, [321](#)
- TransferStatus, [322](#)
- TransferStatusSelector, [322](#)
- TransferStop, [322](#)
- TransferStreamChannel, [322](#)
- TransferTriggerActivation, [322](#)
- TransferTriggerMode, [322](#)
- TransferTriggerSelector, [322](#)
- TransferTriggerSource, [322](#)
- TriggerActivation, [322](#)
- TriggerDelay, [322](#)
- TriggerDivider, [322](#)
- TriggerEventTest, [322](#)
- TriggerMode, [322](#)
- TriggerMultiplier, [322](#)
- TriggerOverlap, [322](#)
- TriggerSelector, [322](#)
- TriggerSoftware, [322](#)
- TriggerSource, [322](#)
- UserOutputSelector, [322](#)
- UserOutputValue, [322](#)
- UserOutputValueAll, [322](#)
- UserOutputValueAllMask, [322](#)
- UserSetDefault, [322](#)
- UserSetFeatureEnable, [323](#)
- UserSetLoad, [323](#)
- UserSetSave, [323](#)
- UserSetSelector, [323](#)
- V3_3Enable, [323](#)
- WhiteClip, [323](#)
- WhiteClipSelector, [323](#)
- Width, [323](#)
- WidthMax, [323](#)
- QuickSpin Access, [113](#)
 - quickSpinInit, [113](#)
 - quickSpinInitEx, [113](#)
 - quickSpinTLDeviceInit, [113](#)
 - quickSpinTLInterfaceInit, [113](#)
 - quickSpinTLStreamInit, [113](#)
- quickSpinBooleanNode
 - QuickSpinDefsC.h, [376](#)
- quickSpinCommandNode
 - QuickSpinDefsC.h, [376](#)
- QuickSpinDefsC.h
 - quickSpinBooleanNode, [376](#)
 - quickSpinCommandNode, [376](#)
 - quickSpinEnumerationNode, [376](#)
 - quickSpinFloatNode, [376](#)
 - quickSpinIntegerNode, [376](#)
 - quickSpinRegisterNode, [376](#)
 - quickSpinStringNode, [376](#)
- quickSpinEnumerationNode
 - QuickSpinDefsC.h, [376](#)
- quickSpinFloatNode
 - QuickSpinDefsC.h, [376](#)
- quickSpinInit
 - QuickSpin Access, [113](#)
- quickSpinInitEx
 - QuickSpin Access, [113](#)
- quickSpinIntegerNode
 - QuickSpinDefsC.h, [376](#)
- quickSpinRegisterNode
 - QuickSpinDefsC.h, [376](#)
- quickSpinStringNode
 - QuickSpinDefsC.h, [376](#)

- quickSpinTLDevice, 323
 - DeviceAccessStatus, 324
 - DeviceCurrentSpeed, 324
 - DeviceDisplayName, 324
 - DeviceDriverVersion, 324
 - DeviceEndiannessMechanism, 324
 - DeviceID, 324
 - DeviceInstanceId, 324
 - DevicesUpdater, 324
 - DeviceLinkSpeed, 324
 - DeviceModelName, 324
 - DeviceMulticastMonitorMode, 324
 - DeviceSerialNumber, 324
 - DeviceType, 324
 - DeviceU3VProtocol, 324
 - DeviceUserID, 324
 - DeviceVendorName, 324
 - DeviceVersion, 325
 - GUXMLLocation, 325
 - GUXMLPath, 325
 - GenICamXMLLocation, 325
 - GenICamXMLPath, 325
 - GevCCP, 325
 - GevDeviceDiscoverMaximumPacketSize, 325
 - GevDeviceGateway, 325
 - GevDeviceIPAddress, 325
 - GevDevicesWrongSubnet, 325
 - GevDeviceMACAddress, 325
 - GevDeviceMaximumPacketSize, 325
 - GevDeviceMaximumRetryCount, 325
 - GevDeviceModelsBigEndian, 325
 - GevDevicePort, 325
 - GevDeviceReadAndWriteTimeout, 325
 - GevDeviceSubnetMask, 325
 - GevVersionMajor, 325
 - GevVersionMinor, 325
- quickSpinTLDeviceInit
 - QuickSpin Access, 113
- quickSpinTLInterface, 326
 - ActionCommand, 326
 - AutoForceIP, 326
 - DeviceAccessStatus, 326
 - DeviceCount, 326
 - DeviceID, 326
 - DeviceModelName, 327
 - DeviceSelector, 327
 - DeviceUnlock, 327
 - DeviceUpdateList, 327
 - DeviceVendorName, 327
 - GevActionDeviceKey, 327
 - GevActionGroupKey, 327
 - GevActionGroupMask, 327
 - GevActionTime, 327
 - GevDeviceIPAddress, 327
 - GevDeviceMACAddress, 327
 - GevDeviceSubnetMask, 327
 - GevInterfaceGateway, 327
 - GevInterfaceIPAddress, 327
 - GevInterfaceMACAddress, 327
 - GevInterfaceSubnetMask, 327
 - HostAdapterDriverVersion, 327
 - HostAdapterName, 327
 - HostAdapterVendor, 327
 - IncompatibleDeviceCount, 327
 - IncompatibleDeviceID, 327
 - IncompatibleDeviceModelName, 327
 - IncompatibleDeviceSelector, 327
 - IncompatibleDeviceVendorName, 328
 - IncompatibleGevDeviceIPAddress, 328
 - IncompatibleGevDeviceMACAddress, 328
 - IncompatibleGevDeviceSubnetMask, 328
 - InterfaceDisplayName, 328
 - InterfaceID, 328
 - InterfaceType, 328
 - POEStatus, 328
- quickSpinTLInterfaceInit
 - QuickSpin Access, 113
- quickSpinTLStream, 328
 - GevFailedPacketCount, 329
 - GevMaximumNumberResendBuffers, 329
 - GevMaximumNumberResendRequests, 329
 - GevPacketResendMode, 329
 - GevPacketResendTimeout, 329
 - GevResendPacketCount, 329
 - GevResendRequestCount, 329
 - GevTotalPacketCount, 329
 - StreamBlockTransferSize, 329
 - StreamBufferCountManual, 329
 - StreamBufferCountMax, 329
 - StreamBufferCountMode, 329
 - StreamBufferCountResult, 329
 - StreamBufferHandlingMode, 329
 - StreamBufferUnderrunCount, 329
 - StreamCRCCheckEnable, 329
 - StreamDefaultBufferCount, 329
 - StreamDefaultBufferCountMax, 329
 - StreamDefaultBufferCountMode, 329
 - StreamFailedBufferCount, 329
 - StreamID, 329
 - StreamTotalBufferCount, 329
 - StreamType, 329
- quickSpinTLStreamInit
 - QuickSpin Access, 113
- RAW
 - Spinnaker C Enumerations, 211
- RED
 - Spinnaker C Enumerations, 213
- RIGOROUS
 - Spinnaker C Enumerations, 209
- RegionDestination
 - quickSpin, 317
- RegionDestination_Stream0
 - Camera Enumerations, 94
- RegionDestination_Stream1
 - Camera Enumerations, 94
- RegionDestination_Stream2

- Camera Enumerations, [94](#)
- RegionMode
 - quickSpin, [317](#)
- RegionMode_Off
 - Camera Enumerations, [95](#)
- RegionMode_On
 - Camera Enumerations, [95](#)
- RegionSelector
 - quickSpin, [317](#)
- RegionSelector_All
 - Camera Enumerations, [95](#)
- RegionSelector_Region0
 - Camera Enumerations, [95](#)
- RegionSelector_Region1
 - Camera Enumerations, [95](#)
- RegionSelector_Region2
 - Camera Enumerations, [95](#)
- RegisterNode
 - Spinnaker C GenICam Enumerations, [267](#)
- reserved
 - spinAVIOption, [330](#)
 - spinBMPOption, [331](#)
 - spinH264Option, [334](#)
 - spinJPEGOption, [335](#)
 - spinJPG2Option, [336](#)
 - spinMJPGOption, [338](#)
 - spinPGMOption, [339](#)
 - spinPNGOption, [339](#)
 - spinPPMOption, [340](#)
 - spinTIFFOption, [341](#)
- ReverseX
 - quickSpin, [317](#)
- ReverseY
 - quickSpin, [318](#)
- RgbTransformLightSource
 - quickSpin, [318](#)
- RgbTransformLightSource_Cloudy6500K
 - Camera Enumerations, [95](#)
- RgbTransformLightSource_CoolFluorescent4000K
 - Camera Enumerations, [95](#)
- RgbTransformLightSource_Custom
 - Camera Enumerations, [95](#)
- RgbTransformLightSource_Daylight5000K
 - Camera Enumerations, [95](#)
- RgbTransformLightSource_General
 - Camera Enumerations, [95](#)
- RgbTransformLightSource_Shade8000K
 - Camera Enumerations, [95](#)
- RgbTransformLightSource_Tungsten2800K
 - Camera Enumerations, [95](#)
- RgbTransformLightSource_WarmFluorescent3000K
 - Camera Enumerations, [95](#)
- RO
 - Spinnaker C GenICam Enumerations, [264](#)
- RW
 - Spinnaker C GenICam Enumerations, [264](#)
- SATURATION
 - Spinnaker C Enumerations, [213](#)
- SPINNAKER_ERR_ABORT
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_ACCESS_DENIED
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_BUFFER_TOO_SMALL
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_BUSY
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_CUSTOM_ID
 - Spinnaker C Enumerations, [211](#)
- SPINNAKER_ERR_ERROR
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_IM_COLOR_CONVERSION
 - Spinnaker C Enumerations, [211](#)
- SPINNAKER_ERR_IM_CONVERT
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_IM_COPY
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_IM_HISTOGRAM_MEAN
 - Spinnaker C Enumerations, [211](#)
- SPINNAKER_ERR_IM_HISTOGRAM_RANGE
 - Spinnaker C Enumerations, [211](#)
- SPINNAKER_ERR_IM_MALLOC
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_IM_MIN_MAX
 - Spinnaker C Enumerations, [211](#)
- SPINNAKER_ERR_IM_NOT_SUPPORTED
 - Spinnaker C Enumerations, [211](#)
- SPINNAKER_ERR_INVALID_ADDRESS
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_INVALID_BUFFER
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_INVALID_HANDLE
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_INVALID_INDEX
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_INVALID_ID
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_INVALID_PARAMETER
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_INVALID_VALUE
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_IO
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_NO_DATA
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_NOT_AVAILABLE
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_NOT_IMPLEMENTED
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_NOT_INITIALIZED
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_OUT_OF_MEMORY
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_PARSING_CHUNK_DATA
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_RESOURCE_EXHAUSTED
 - Spinnaker C Enumerations, [210](#)

- SPINNAKER_ERR_RESOURCE_IN_USE
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_SUCCESS
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_ERR_TIMEOUT
 - Spinnaker C Enumerations, [210](#)
- SPINNAKER_PIXELFORMAT_NAMESPACE_CUST↔
 - OM_ID
 - Spinnaker C Enumerations, [212](#)
- SPINNAKER_PIXELFORMAT_NAMESPACE_GEV
 - Spinnaker C Enumerations, [212](#)
- SPINNAKER_PIXELFORMAT_NAMESPACE_IIDC
 - Spinnaker C Enumerations, [212](#)
- SPINNAKER_PIXELFORMAT_NAMESPACE_PFNC↔
 - _16BIT
 - Spinnaker C Enumerations, [212](#)
- SPINNAKER_PIXELFORMAT_NAMESPACE_PFNC↔
 - _32BIT
 - Spinnaker C Enumerations, [212](#)
- SPINNAKER_PIXELFORMAT_NAMESPACE_UNKN↔
 - OWN
 - Spinnaker C Enumerations, [212](#)
- SPINNAKER_API_DEPRECATED
 - AVIRecorder Access, [198](#), [199](#)
- SPINNAKER_API
 - SpinnakerPlatformC.h, [398](#)
- Saturation
 - quickSpin, [318](#)
- SaturationEnable
 - quickSpin, [318](#)
- Scan3dAxisMax
 - quickSpin, [318](#)
- Scan3dAxisMin
 - quickSpin, [318](#)
- Scan3dCoordinateOffset
 - quickSpin, [318](#)
- Scan3dCoordinateReferenceSelector
 - quickSpin, [318](#)
- Scan3dCoordinateReferenceSelector_RotationX
 - Camera Enumerations, [96](#)
- Scan3dCoordinateReferenceSelector_RotationY
 - Camera Enumerations, [96](#)
- Scan3dCoordinateReferenceSelector_RotationZ
 - Camera Enumerations, [96](#)
- Scan3dCoordinateReferenceSelector_TranslationX
 - Camera Enumerations, [96](#)
- Scan3dCoordinateReferenceSelector_TranslationY
 - Camera Enumerations, [96](#)
- Scan3dCoordinateReferenceSelector_TranslationZ
 - Camera Enumerations, [96](#)
- Scan3dCoordinateReferenceValue
 - quickSpin, [318](#)
- Scan3dCoordinateScale
 - quickSpin, [318](#)
- Scan3dCoordinateSelector
 - quickSpin, [318](#)
- Scan3dCoordinateSelector_CoordinateA
 - Camera Enumerations, [96](#)
- Scan3dCoordinateSelector_CoordinateB
 - Camera Enumerations, [96](#)
- Scan3dCoordinateSelector_CoordinateC
 - Camera Enumerations, [96](#)
- Scan3dCoordinateSystem
 - quickSpin, [318](#)
- Scan3dCoordinateSystem_Cartesian
 - Camera Enumerations, [96](#)
- Scan3dCoordinateSystem_Cylindrical
 - Camera Enumerations, [96](#)
- Scan3dCoordinateSystem_Spherical
 - Camera Enumerations, [96](#)
- Scan3dCoordinateSystemReference
 - quickSpin, [318](#)
- Scan3dCoordinateSystemReference_Anchor
 - Camera Enumerations, [96](#)
- Scan3dCoordinateSystemReference_Transformed
 - Camera Enumerations, [96](#)
- Scan3dCoordinateTransformSelector
 - quickSpin, [318](#)
- Scan3dCoordinateTransformSelector_RotationX
 - Camera Enumerations, [97](#)
- Scan3dCoordinateTransformSelector_RotationY
 - Camera Enumerations, [97](#)
- Scan3dCoordinateTransformSelector_RotationZ
 - Camera Enumerations, [97](#)
- Scan3dCoordinateTransformSelector_TranslationX
 - Camera Enumerations, [97](#)
- Scan3dCoordinateTransformSelector_TranslationY
 - Camera Enumerations, [97](#)
- Scan3dCoordinateTransformSelector_TranslationZ
 - Camera Enumerations, [97](#)
- Scan3dDistanceUnit
 - quickSpin, [318](#)
- Scan3dDistanceUnit_Inch
 - Camera Enumerations, [97](#)
- Scan3dDistanceUnit_Millimeter
 - Camera Enumerations, [97](#)
- Scan3dInvalidDataFlag
 - quickSpin, [318](#)
- Scan3dInvalidDataValue
 - quickSpin, [318](#)
- Scan3dOutputMode
 - quickSpin, [318](#)
- Scan3dOutputMode_CalibratedABC_Grid
 - Camera Enumerations, [97](#)
- Scan3dOutputMode_CalibratedABC_PointCloud
 - Camera Enumerations, [97](#)
- Scan3dOutputMode_CalibratedAC_Linescan
 - Camera Enumerations, [97](#)
- Scan3dOutputMode_CalibratedAC
 - Camera Enumerations, [97](#)
- Scan3dOutputMode_CalibratedC_Linescan
 - Camera Enumerations, [97](#)
- Scan3dOutputMode_CalibratedC
 - Camera Enumerations, [97](#)
- Scan3dOutputMode_DisparityC_Linescan
 - Camera Enumerations, [98](#)

- Scan3dOutputMode_DisparityC
 - Camera Enumerations, [98](#)
- Scan3dOutputMode_RectifiedC_Linescan
 - Camera Enumerations, [97](#)
- Scan3dOutputMode_RectifiedC
 - Camera Enumerations, [97](#)
- Scan3dOutputMode_UncalibratedC
 - Camera Enumerations, [97](#)
- Scan3dTransformValue
 - quickSpin, [318](#)
- SensorDescription
 - quickSpin, [318](#)
- SensorDigitizationTaps
 - quickSpin, [318](#)
- SensorDigitizationTaps_Eight
 - Camera Enumerations, [98](#)
- SensorDigitizationTaps_Four
 - Camera Enumerations, [98](#)
- SensorDigitizationTaps_One
 - Camera Enumerations, [98](#)
- SensorDigitizationTaps_Ten
 - Camera Enumerations, [98](#)
- SensorDigitizationTaps_Three
 - Camera Enumerations, [98](#)
- SensorDigitizationTaps_Two
 - Camera Enumerations, [98](#)
- SensorHeight
 - quickSpin, [318](#)
- SensorShutterMode
 - quickSpin, [318](#)
- SensorShutterMode_Global
 - Camera Enumerations, [98](#)
- SensorShutterMode_GlobalReset
 - Camera Enumerations, [98](#)
- SensorShutterMode_Rolling
 - Camera Enumerations, [98](#)
- SensorTaps
 - quickSpin, [319](#)
- SensorTaps_Eight
 - Camera Enumerations, [98](#)
- SensorTaps_Four
 - Camera Enumerations, [98](#)
- SensorTaps_One
 - Camera Enumerations, [98](#)
- SensorTaps_Ten
 - Camera Enumerations, [98](#)
- SensorTaps_Three
 - Camera Enumerations, [98](#)
- SensorTaps_Two
 - Camera Enumerations, [98](#)
- SensorWidth
 - quickSpin, [319](#)
- SequencerConfigurationMode
 - quickSpin, [319](#)
- SequencerConfigurationMode_Off
 - Camera Enumerations, [99](#)
- SequencerConfigurationMode_On
 - Camera Enumerations, [99](#)
- SequencerConfigurationValid
 - quickSpin, [319](#)
- SequencerConfigurationValid_No
 - Camera Enumerations, [99](#)
- SequencerConfigurationValid_Yes
 - Camera Enumerations, [99](#)
- SequencerFeatureEnable
 - quickSpin, [319](#)
- SequencerMode
 - quickSpin, [319](#)
- SequencerMode_Off
 - Camera Enumerations, [99](#)
- SequencerMode_On
 - Camera Enumerations, [99](#)
- SequencerPathSelector
 - quickSpin, [319](#)
- SequencerSetActive
 - quickSpin, [319](#)
- SequencerSetLoad
 - quickSpin, [319](#)
- SequencerSetNext
 - quickSpin, [319](#)
- SequencerSetSave
 - quickSpin, [319](#)
- SequencerSetSelector
 - quickSpin, [319](#)
- SequencerSetStart
 - quickSpin, [319](#)
- SequencerSetValid
 - quickSpin, [319](#)
- SequencerSetValid_No
 - Camera Enumerations, [99](#)
- SequencerSetValid_Yes
 - Camera Enumerations, [99](#)
- SequencerTriggerActivation
 - quickSpin, [319](#)
- SequencerTriggerActivation_AnyEdge
 - Camera Enumerations, [99](#)
- SequencerTriggerActivation_FallingEdge
 - Camera Enumerations, [99](#)
- SequencerTriggerActivation_LevelHigh
 - Camera Enumerations, [99](#)
- SequencerTriggerActivation_LevelLow
 - Camera Enumerations, [99](#)
- SequencerTriggerActivation_RisingEdge
 - Camera Enumerations, [99](#)
- SequencerTriggerSource
 - quickSpin, [319](#)
- SequencerTriggerSource_FrameStart
 - Camera Enumerations, [100](#)
- SequencerTriggerSource_Off
 - Camera Enumerations, [100](#)
- SerialPortBaudRate
 - quickSpin, [319](#)
- SerialPortBaudRate_Baud115200
 - Camera Enumerations, [100](#)
- SerialPortBaudRate_Baud1200
 - Camera Enumerations, [100](#)

- SerialPortBaudRate_Baud14400
 - Camera Enumerations, [100](#)
- SerialPortBaudRate_Baud19200
 - Camera Enumerations, [100](#)
- SerialPortBaudRate_Baud230400
 - Camera Enumerations, [100](#)
- SerialPortBaudRate_Baud2400
 - Camera Enumerations, [100](#)
- SerialPortBaudRate_Baud300
 - Camera Enumerations, [100](#)
- SerialPortBaudRate_Baud38400
 - Camera Enumerations, [100](#)
- SerialPortBaudRate_Baud460800
 - Camera Enumerations, [100](#)
- SerialPortBaudRate_Baud4800
 - Camera Enumerations, [100](#)
- SerialPortBaudRate_Baud57600
 - Camera Enumerations, [100](#)
- SerialPortBaudRate_Baud600
 - Camera Enumerations, [100](#)
- SerialPortBaudRate_Baud921600
 - Camera Enumerations, [100](#)
- SerialPortBaudRate_Baud9600
 - Camera Enumerations, [100](#)
- SerialPortDataBits
 - quickSpin, [319](#)
- SerialPortParity
 - quickSpin, [319](#)
- SerialPortParity_Even
 - Camera Enumerations, [100](#)
- SerialPortParity_Mark
 - Camera Enumerations, [100](#)
- SerialPortParity_None
 - Camera Enumerations, [100](#)
- SerialPortParity_Odd
 - Camera Enumerations, [100](#)
- SerialPortParity_Space
 - Camera Enumerations, [100](#)
- SerialPortSelector
 - quickSpin, [319](#)
- SerialPortSelector_SerialPort0
 - Camera Enumerations, [101](#)
- SerialPortSource
 - quickSpin, [319](#)
- SerialPortSource_Line0
 - Camera Enumerations, [101](#)
- SerialPortSource_Line1
 - Camera Enumerations, [101](#)
- SerialPortSource_Line2
 - Camera Enumerations, [101](#)
- SerialPortSource_Line3
 - Camera Enumerations, [101](#)
- SerialPortSource_Off
 - Camera Enumerations, [101](#)
- SerialPortStopBits
 - quickSpin, [319](#)
- SerialPortStopBits_Bits1
 - Camera Enumerations, [101](#)
- SerialPortStopBits_Bits1AndAHalf
 - Camera Enumerations, [101](#)
- SerialPortStopBits_Bits2
 - Camera Enumerations, [101](#)
- SerialReceiveFramingErrorCount
 - quickSpin, [319](#)
- SerialReceiveParityErrorCount
 - quickSpin, [320](#)
- SerialReceiveQueueClear
 - quickSpin, [320](#)
- SerialReceiveQueueCurrentCharacterCount
 - quickSpin, [320](#)
- SerialReceiveQueueMaxCharacterCount
 - quickSpin, [320](#)
- SerialTransmitQueueCurrentCharacterCount
 - quickSpin, [320](#)
- SerialTransmitQueueMaxCharacterCount
 - quickSpin, [320](#)
- Sharpening
 - quickSpin, [320](#)
- SharpeningAuto
 - quickSpin, [320](#)
- SharpeningEnable
 - quickSpin, [320](#)
- SharpeningThreshold
 - quickSpin, [320](#)
- Signed
 - Spinnaker C GenICam Enumerations, [268](#)
- SoftwareSignalPulse
 - quickSpin, [320](#)
- SoftwareSignalSelector
 - quickSpin, [320](#)
- SoftwareSignalSelector_SoftwareSignal0
 - Camera Enumerations, [101](#)
- SoftwareSignalSelector_SoftwareSignal1
 - Camera Enumerations, [101](#)
- SoftwareSignalSelector_SoftwareSignal2
 - Camera Enumerations, [101](#)
- SourceCount
 - quickSpin, [320](#)
- SourceSelector
 - quickSpin, [320](#)
- SourceSelector_All
 - Camera Enumerations, [102](#)
- SourceSelector_Source0
 - Camera Enumerations, [102](#)
- SourceSelector_Source1
 - Camera Enumerations, [102](#)
- SourceSelector_Source2
 - Camera Enumerations, [102](#)
- spinAVIOption, [330](#)
 - frameRate, [330](#)
 - reserved, [330](#)
- spinAVIRecorder
 - Spinnaker C Handles, [202](#)
- spinAccessMode
 - Spinnaker C GenICam Enumerations, [264](#)
- spinAcquisitionModeEnums

- Camera Enumerations, [43](#)
- spinAcquisitionStatusSelectorEnums
 - Camera Enumerations, [43](#)
- spinActionUnconditionalModeEnums
 - Camera Enumerations, [44](#)
- spinAdcBitDepthEnums
 - Camera Enumerations, [44](#)
- spinArrivalEvent
 - Spinnaker C Handles, [202](#)
- spinArrivalEventCreate
 - Event Access, [178](#)
- spinArrivalEventDestroy
 - Event Access, [179](#)
- spinArrivalEventFunction
 - Spinnaker C Function Signatures, [205](#)
- spinAutoAlgorithmSelectorEnums
 - Camera Enumerations, [44](#)
- spinAutoExposureControlPriorityEnums
 - Camera Enumerations, [44](#)
- spinAutoExposureLightingModeEnums
 - Camera Enumerations, [45](#)
- spinAutoExposureMeteringModeEnums
 - Camera Enumerations, [45](#)
- spinAutoExposureTargetGreyValueAutoEnums
 - Camera Enumerations, [45](#)
- spinBMPOption, [330](#)
 - indexedColor_8bit, [331](#)
 - reserved, [331](#)
- spinBalanceRatioSelectorEnums
 - Camera Enumerations, [46](#)
- spinBalanceWhiteAutoEnums
 - Camera Enumerations, [46](#)
- spinBalanceWhiteAutoProfileEnums
 - Camera Enumerations, [46](#)
- spinBinningHorizontalModeEnums
 - Camera Enumerations, [46](#)
- spinBinningSelectorEnums
 - Camera Enumerations, [47](#)
- spinBinningVerticalModeEnums
 - Camera Enumerations, [47](#)
- spinBlackLevelAutoBalanceEnums
 - Camera Enumerations, [47](#)
- spinBlackLevelAutoEnums
 - Camera Enumerations, [47](#)
- spinBlackLevelSelectorEnums
 - Camera Enumerations, [48](#)
- spinBooleanGetValue
 - IBoolean Access, [251](#)
- spinBooleanSetValue
 - IBoolean Access, [252](#)
- spinCachingMode
 - Spinnaker C GenICam Enumerations, [264](#)
- spinCamera
 - Spinnaker C Handles, [202](#)
- spinCameraBeginAcquisition
 - Camera Access, [146](#)
- spinCameraDeInit
 - Camera Access, [146](#)
- spinCameraDiscoverMaxPacketSize
 - Spinnaker C API, [115](#)
- spinCameraEndAcquisition
 - Camera Access, [147](#)
- spinCameraGetAccessMode
 - Camera Access, [147](#)
- spinCameraGetGuiXml
 - Camera Access, [147](#)
- spinCameraGetNextImage
 - Camera Access, [148](#)
- spinCameraGetNextImageEx
 - Camera Access, [148](#)
- spinCameraGetNodeMap
 - Camera Access, [149](#)
- spinCameraGetTLDeviceNodeMap
 - Camera Access, [149](#)
- spinCameraGetTLStreamNodeMap
 - Camera Access, [149](#)
- spinCameraGetUniqueID
 - Camera Access, [150](#)
- spinCameraInit
 - Camera Access, [150](#)
- spinCamerasInitialized
 - Camera Access, [151](#)
- spinCamerasStreaming
 - Camera Access, [151](#)
- spinCamerasValid
 - Camera Access, [151](#)
- spinCameraList
 - Spinnaker C Handles, [202](#)
- spinCameraListAppend
 - CameraList Access, [133](#)
- spinCameraListClear
 - CameraList Access, [134](#)
- spinCameraListCreateEmpty
 - CameraList Access, [134](#)
- spinCameraListDestroy
 - CameraList Access, [134](#)
- spinCameraListGet
 - CameraList Access, [135](#)
- spinCameraListGetBySerial
 - CameraList Access, [135](#)
- spinCameraListGetSize
 - CameraList Access, [136](#)
- spinCameraListRemove
 - CameraList Access, [136](#)
- spinCameraListRemoveBySerial
 - CameraList Access, [136](#)
- spinCameraReadPort
 - Camera Access, [152](#)
- spinCameraRegisterDeviceEvent
 - Camera Access, [152](#)
- spinCameraRegisterDeviceEventEx
 - Camera Access, [152](#)
- spinCameraRegisterImageEvent
 - Camera Access, [153](#)
- spinCameraRelease
 - Camera Access, [153](#)

- spinCameraUnregisterDeviceEvent
 - Camera Access, [153](#)
- spinCameraUnregisterImageEvent
 - Camera Access, [154](#)
- spinCameraWritePort
 - Camera Access, [154](#)
- spinCategoryGetFeatureByIndex
 - ICategory Access, [255](#)
- spinCategoryGetNumFeatures
 - ICategory Access, [256](#)
- spinChunkBlackLevelSelectorEnums
 - Camera Enumerations, [48](#)
- spinChunkCounterSelectorEnums
 - Camera Enumerations, [48](#)
- spinChunkData, [331](#)
 - m_blackLevel, [332](#)
 - m_cRC, [332](#)
 - m_counterValue, [332](#)
 - m_encoderValue, [332](#)
 - m_exposureEndLineStatusAll, [332](#)
 - m_exposureTime, [332](#)
 - m_frameID, [332](#)
 - m_gain, [332](#)
 - m_height, [332](#)
 - m_image, [332](#)
 - m_inferenceConfidence, [332](#)
 - m_inferenceResult, [332](#)
 - m_linePitch, [332](#)
 - m_lineStatusAll, [332](#)
 - m_offsetX, [332](#)
 - m_offsetY, [333](#)
 - m_partSelector, [333](#)
 - m_pixelDynamicRangeMax, [333](#)
 - m_pixelDynamicRangeMin, [333](#)
 - m_scan3dAxisMax, [333](#)
 - m_scan3dAxisMin, [333](#)
 - m_scan3dCoordinateOffset, [333](#)
 - m_scan3dCoordinateReferenceValue, [333](#)
 - m_scan3dCoordinateScale, [333](#)
 - m_scan3dInvalidDataValue, [333](#)
 - m_scan3dTransformValue, [333](#)
 - m_scanLineSelector, [333](#)
 - m_sequencerSetActive, [333](#)
 - m_serialDataLength, [333](#)
 - m_streamChannelID, [333](#)
 - m_timerValue, [333](#)
 - m_timestamp, [333](#)
 - m_timestampLatchValue, [333](#)
 - m_transferBlockID, [333](#)
 - m_transferQueueCurrentBlockCount, [333](#)
 - m_width, [333](#)
- spinChunkEncoderSelectorEnums
 - Camera Enumerations, [48](#)
- spinChunkEncoderStatusEnums
 - Camera Enumerations, [49](#)
- spinChunkExposureTimeSelectorEnums
 - Camera Enumerations, [49](#)
- spinChunkGainSelectorEnums
 - Camera Enumerations, [49](#)
- spinChunkImageComponentEnums
 - Camera Enumerations, [50](#)
- spinChunkPixelFormatEnums
 - Camera Enumerations, [50](#)
- spinChunkRegionIDEnums
 - Camera Enumerations, [50](#)
- spinChunkScan3dCoordinateReferenceSelectorEnums
 - Camera Enumerations, [51](#)
- spinChunkScan3dCoordinateSelectorEnums
 - Camera Enumerations, [51](#)
- spinChunkScan3dCoordinateSystemEnums
 - Camera Enumerations, [51](#)
- spinChunkScan3dCoordinateSystemReferenceEnums
 - Camera Enumerations, [51](#)
- spinChunkScan3dCoordinateTransformSelectorEnums
 - Camera Enumerations, [52](#)
- spinChunkScan3dDistanceUnitEnums
 - Camera Enumerations, [52](#)
- spinChunkScan3dOutputModeEnums
 - Camera Enumerations, [52](#)
- spinChunkSelectorEnums
 - Camera Enumerations, [53](#)
- spinChunkSourceIDEnums
 - Camera Enumerations, [54](#)
- spinChunkTimerSelectorEnums
 - Camera Enumerations, [54](#)
- spinChunkTransferStreamIDEnums
 - Camera Enumerations, [54](#)
- spinCIConfigurationEnums
 - Camera Enumerations, [54](#)
- spinCITimeSlotsCountEnums
 - Camera Enumerations, [55](#)
- spinColorProcessingAlgorithm
 - Spinnaker C Enumerations, [209](#)
- spinColorTransformationSelectorEnums
 - Camera Enumerations, [55](#)
- spinColorTransformationValueSelectorEnums
 - Camera Enumerations, [55](#)
- spinCommandExecute
 - ICommand Access, [253](#)
- spinCommandIsDone
 - ICommand Access, [253](#)
- spinCompressionMethod
 - Spinnaker C Structures, [215](#)
- spinCounterEventActivationEnums
 - Camera Enumerations, [56](#)
- spinCounterEventSourceEnums
 - Camera Enumerations, [56](#)
- spinCounterResetActivationEnums
 - Camera Enumerations, [57](#)
- spinCounterResetSourceEnums
 - Camera Enumerations, [57](#)
- spinCounterSelectorEnums
 - Camera Enumerations, [57](#)
- spinCounterStatusEnums
 - Camera Enumerations, [58](#)
- spinCounterTriggerActivationEnums

- Camera Enumerations, [58](#)
- spinCounterTriggerSourceEnums
 - Camera Enumerations, [58](#)
- spinCxpConnectionTestModeEnums
 - Camera Enumerations, [59](#)
- spinCxpLinkConfigurationEnums
 - Camera Enumerations, [59](#)
- spinCxpLinkConfigurationPreferredEnums
 - Camera Enumerations, [60](#)
- spinCxpLinkConfigurationStatusEnums
 - Camera Enumerations, [61](#)
- spinCxpPoCxpStatusEnums
 - Camera Enumerations, [62](#)
- spinDecimationHorizontalModeEnums
 - Camera Enumerations, [62](#)
- spinDecimationSelectorEnums
 - Camera Enumerations, [62](#)
- spinDecimationVerticalModeEnums
 - Camera Enumerations, [63](#)
- spinDefectCorrectionModeEnums
 - Camera Enumerations, [63](#)
- spinDeinterlacingEnums
 - Camera Enumerations, [63](#)
- spinDeviceCharacterSetEnums
 - Camera Enumerations, [63](#)
- spinDeviceClockSelectorEnums
 - Camera Enumerations, [64](#)
- spinDeviceConnectionStatusEnums
 - Camera Enumerations, [64](#)
- spinDeviceEvent
 - Spinnaker C Handles, [202](#)
- spinDeviceEventCreate
 - Event Access, [179](#)
- spinDeviceEventData
 - Spinnaker C Handles, [202](#)
- spinDeviceEventDestroy
 - Event Access, [180](#)
- spinDeviceEventFunction
 - Spinnaker C Function Signatures, [205](#)
- spinDeviceEventGetId
 - Device Event Data Access, [195](#)
- spinDeviceEventGetName
 - Device Event Data Access, [195](#)
- spinDeviceEventGetPayloadData
 - Device Event Data Access, [196](#)
- spinDeviceEventGetPayloadDataSize
 - Device Event Data Access, [196](#)
- spinDeviceIndicatorModeEnums
 - Camera Enumerations, [64](#)
- spinDeviceLinkHeartbeatModeEnums
 - Camera Enumerations, [64](#)
- spinDeviceLinkThroughputLimitModeEnums
 - Camera Enumerations, [65](#)
- spinDevicePowerSupplySelectorEnums
 - Camera Enumerations, [65](#)
- spinDeviceRegistersEndiannessEnums
 - Camera Enumerations, [65](#)
- spinDeviceScanTypeEnums
 - Camera Enumerations, [65](#)
- spinDeviceSerialPortBaudRateEnums
 - Camera Enumerations, [65](#)
- spinDeviceSerialPortSelectorEnums
 - Camera Enumerations, [66](#)
- spinDeviceStreamChannelEndiannessEnums
 - Camera Enumerations, [66](#)
- spinDeviceStreamChannelTypeEnums
 - Camera Enumerations, [66](#)
- spinDeviceTLTypeEnums
 - Camera Enumerations, [68](#)
- spinDeviceTapGeometryEnums
 - Camera Enumerations, [66](#)
- spinDeviceTemperatureSelectorEnums
 - Camera Enumerations, [68](#)
- spinDeviceTypeEnums
 - Camera Enumerations, [68](#)
- spinDisplayNotation
 - Spinnaker C GenICam Enumerations, [265](#)
- spinEncoderModeEnums
 - Camera Enumerations, [68](#)
- spinEncoderOutputModeEnums
 - Camera Enumerations, [69](#)
- spinEncoderResetActivationEnums
 - Camera Enumerations, [69](#)
- spinEncoderResetSourceEnums
 - Camera Enumerations, [69](#)
- spinEncoderSelectorEnums
 - Camera Enumerations, [70](#)
- spinEncoderSourceAEnums
 - Camera Enumerations, [71](#)
- spinEncoderSourceBEnums
 - Camera Enumerations, [71](#)
- spinEncoderStatusEnums
 - Camera Enumerations, [71](#)
- spinEndianness
 - Spinnaker C GenICam Enumerations, [265](#)
- spinEnumerationEntryGetEnumValue
 - IEnumEntry Access, [249](#)
- spinEnumerationEntryGetIntValue
 - IEnumEntry Access, [250](#)
- spinEnumerationEntryGetSymbolic
 - IEnumEntry Access, [250](#)
- spinEnumerationGetCurrentEntry
 - IEnumeration Access, [245](#)
- spinEnumerationGetEntryByIndex
 - IEnumeration Access, [246](#)
- spinEnumerationGetEntryByName
 - IEnumeration Access, [246](#)
- spinEnumerationGetNumEntries
 - IEnumeration Access, [246](#)
- spinEnumerationSetEnumValue
 - IEnumeration Access, [247](#)
- spinEnumerationSetIntValue
 - IEnumeration Access, [247](#)
- spinError
 - Spinnaker C Enumerations, [209](#)
- spinErrorGetLast

- Error Handling, [116](#)
- `spinErrorGetLastBuildDate`
 - Error Handling, [117](#)
- `spinErrorGetLastBuildTime`
 - Error Handling, [117](#)
- `spinErrorGetLastFileName`
 - Error Handling, [117](#)
- `spinErrorGetLastFullMessage`
 - Error Handling, [118](#)
- `spinErrorGetLastFunctionName`
 - Error Handling, [118](#)
- `spinErrorGetLastLineNumber`
 - Error Handling, [119](#)
- `spinErrorGetLastMessage`
 - Error Handling, [119](#)
- `spinEventNotificationEnums`
 - Camera Enumerations, [71](#)
- `spinEventSelectorEnums`
 - Camera Enumerations, [72](#)
- `spinExposureActiveModeEnums`
 - Camera Enumerations, [72](#)
- `spinExposureAutoEnums`
 - Camera Enumerations, [72](#)
- `spinExposureModeEnums`
 - Camera Enumerations, [72](#)
- `spinExposureTimeModeEnums`
 - Camera Enumerations, [73](#)
- `spinExposureTimeSelectorEnums`
 - Camera Enumerations, [73](#)
- `spinFileOpenModeEnums`
 - Camera Enumerations, [73](#)
- `spinFileOperationSelectorEnums`
 - Camera Enumerations, [74](#)
- `spinFileOperationStatusEnums`
 - Camera Enumerations, [74](#)
- `spinFileSelectorEnums`
 - Camera Enumerations, [74](#)
- `spinFloatGetMax`
 - IFloat Access, [241](#)
- `spinFloatGetMin`
 - IFloat Access, [242](#)
- `spinFloatGetRepresentation`
 - IFloat Access, [242](#)
- `spinFloatGetUnit`
 - IFloat Access, [242](#)
- `spinFloatGetValue`
 - IFloat Access, [243](#)
- `spinFloatGetValueEx`
 - IFloat Access, [243](#)
- `spinFloatSetValue`
 - IFloat Access, [244](#)
- `spinFloatSetValueEx`
 - IFloat Access, [244](#)
- `spinGainAutoBalanceEnums`
 - Camera Enumerations, [74](#)
- `spinGainAutoEnums`
 - Camera Enumerations, [75](#)
- `spinGainSelectorEnums`
 - Camera Enumerations, [75](#)
- `spinGevCCPEnums`
 - Camera Enumerations, [75](#)
- `spinGevCurrentPhysicalLinkConfigurationEnums`
 - Camera Enumerations, [75](#)
- `spinGevGVCPExtendedStatusCodesSelectorEnums`
 - Camera Enumerations, [76](#)
- `spinGevGVSPExtendedIDModeEnums`
 - Camera Enumerations, [76](#)
- `spinGevIEEE1588ClockAccuracyEnums`
 - Camera Enumerations, [76](#)
- `spinGevIEEE1588ModeEnums`
 - Camera Enumerations, [76](#)
- `spinGevIEEE1588StatusEnums`
 - Camera Enumerations, [77](#)
- `spinGevIPConfigurationStatusEnums`
 - Camera Enumerations, [77](#)
- `spinGevPhysicalLinkConfigurationEnums`
 - Camera Enumerations, [77](#)
- `spinGevSupportedOptionSelectorEnums`
 - Camera Enumerations, [77](#)
- `spinH264Option`, [334](#)
 - bitrate, [334](#)
 - frameRate, [334](#)
 - height, [334](#)
 - reserved, [334](#)
 - width, [334](#)
- `spinImage`
 - Spinnaker C Handles, [202](#)
- `spinImageCalculateStatistics`
 - Image Access, [157](#)
- `spinImageCheckCRC`
 - Image Access, [158](#)
- `spinImageChunkDataGetFloatValue`
 - Chunk data access, [200](#)
- `spinImageChunkDataGetIntValue`
 - Chunk data access, [200](#)
- `spinImageComponentSelectorEnums`
 - Camera Enumerations, [78](#)
- `spinImageCompressionJPEGFormatOptionEnums`
 - Camera Enumerations, [79](#)
- `spinImageCompressionModeEnums`
 - Camera Enumerations, [79](#)
- `spinImageCompressionRateOptionEnums`
 - Camera Enumerations, [79](#)
- `spinImageConvert`
 - Image Access, [158](#)
- `spinImageConvertEx`
 - Image Access, [158](#)
- `spinImageCreate`
 - Image Access, [159](#)
- `spinImageCreateEmpty`
 - Image Access, [159](#)
- `spinImageCreateEx`
 - Image Access, [159](#)
- `spinImageDeepCopy`
 - Image Access, [160](#)
- `spinImageDestroy`

- Image Access, [160](#)
- spinImageEvent
 - Spinnaker C Handles, [203](#)
- spinImageEventCreate
 - Event Access, [180](#)
- spinImageEventDestroy
 - Event Access, [180](#)
- spinImageEventFunction
 - Spinnaker C Function Signatures, [205](#)
- spinImageFileFormat
 - Spinnaker C Enumerations, [211](#)
- spinImageGetBitsPerPixel
 - Image Access, [161](#)
- spinImageGetBufferSize
 - Image Access, [161](#)
- spinImageGetChunkLayoutID
 - Image Access, [161](#)
- spinImageGetColorProcessing
 - Image Access, [162](#)
- spinImageGetData
 - Image Access, [162](#)
- spinImageGetDefaultColorProcessing
 - Image Access, [163](#)
- spinImageGetFrameID
 - Image Access, [163](#)
- spinImageGetHeight
 - Image Access, [163](#)
- spinImageGetID
 - Image Access, [164](#)
- spinImageGetOffsetX
 - Image Access, [164](#)
- spinImageGetOffsetY
 - Image Access, [164](#)
- spinImageGetPaddingX
 - Image Access, [165](#)
- spinImageGetPaddingY
 - Image Access, [165](#)
- spinImageGetPayloadType
 - Image Access, [165](#)
- spinImageGetPixelFormat
 - Image Access, [166](#)
- spinImageGetPixelFormatName
 - Image Access, [166](#)
- spinImageGetPrivateData
 - Image Access, [167](#)
- spinImageGetSize
 - Image Access, [167](#)
- spinImageGetStatus
 - Image Access, [167](#)
- spinImageGetStatusDescription
 - Image Access, [168](#)
- spinImageGetStride
 - Image Access, [168](#)
- spinImageGetTLPayloadType
 - Image Access, [169](#)
- spinImageGetTLPixelFormat
 - Image Access, [169](#)
- spinImageGetTLPixelFormatNamespace
 - Image Access, [170](#)
- spinImageGetTimeStamp
 - Image Access, [169](#)
- spinImageGetValidPayloadSize
 - Image Access, [170](#)
- spinImageGetWidth
 - Image Access, [171](#)
- spinImageHasCRC
 - Image Access, [171](#)
- spinImageIsIncomplete
 - Image Access, [171](#)
- spinImageRelease
 - Image Access, [172](#)
- spinImageReset
 - Image Access, [172](#)
- spinImageResetEx
 - Image Access, [173](#)
- spinImageSave
 - Image Access, [173](#)
- spinImageSaveBmp
 - Image Access, [173](#)
- spinImageSaveFromExt
 - Image Access, [174](#)
- spinImageSaveJpeg
 - Image Access, [174](#)
- spinImageSaveJpg2
 - Image Access, [175](#)
- spinImageSavePgm
 - Image Access, [175](#)
- spinImageSavePng
 - Image Access, [176](#)
- spinImageSavePpm
 - Image Access, [176](#)
- spinImageSaveTiff
 - Image Access, [176](#)
- spinImageSetDefaultColorProcessing
 - Image Access, [177](#)
- spinImageStatistics
 - Spinnaker C Handles, [203](#)
- spinImageStatisticsCreate
 - ImageStatistics Access, [185](#)
- spinImageStatisticsDestroy
 - ImageStatistics Access, [185](#)
- spinImageStatisticsDisableAll
 - ImageStatistics Access, [185](#)
- spinImageStatisticsEnableAll
 - ImageStatistics Access, [185](#)
- spinImageStatisticsEnableGreyOnly
 - ImageStatistics Access, [186](#)
- spinImageStatisticsEnableHslOnly
 - ImageStatistics Access, [186](#)
- spinImageStatisticsEnableRgbOnly
 - ImageStatistics Access, [187](#)
- spinImageStatisticsGetAll
 - ImageStatistics Access, [187](#)
- spinImageStatisticsGetChannelStatus
 - ImageStatistics Access, [187](#)
- spinImageStatisticsGetHistogram

- ImageStatistics Access, [188](#)
- spinImageStatisticsGetMean
 - ImageStatistics Access, [188](#)
- spinImageStatisticsGetNumPixelValues
 - ImageStatistics Access, [189](#)
- spinImageStatisticsGetPixelValueRange
 - ImageStatistics Access, [189](#)
- spinImageStatisticsGetRange
 - ImageStatistics Access, [190](#)
- spinImageStatisticsSetChannelStatus
 - ImageStatistics Access, [190](#)
- spinImageStatus
 - Spinnaker C Enumerations, [211](#)
- spinIncMode
 - Spinnaker C GenICam Enumerations, [265](#)
- spinInputDirection
 - Spinnaker C GenICam Enumerations, [265](#)
- spinIntegerGetInc
 - Integer Access, [237](#)
- spinIntegerGetMax
 - Integer Access, [238](#)
- spinIntegerGetMin
 - Integer Access, [238](#)
- spinIntegerGetRepresentation
 - Integer Access, [238](#)
- spinIntegerGetValue
 - Integer Access, [239](#)
- spinIntegerGetValueEx
 - Integer Access, [239](#)
- spinIntegerSetValue
 - Integer Access, [240](#)
- spinIntegerSetValueEx
 - Integer Access, [240](#)
- spinInterface
 - Spinnaker C Handles, [203](#)
- spinInterfaceEvent
 - Spinnaker C Handles, [203](#)
- spinInterfaceEventCreate
 - Event Access, [181](#)
- spinInterfaceEventDestroy
 - Event Access, [181](#)
- spinInterfaceGetCameras
 - Interface Access, [139](#)
- spinInterfaceGetCamerasEx
 - Interface Access, [139](#)
- spinInterfaceGetTLNodeMap
 - Interface Access, [140](#)
- spinInterfaceIsInUse
 - Interface Access, [140](#)
- spinInterfaceList
 - Spinnaker C Handles, [203](#)
- spinInterfaceListClear
 - InterfaceList Access, [130](#)
- spinInterfaceListCreateEmpty
 - InterfaceList Access, [131](#)
- spinInterfaceListDestroy
 - InterfaceList Access, [131](#)
- spinInterfaceListGet
 - InterfaceList Access, [131](#)
- spinInterfaceListGetSize
 - InterfaceList Access, [132](#)
- spinInterfaceRegisterArrivalEvent
 - Interface Access, [140](#)
- spinInterfaceRegisterInterfaceEvent
 - Interface Access, [141](#)
- spinInterfaceRegisterRemovalEvent
 - Interface Access, [141](#)
- spinInterfaceRelease
 - Interface Access, [141](#)
- spinInterfaceSendActionCommand
 - Interface Access, [142](#)
- spinInterfaceType
 - Spinnaker C GenICam Enumerations, [266](#)
- spinInterfaceUnregisterArrivalEvent
 - Interface Access, [142](#)
- spinInterfaceUnregisterInterfaceEvent
 - Interface Access, [143](#)
- spinInterfaceUnregisterRemovalEvent
 - Interface Access, [143](#)
- spinInterfaceUpdateCameras
 - Interface Access, [144](#)
- spinJPEGOption, [335](#)
 - progressive, [335](#)
 - quality, [335](#)
 - reserved, [335](#)
- spinJPG2Option, [336](#)
 - quality, [336](#)
 - reserved, [336](#)
- spinLUTSelectorEnums
 - Camera Enumerations, [82](#)
- spinLibraryVersion, [336](#)
 - build, [337](#)
 - major, [337](#)
 - minor, [337](#)
 - type, [337](#)
- spinLineFormatEnums
 - Camera Enumerations, [79](#)
- spinLineInputFilterSelectorEnums
 - Camera Enumerations, [80](#)
- spinLineModeEnums
 - Camera Enumerations, [80](#)
- spinLineSelectorEnums
 - Camera Enumerations, [80](#)
- spinLineSourceEnums
 - Camera Enumerations, [80](#)
- spinLinkType
 - Spinnaker C GenICam Enumerations, [266](#)
- spinLogDataGetCategoryName
 - Logging Event Data Access, [191](#)
- spinLogDataGetLogMessage
 - Logging Event Data Access, [192](#)
- spinLogDataGetNDC
 - Logging Event Data Access, [192](#)
- spinLogDataGetPriority
 - Logging Event Data Access, [192](#)
- spinLogDataGetPriorityName

- Logging Event Data Access, [193](#)
- spinLogDataGetThreadName
 - Logging Event Data Access, [193](#)
- spinLogDataGetTimestamp
 - Logging Event Data Access, [194](#)
- spinLogEvent
 - Spinnaker C Handles, [203](#)
- spinLogEventCreate
 - Event Access, [181](#)
- spinLogEventData
 - Spinnaker C Handles, [203](#)
- spinLogEventDestroy
 - Event Access, [182](#)
- spinLogEventFunction
 - Spinnaker C Function Signatures, [205](#)
- spinLogicBlockLUTInputActivationEnums
 - Camera Enumerations, [81](#)
- spinLogicBlockLUTInputSelectorEnums
 - Camera Enumerations, [81](#)
- spinLogicBlockLUTInputSourceEnums
 - Camera Enumerations, [81](#)
- spinLogicBlockLUTSelectorEnums
 - Camera Enumerations, [82](#)
- spinLogicBlockSelectorEnums
 - Camera Enumerations, [82](#)
- spinMJPGOption, [337](#)
 - frameRate, [338](#)
 - quality, [338](#)
 - reserved, [338](#)
- spinNameSpace
 - Spinnaker C GenICam Enumerations, [266](#)
- spinNodeCallbackFunction
 - Spinnaker C GenICam Handles, [261](#)
- spinNodeCallbackHandle
 - Spinnaker C GenICam Handles, [261](#)
- spinNodeDeregisterCallback
 - Node Access, [222](#)
- spinNodeFromString
 - IValue Access, [231](#)
- spinNodeFromStringEx
 - IValue Access, [232](#)
- spinNodeGetAccessMode
 - Node Access, [222](#)
- spinNodeGetCachingMode
 - Node Access, [223](#)
- spinNodeGetDescription
 - Node Access, [223](#)
- spinNodeGetDisplayName
 - Node Access, [223](#)
- spinNodeGetImposedAccessMode
 - Node Access, [224](#)
- spinNodeGetImposedVisibility
 - Node Access, [224](#)
- spinNodeGetName
 - Node Access, [225](#)
- spinNodeGetNameSpace
 - Node Access, [225](#)
- spinNodeGetPollingTime
 - Node Access, [225](#)
- spinNodeGetToolTip
 - Node Access, [226](#)
- spinNodeGetType
 - Node Access, [226](#)
- spinNodeGetVisibility
 - Node Access, [227](#)
- spinNodeHandle
 - Spinnaker C GenICam Handles, [261](#)
- spinNodeInvalidateNode
 - Node Access, [227](#)
- spinNodesAvailable
 - Node Access, [227](#)
- spinNodesEqual
 - Node Access, [228](#)
- spinNodesImplemented
 - Node Access, [228](#)
- spinNodesReadable
 - Node Access, [229](#)
- spinNodesWritable
 - Node Access, [229](#)
- spinNodeMapGetNode
 - Node Map Access, [218](#)
- spinNodeMapGetNodeByIndex
 - Node Map Access, [219](#)
- spinNodeMapGetNumNodes
 - Node Map Access, [219](#)
- spinNodeMapHandle
 - Spinnaker C GenICam Handles, [261](#)
- spinNodeMapPoll
 - Node Map Access, [219](#)
- spinNodeRegisterCallback
 - Node Access, [229](#)
- spinNodeToString
 - IValue Access, [232](#)
- spinNodeToStringEx
 - IValue Access, [232](#)
- spinNodeType
 - Spinnaker C GenICam Enumerations, [267](#)
- spinPGMOption, [338](#)
 - binaryFile, [339](#)
 - reserved, [339](#)
- spinPNGOption, [339](#)
 - compressionLevel, [339](#)
 - interlaced, [339](#)
 - reserved, [339](#)
- spinPPMOption, [340](#)
 - binaryFile, [340](#)
 - reserved, [340](#)
- spinPayloadTypeInfoFolds
 - Spinnaker C Enumerations, [212](#)
- spinPixelFormatEnums
 - Camera Enumerations, [83](#)
- spinPixelFormatInfoSelectorEnums
 - Camera Enumerations, [88](#)
- spinPixelFormatNamespaceID

- Spinnaker C Enumerations, [212](#)
- spinPixelFormatEnums
 - Camera Enumerations, [94](#)
- spinRegionDestinationEnums
 - Camera Enumerations, [94](#)
- spinRegionModeEnums
 - Camera Enumerations, [94](#)
- spinRegionSelectorEnums
 - Camera Enumerations, [95](#)
- spinRegisterGet
 - IRegister Access, [257](#)
- spinRegisterGetAddress
 - IRegister Access, [258](#)
- spinRegisterGetEx
 - IRegister Access, [258](#)
- spinRegisterGetLength
 - IRegister Access, [259](#)
- spinRegisterSet
 - IRegister Access, [259](#)
- spinRegisterSetEx
 - IRegister Access, [259](#)
- spinRegisterSetReference
 - IRegister Access, [260](#)
- spinRemovalEvent
 - Spinnaker C Handles, [204](#)
- spinRemovalEventCreate
 - Event Access, [182](#)
- spinRemovalEventDestroy
 - Event Access, [183](#)
- spinRemovalEventFunction
 - Spinnaker C Function Signatures, [205](#)
- spinRepresentation
 - Spinnaker C GenICam Enumerations, [267](#)
- spinRgbTransformLightSourceEnums
 - Camera Enumerations, [95](#)
- spinScan3dCoordinateReferenceSelectorEnums
 - Camera Enumerations, [95](#)
- spinScan3dCoordinateSelectorEnums
 - Camera Enumerations, [96](#)
- spinScan3dCoordinateSystemEnums
 - Camera Enumerations, [96](#)
- spinScan3dCoordinateSystemReferenceEnums
 - Camera Enumerations, [96](#)
- spinScan3dCoordinateTransformSelectorEnums
 - Camera Enumerations, [96](#)
- spinScan3dDistanceUnitEnums
 - Camera Enumerations, [97](#)
- spinScan3dOutputModeEnums
 - Camera Enumerations, [97](#)
- spinSensorDigitizationTapsEnums
 - Camera Enumerations, [98](#)
- spinSensorShutterModeEnums
 - Camera Enumerations, [98](#)
- spinSensorTapsEnums
 - Camera Enumerations, [98](#)
- spinSequencerConfigurationModeEnums
 - Camera Enumerations, [98](#)
- spinSequencerConfigurationValidEnums
 - Camera Enumerations, [99](#)
- spinSequencerModeEnums
 - Camera Enumerations, [99](#)
- spinSequencerSetValidEnums
 - Camera Enumerations, [99](#)
- spinSequencerTriggerActivationEnums
 - Camera Enumerations, [99](#)
- spinSequencerTriggerSourceEnums
 - Camera Enumerations, [99](#)
- spinSerialPortBaudRateEnums
 - Camera Enumerations, [100](#)
- spinSerialPortParityEnums
 - Camera Enumerations, [100](#)
- spinSerialPortSelectorEnums
 - Camera Enumerations, [100](#)
- spinSerialPortSourceEnums
 - Camera Enumerations, [101](#)
- spinSerialPortStopBitsEnums
 - Camera Enumerations, [101](#)
- spinSign
 - Spinnaker C GenICam Enumerations, [267](#)
- spinSlope
 - Spinnaker C GenICam Enumerations, [268](#)
- spinSoftwareSignalSelectorEnums
 - Camera Enumerations, [101](#)
- spinSourceSelectorEnums
 - Camera Enumerations, [101](#)
- spinStandardNameSpace
 - Spinnaker C GenICam Enumerations, [268](#)
- spinStatisticsChannel
 - Spinnaker C Enumerations, [212](#)
- spinStringGetMaxLength
 - String Access, [234](#)
- spinStringGetValue
 - String Access, [235](#)
- spinStringGetValueEx
 - String Access, [235](#)
- spinStringSetValue
 - String Access, [236](#)
- spinStringSetValueEx
 - String Access, [236](#)
- spinSystem
 - Spinnaker C Handles, [204](#)
- spinSystemGetCameras
 - System Access, [121](#)
- spinSystemGetCamerasEx
 - System Access, [122](#)
- spinSystemGetInstance
 - System Access, [122](#)
- spinSystemGetInterfaces
 - System Access, [122](#)
- spinSystemGetLibraryVersion
 - System Access, [123](#)
- spinSystemGetLoggingLevel
 - System Access, [123](#)
- spinSystemIsInUse
 - System Access, [123](#)
- spinSystemRegisterArrivalEvent

- System Access, [124](#)
- spinSystemRegisterInterfaceEvent
 - System Access, [124](#)
- spinSystemRegisterLogEvent
 - System Access, [124](#)
- spinSystemRegisterRemovalEvent
 - System Access, [125](#)
- spinSystemReleaseInstance
 - System Access, [125](#)
- spinSystemSendActionCommand
 - System Access, [126](#)
- spinSystemSetLoggingLevel
 - System Access, [126](#)
- spinSystemUnregisterAllLogEvents
 - System Access, [127](#)
- spinSystemUnregisterArrivalEvent
 - System Access, [127](#)
- spinSystemUnregisterInterfaceEvent
 - System Access, [127](#)
- spinSystemUnregisterLogEvent
 - System Access, [128](#)
- spinSystemUnregisterRemovalEvent
 - System Access, [128](#)
- spinSystemUpdateCameras
 - System Access, [128](#)
- spinSystemUpdateCamerasEx
 - System Access, [129](#)
- spinTIFFOption, [340](#)
 - compression, [341](#)
 - reserved, [341](#)
- spinTLDeviceAccessStatusEnums
 - Transport Layer Enumerations, [273](#)
- spinTLDeviceCurrentSpeedEnums
 - Transport Layer Enumerations, [273](#)
- spinTLDeviceEndianessMechanismEnums
 - Transport Layer Enumerations, [274](#)
- spinTLDeviceTypeEnums
 - Transport Layer Enumerations, [274](#)
- spinTLGUIXMLLocationEnums
 - Transport Layer Enumerations, [275](#)
- spinTLGenICamXMLLocationEnums
 - Transport Layer Enumerations, [274](#)
- spinTLGevCCPEnums
 - Transport Layer Enumerations, [275](#)
- spinTLPOEStatusEnums
 - Transport Layer Enumerations, [275](#)
- spinTLStreamBufferCountModeEnums
 - Transport Layer Enumerations, [275](#)
- spinTLStreamBufferHandlingModeEnums
 - Transport Layer Enumerations, [276](#)
- spinTLStreamDefaultBufferCountModeEnums
 - Transport Layer Enumerations, [276](#)
- spinTLStreamTypeEnums
 - Transport Layer Enumerations, [276](#)
- spinTestPatternEnums
 - Camera Enumerations, [102](#)
- spinTestPatternGeneratorSelectorEnums
 - Camera Enumerations, [102](#)
- spinTimerSelectorEnums
 - Camera Enumerations, [102](#)
- spinTimerStatusEnums
 - Camera Enumerations, [102](#)
- spinTimerTriggerActivationEnums
 - Camera Enumerations, [103](#)
- spinTimerTriggerSourceEnums
 - Camera Enumerations, [103](#)
- spinTransferComponentSelectorEnums
 - Camera Enumerations, [104](#)
- spinTransferControlModeEnums
 - Camera Enumerations, [105](#)
- spinTransferOperationModeEnums
 - Camera Enumerations, [105](#)
- spinTransferQueueModeEnums
 - Camera Enumerations, [105](#)
- spinTransferSelectorEnums
 - Camera Enumerations, [105](#)
- spinTransferStatusSelectorEnums
 - Camera Enumerations, [106](#)
- spinTransferTriggerActivationEnums
 - Camera Enumerations, [106](#)
- spinTransferTriggerModeEnums
 - Camera Enumerations, [106](#)
- spinTransferTriggerSelectorEnums
 - Camera Enumerations, [107](#)
- spinTransferTriggerSourceEnums
 - Camera Enumerations, [107](#)
- spinTriggerActivationEnums
 - Camera Enumerations, [108](#)
- spinTriggerModeEnums
 - Camera Enumerations, [108](#)
- spinTriggerOverlapEnums
 - Camera Enumerations, [108](#)
- spinTriggerSelectorEnums
 - Camera Enumerations, [109](#)
- spinTriggerSourceEnums
 - Camera Enumerations, [109](#)
- spinUserOutputSelectorEnums
 - Camera Enumerations, [109](#)
- spinUserSetDefaultEnums
 - Camera Enumerations, [110](#)
- spinUserSetSelectorEnums
 - Camera Enumerations, [110](#)
- spinVideo
 - Spinnaker C Handles, [204](#)
- SpinVideo Recording Access, [270](#)
 - spinVideoAppend, [270](#)
 - spinVideoClose, [270](#)
 - spinVideoOpenH264, [270](#)
 - spinVideoOpenMJPEG, [270](#)
 - spinVideoOpenUncompressed, [270](#)
 - spinVideoSetMaximumFileSize, [270](#)
- spinVideoAppend
 - SpinVideo Recording Access, [270](#)
- spinVideoClose
 - SpinVideo Recording Access, [270](#)
- spinVideoOpenH264

- SpinVideo Recording Access, [270](#)
- spinVideoOpenMJPEG
 - SpinVideo Recording Access, [270](#)
- spinVideoOpenUncompressed
 - SpinVideo Recording Access, [270](#)
- spinVideoSetMaximumFileSize
 - SpinVideo Recording Access, [270](#)
- spinVisibility
 - Spinnaker C GenICam Enumerations, [268](#)
- spinWhiteClipSelectorEnums
 - Camera Enumerations, [110](#)
- spinXMLValidation
 - Spinnaker C GenICam Enumerations, [268](#)
- spinYesNo
 - Spinnaker C GenICam Enumerations, [269](#)
- Spinnaker C API, [114](#)
 - spinCameraDiscoverMaxPacketSize, [115](#)
- Spinnaker C Definitions, [11](#)
 - bool8_t, [12](#)
 - False, [12](#)
 - True, [12](#)
- Spinnaker C Enumerations, [206](#)
 - BLUE, [213](#)
 - BMP, [211](#)
 - DEFAULT, [209](#)
 - DIRECTIONAL_FILTER, [209](#)
 - EDGE_SENSING, [209](#)
 - FROM_FILE_EXT, [211](#)
 - GENICAM_ERR_ACCESS, [210](#)
 - GENICAM_ERR_BAD_ALLOCATION, [210](#)
 - GENICAM_ERR_DYNAMIC_CAST, [210](#)
 - GENICAM_ERR_GENERIC, [210](#)
 - GENICAM_ERR_INVALID_ARGUMENT, [210](#)
 - GENICAM_ERR_LOGICAL, [210](#)
 - GENICAM_ERR_OUT_OF_RANGE, [210](#)
 - GENICAM_ERR_PROPERTY, [210](#)
 - GENICAM_ERR_RUN_TIME, [210](#)
 - GENICAM_ERR_TIMEOUT, [210](#)
 - GREEN, [213](#)
 - GREY, [213](#)
 - HQ_LINEAR, [209](#)
 - HUE, [213](#)
 - IMAGE_CHUNK_DATA_INVALID, [211](#)
 - IMAGE_CRC_CHECK_FAILED, [211](#)
 - IMAGE_DATA_INCOMPLETE, [211](#)
 - IMAGE_DATA_OVERFLOW, [211](#)
 - IMAGE_FILE_FORMAT_FORCE_32BITS, [211](#)
 - IMAGE_INFO_INCONSISTENT, [211](#)
 - IMAGE_LEADER_BUFFER_SIZE_INCONSISTENT, [211](#)
 - IMAGE_MISSING_LEADER, [211](#)
 - IMAGE_MISSING_PACKETS, [211](#)
 - IMAGE_MISSING_TRAILER, [211](#)
 - IMAGE_NO_ERROR, [211](#)
 - IMAGE_NO_SYSTEM_RESOURCES, [211](#)
 - IMAGE_PACKETID_INCONSISTENT, [211](#)
 - IMAGE_TRAILER_BUFFER_SIZE_INCONSISTENT, [211](#)
 - IMAGE_UNKNOWN_ERROR, [211](#)
 - IPP, [209](#)
 - JPEG2000, [211](#)
 - JPEG, [211](#)
 - LIGHTNESS, [213](#)
 - LOG_LEVEL_ALERT, [212](#)
 - LOG_LEVEL_CRIT, [212](#)
 - LOG_LEVEL_DEBUG, [212](#)
 - LOG_LEVEL_ERROR, [212](#)
 - LOG_LEVEL_FATAL, [212](#)
 - LOG_LEVEL_INFO, [212](#)
 - LOG_LEVEL_NOTICE, [212](#)
 - LOG_LEVEL_NOTSET, [212](#)
 - LOG_LEVEL_OFF, [212](#)
 - LOG_LEVEL_WARN, [212](#)
 - NEAREST_NEIGHBOR, [209](#)
 - NO_COLOR_PROCESSING, [209](#)
 - NUM_STATISTICS_CHANNELS, [213](#)
 - PAYLOAD_TYPE_CHUNK_DATA, [212](#)
 - PAYLOAD_TYPE_CHUNK_ONLY, [212](#)
 - PAYLOAD_TYPE_CUSTOM_ID, [212](#)
 - PAYLOAD_TYPE_DEVICE_SPECIFIC, [212](#)
 - PAYLOAD_TYPE_EXTENDED_CHUNK, [212](#)
 - PAYLOAD_TYPE_FILE, [212](#)
 - PAYLOAD_TYPE_H264, [212](#)
 - PAYLOAD_TYPE_IMAGE, [212](#)
 - PAYLOAD_TYPE_JPEG2000, [212](#)
 - PAYLOAD_TYPE_JPEG, [212](#)
 - PAYLOAD_TYPE_MULTI_PART, [212](#)
 - PAYLOAD_TYPE_RAW_DATA, [212](#)
 - PAYLOAD_TYPE_UNKNOWN, [212](#)
 - PGM, [211](#)
 - PNG, [211](#)
 - PPM, [211](#)
 - RAW, [211](#)
 - RED, [213](#)
 - RIGOROUS, [209](#)
 - SATURATION, [213](#)
 - SPINNAKER_ERR_ABORT, [210](#)
 - SPINNAKER_ERR_ACCESS_DENIED, [210](#)
 - SPINNAKER_ERR_BUFFER_TOO_SMALL, [210](#)
 - SPINNAKER_ERR_BUSY, [210](#)
 - SPINNAKER_ERR_CUSTOM_ID, [211](#)
 - SPINNAKER_ERR_ERROR, [210](#)
 - SPINNAKER_ERR_IM_COLOR_CONVERSION, [211](#)
 - SPINNAKER_ERR_IM_CONVERT, [210](#)
 - SPINNAKER_ERR_IM_COPY, [210](#)
 - SPINNAKER_ERR_IM_HISTOGRAM_MEAN, [211](#)
 - SPINNAKER_ERR_IM_HISTOGRAM_RANGE, [211](#)
 - SPINNAKER_ERR_IM_MALLOC, [210](#)
 - SPINNAKER_ERR_IM_MIN_MAX, [211](#)
 - SPINNAKER_ERR_IM_NOT_SUPPORTED, [211](#)
 - SPINNAKER_ERR_INVALID_ADDRESS, [210](#)
 - SPINNAKER_ERR_INVALID_BUFFER, [210](#)
 - SPINNAKER_ERR_INVALID_HANDLE, [210](#)
 - SPINNAKER_ERR_INVALID_INDEX, [210](#)

- SPINNAKER_ERR_INVALID_ID, [210](#)
- SPINNAKER_ERR_INVALID_PARAMETER, [210](#)
- SPINNAKER_ERR_INVALID_VALUE, [210](#)
- SPINNAKER_ERR_IO, [210](#)
- SPINNAKER_ERR_NO_DATA, [210](#)
- SPINNAKER_ERR_NOT_AVAILABLE, [210](#)
- SPINNAKER_ERR_NOT_IMPLEMENTED, [210](#)
- SPINNAKER_ERR_NOT_INITIALIZED, [210](#)
- SPINNAKER_ERR_OUT_OF_MEMORY, [210](#)
- SPINNAKER_ERR_PARSING_CHUNK_DATA, [210](#)
- SPINNAKER_ERR_RESOURCE_EXHAUSTED, [210](#)
- SPINNAKER_ERR_RESOURCE_IN_USE, [210](#)
- SPINNAKER_ERR_SUCCESS, [210](#)
- SPINNAKER_ERR_TIMEOUT, [210](#)
- SPINNAKER_PIXELFORMAT_NAMESPACE_C↔
CUSTOM_ID, [212](#)
- SPINNAKER_PIXELFORMAT_NAMESPACE_↔
GEV, [212](#)
- SPINNAKER_PIXELFORMAT_NAMESPACE_I↔
DC, [212](#)
- SPINNAKER_PIXELFORMAT_NAMESPACE_P↔
FNC_16BIT, [212](#)
- SPINNAKER_PIXELFORMAT_NAMESPACE_P↔
FNC_32BIT, [212](#)
- SPINNAKER_PIXELFORMAT_NAMESPACE_U↔
UNKNOWN, [212](#)
- spinColorProcessingAlgorithm, [209](#)
- spinError, [209](#)
- spinImageFileFormat, [211](#)
- spinImageStatus, [211](#)
- spinPayloadTypeInfoIDs, [212](#)
- spinPixelFormatNamespaceID, [212](#)
- spinStatisticsChannel, [212](#)
- spinnakerLogLevel, [211](#)
- TIFF, [211](#)
- WEIGHTED_DIRECTIONAL_FILTER, [209](#)
- Spinnaker C Function Signatures, [205](#)
 - spinArrivalEventFunction, [205](#)
 - spinDeviceEventFunction, [205](#)
 - spinImageEventFunction, [205](#)
 - spinLogEventFunction, [205](#)
 - spinRemovalEventFunction, [205](#)
- Spinnaker C GenICam API, [216](#)
- Spinnaker C GenICam Enumerations, [262](#)
 - _CycleDetectAccessMode, [264](#)
 - _UndefinedAccessMode, [264](#)
 - _UndefinedCachingMode, [265](#)
 - _UndefinedEDisplayNotation, [265](#)
 - _UndefinedESlope, [268](#)
 - _UndefinedEXMLValidation, [269](#)
 - _UndefinedEndian, [265](#)
 - _UndefinedNameSpace, [267](#)
 - _UndefinedRepresentation, [267](#)
 - _UndefinedSign, [268](#)
 - _UndefinedStandardNameSpace, [268](#)
 - _UndefinedVisibility, [268](#)
 - _UndefinedYesNo, [269](#)
 - Automatic, [268](#)
 - BaseNode, [267](#)
 - Beginner, [268](#)
 - BigEndian, [265](#)
 - Boolean, [267](#)
 - BooleanNode, [267](#)
 - CategoryNode, [267](#)
 - CL, [268](#)
 - CommandNode, [267](#)
 - ctAllDependingNodes, [266](#)
 - ctAllTerminalNodes, [266](#)
 - ctDependingChildren, [266](#)
 - ctInvalidators, [266](#)
 - ctReadingChildren, [266](#)
 - ctWritingChildren, [266](#)
 - Custom, [267](#)
 - Decreasing, [268](#)
 - EnumEntryNode, [267](#)
 - EnumerationNode, [267](#)
 - Expert, [268](#)
 - fixedIncrement, [265](#)
 - FloatNode, [267](#)
 - fnAutomatic, [265](#)
 - fnFixed, [265](#)
 - fnScientific, [265](#)
 - GEV, [268](#)
 - Guru, [268](#)
 - HexNumber, [267](#)
 - IIDC, [268](#)
 - IPV4Address, [267](#)
 - idFrom, [266](#)
 - idNone, [266](#)
 - idTo, [266](#)
 - Increasing, [268](#)
 - IntegerNode, [267](#)
 - intfIBase, [266](#)
 - intfIBoolean, [266](#)
 - intfICategory, [266](#)
 - intfICommand, [266](#)
 - intfIEnumEntry, [266](#)
 - intfIEnumeration, [266](#)
 - intfIFloat, [266](#)
 - intfIInteger, [266](#)
 - intfIPort, [266](#)
 - intfIRegister, [266](#)
 - intfIString, [266](#)
 - intfIValue, [266](#)
 - Invisible, [268](#)
 - Linear, [267](#)
 - listIncrement, [265](#)
 - LittleEndian, [265](#)
 - Logarithmic, [267](#)
 - MACAddress, [267](#)
 - NA, [264](#)
 - NI, [264](#)
 - No, [269](#)
 - NoCache, [265](#)

- noIncrement, [265](#)
- None, [268](#)
- PortNode, [267](#)
- PureNumber, [267](#)
- RegisterNode, [267](#)
- RO, [264](#)
- RW, [264](#)
- Signed, [268](#)
- spinAccessMode, [264](#)
- spinCachingMode, [264](#)
- spinDisplayNotation, [265](#)
- spinEndianess, [265](#)
- spinIncMode, [265](#)
- spinInputDirection, [265](#)
- spinInterfaceType, [266](#)
- spinLinkType, [266](#)
- spinNameSpace, [266](#)
- spinNodeType, [267](#)
- spinRepresentation, [267](#)
- spinSign, [267](#)
- spinSlope, [268](#)
- spinStandardNameSpace, [268](#)
- spinVisibility, [268](#)
- spinXMLValidation, [268](#)
- spinYesNo, [269](#)
- Standard, [267](#)
- StringNode, [267](#)
- USB, [268](#)
- UnknownNode, [267](#)
- Unsigned, [268](#)
- ValueNode, [267](#)
- Varying, [268](#)
- WO, [264](#)
- WriteAround, [265](#)
- WriteThrough, [265](#)
- xvAll, [269](#)
- xvCycles, [269](#)
- xvDefault, [269](#)
- xvLoad, [269](#)
- xvSFNC, [269](#)
- Yes, [269](#)
- Spinnaker C GenICam Handles, [261](#)
 - spinNodeCallbackFunction, [261](#)
 - spinNodeCallbackHandle, [261](#)
 - spinNodeHandle, [261](#)
 - spinNodeMapHandle, [261](#)
- Spinnaker C Handles, [201](#)
 - spinAVIRecorder, [202](#)
 - spinArrivalEvent, [202](#)
 - spinCamera, [202](#)
 - spinCameraList, [202](#)
 - spinDeviceEvent, [202](#)
 - spinDeviceEventData, [202](#)
 - spinImage, [202](#)
 - spinImageEvent, [203](#)
 - spinImageStatistics, [203](#)
 - spinInterface, [203](#)
 - spinInterfaceEvent, [203](#)
 - spinInterfaceList, [203](#)
 - spinLogEvent, [203](#)
 - spinLogEventData, [203](#)
 - spinRemovalEvent, [204](#)
 - spinSystem, [204](#)
 - spinVideo, [204](#)
- Spinnaker C QuickSpin API, [112](#)
- Spinnaker C Structures, [214](#)
 - ACTION_COMMAND_STATUS_ACTION_LATE, [215](#)
 - ACTION_COMMAND_STATUS_ERROR, [215](#)
 - ACTION_COMMAND_STATUS_NO_REF_TIME, [215](#)
 - ACTION_COMMAND_STATUS_OVERFLOW, [215](#)
 - ACTION_COMMAND_STATUS_OK, [215](#)
 - ADOBE_DEFLATE, [215](#)
 - actionCommandStatus, [215](#)
 - CCITTFAX3, [215](#)
 - CCITTFAX4, [215](#)
 - DEFLATE, [215](#)
 - JPG, [215](#)
 - LZW, [215](#)
 - NONE, [215](#)
 - PACKBITS, [215](#)
 - spinCompressionMethod, [215](#)
- spinnakerLogLevel
 - Spinnaker C Enumerations, [211](#)
- SpinnakerPlatformC.h
 - SPINNAKERC_API, [398](#)
- Standard
 - Spinnaker C GenICam Enumerations, [267](#)
- Status
 - actionCommandResult, [281](#)
- StreamBlockTransferSize
 - quickSpinTLStream, [329](#)
- StreamBufferCountManual
 - quickSpinTLStream, [329](#)
- StreamBufferCountMax
 - quickSpinTLStream, [329](#)
- StreamBufferCountMode
 - quickSpinTLStream, [329](#)
- StreamBufferCountMode_Auto
 - Transport Layer Enumerations, [276](#)
- StreamBufferCountMode_Manual
 - Transport Layer Enumerations, [276](#)
- StreamBufferCountResult
 - quickSpinTLStream, [329](#)
- StreamBufferHandlingMode
 - quickSpinTLStream, [329](#)
- StreamBufferHandlingMode_NewestFirst
 - Transport Layer Enumerations, [276](#)
- StreamBufferHandlingMode_NewestFirstOverwrite
 - Transport Layer Enumerations, [276](#)
- StreamBufferHandlingMode_NewestOnly
 - Transport Layer Enumerations, [276](#)
- StreamBufferHandlingMode_OldestFirst
 - Transport Layer Enumerations, [276](#)

- StreamBufferHandlingMode_OldestFirstOverwrite
 - Transport Layer Enumerations, 276
- StreamBufferUnderrunCount
 - quickSpinTLStream, 329
- StreamCRCCheckEnable
 - quickSpinTLStream, 329
- StreamDefaultBufferCount
 - quickSpinTLStream, 329
- StreamDefaultBufferCountMax
 - quickSpinTLStream, 329
- StreamDefaultBufferCountMode
 - quickSpinTLStream, 329
- StreamDefaultBufferCountMode_Auto
 - Transport Layer Enumerations, 276
- StreamDefaultBufferCountMode_Manual
 - Transport Layer Enumerations, 276
- StreamFailedBufferCount
 - quickSpinTLStream, 329
- StreamID
 - quickSpinTLStream, 329
- StreamTotalBufferCount
 - quickSpinTLStream, 329
- StreamType
 - quickSpinTLStream, 329
- StreamType_CLHS
 - Transport Layer Enumerations, 277
- StreamType_CXP
 - Transport Layer Enumerations, 277
- StreamType_CL
 - Transport Layer Enumerations, 277
- StreamType_Custom
 - Transport Layer Enumerations, 277
- StreamType_ETHERNET
 - Transport Layer Enumerations, 277
- StreamType_GEV
 - Transport Layer Enumerations, 277
- StreamType_IIDC
 - Transport Layer Enumerations, 277
- StreamType_Mixed
 - Transport Layer Enumerations, 277
- StreamType_PCI
 - Transport Layer Enumerations, 277
- StreamType_U3V
 - Transport Layer Enumerations, 277
- StreamType_UVC
 - Transport Layer Enumerations, 277
- String Access, 234
 - spinStringGetMaxLength, 234
 - spinStringGetValue, 235
 - spinStringGetValueEx, 235
 - spinStringSetValue, 236
 - spinStringSetValueEx, 236
- StringNode
 - Spinnaker C GenICam Enumerations, 267
- System Access, 120
 - spinSystemGetCameras, 121
 - spinSystemGetCamerasEx, 122
 - spinSystemGetInstance, 122
 - spinSystemGetInterfaces, 122
 - spinSystemGetLibraryVersion, 123
 - spinSystemGetLoggingLevel, 123
 - spinSystemIsInUse, 123
 - spinSystemRegisterArrivalEvent, 124
 - spinSystemRegisterInterfaceEvent, 124
 - spinSystemRegisterLogEvent, 124
 - spinSystemRegisterRemovalEvent, 125
 - spinSystemReleaseInstance, 125
 - spinSystemSendActionCommand, 126
 - spinSystemSetLoggingLevel, 126
 - spinSystemUnregisterAllLogEvents, 127
 - spinSystemUnregisterArrivalEvent, 127
 - spinSystemUnregisterInterfaceEvent, 127
 - spinSystemUnregisterLogEvent, 128
 - spinSystemUnregisterRemovalEvent, 128
 - spinSystemUpdateCameras, 128
 - spinSystemUpdateCamerasEx, 129
- TIFF
 - Spinnaker C Enumerations, 211
- TLDevice Structures, 278
- TLInterface Structures, 279
- TLParamsLocked
 - quickSpin, 321
- TLStream Structures, 280
- Test0001
 - quickSpin, 320
- TestEventGenerate
 - quickSpin, 320
- TestPattern
 - quickSpin, 320
- TestPattern_Increment
 - Camera Enumerations, 102
- TestPattern_Off
 - Camera Enumerations, 102
- TestPattern_SensorTestPattern
 - Camera Enumerations, 102
- TestPatternGeneratorSelector
 - quickSpin, 320
- TestPatternGeneratorSelector_PipelineStart
 - Camera Enumerations, 102
- TestPatternGeneratorSelector_Sensor
 - Camera Enumerations, 102
- TestPendingAck
 - quickSpin, 320
- TimerDelay
 - quickSpin, 320
- TimerDuration
 - quickSpin, 320
- TimerReset
 - quickSpin, 320
- TimerSelector
 - quickSpin, 320
- TimerSelector_Timer0
 - Camera Enumerations, 102
- TimerSelector_Timer1
 - Camera Enumerations, 102
- TimerSelector_Timer2

- Camera Enumerations, [102](#)
- TimerStatus
 - quickSpin, [321](#)
- TimerStatus_TimerActive
 - Camera Enumerations, [103](#)
- TimerStatus_TimerCompleted
 - Camera Enumerations, [103](#)
- TimerStatus_TimerIdle
 - Camera Enumerations, [103](#)
- TimerStatus_TimerTriggerWait
 - Camera Enumerations, [103](#)
- TimerTriggerActivation
 - quickSpin, [321](#)
- TimerTriggerActivation_AnyEdge
 - Camera Enumerations, [103](#)
- TimerTriggerActivation_FallingEdge
 - Camera Enumerations, [103](#)
- TimerTriggerActivation_LevelHigh
 - Camera Enumerations, [103](#)
- TimerTriggerActivation_LevelLow
 - Camera Enumerations, [103](#)
- TimerTriggerActivation_RisingEdge
 - Camera Enumerations, [103](#)
- TimerTriggerSource
 - quickSpin, [321](#)
- TimerTriggerSource_AcquisitionEnd
 - Camera Enumerations, [103](#)
- TimerTriggerSource_AcquisitionStart
 - Camera Enumerations, [103](#)
- TimerTriggerSource_AcquisitionTrigger
 - Camera Enumerations, [103](#)
- TimerTriggerSource_Action0
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Action1
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Action2
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Counter0End
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Counter0Start
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Counter1End
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Counter1Start
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Counter2End
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Counter2Start
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Encoder0
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Encoder1
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Encoder2
 - Camera Enumerations, [104](#)
- TimerTriggerSource_ExposureEnd
 - Camera Enumerations, [103](#)
- TimerTriggerSource_ExposureStart
 - Camera Enumerations, [103](#)
- Camera Enumerations, [103](#)
- TimerTriggerSource_FrameBurstEnd
 - Camera Enumerations, [103](#)
- TimerTriggerSource_FrameBurstStart
 - Camera Enumerations, [103](#)
- TimerTriggerSource_FrameEnd
 - Camera Enumerations, [103](#)
- TimerTriggerSource_FrameStart
 - Camera Enumerations, [103](#)
- TimerTriggerSource_FrameTrigger
 - Camera Enumerations, [103](#)
- TimerTriggerSource_Line0
 - Camera Enumerations, [103](#)
- TimerTriggerSource_Line1
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Line2
 - Camera Enumerations, [104](#)
- TimerTriggerSource_LineEnd
 - Camera Enumerations, [103](#)
- TimerTriggerSource_LineStart
 - Camera Enumerations, [103](#)
- TimerTriggerSource_LineTrigger
 - Camera Enumerations, [103](#)
- TimerTriggerSource_LinkTrigger0
 - Camera Enumerations, [104](#)
- TimerTriggerSource_LinkTrigger1
 - Camera Enumerations, [104](#)
- TimerTriggerSource_LinkTrigger2
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Off
 - Camera Enumerations, [103](#)
- TimerTriggerSource_SoftwareSignal0
 - Camera Enumerations, [104](#)
- TimerTriggerSource_SoftwareSignal1
 - Camera Enumerations, [104](#)
- TimerTriggerSource_SoftwareSignal2
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Timer0End
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Timer0Start
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Timer1End
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Timer1Start
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Timer2End
 - Camera Enumerations, [104](#)
- TimerTriggerSource_Timer2Start
 - Camera Enumerations, [104](#)
- TimerTriggerSource_UserOutput0
 - Camera Enumerations, [104](#)
- TimerTriggerSource_UserOutput1
 - Camera Enumerations, [104](#)
- TimerTriggerSource_UserOutput2
 - Camera Enumerations, [104](#)
- TimerValue
 - quickSpin, [321](#)
- Timestamp

- quickSpin, [321](#)
- TimestampLatch
 - quickSpin, [321](#)
- TimestampLatchValue
 - quickSpin, [321](#)
- TimestampReset
 - quickSpin, [321](#)
- TransferAbort
 - quickSpin, [321](#)
- TransferBlockCount
 - quickSpin, [321](#)
- TransferBurstCount
 - quickSpin, [321](#)
- TransferComponentSelector
 - quickSpin, [321](#)
- TransferComponentSelector_All
 - Camera Enumerations, [105](#)
- TransferComponentSelector_Blue
 - Camera Enumerations, [105](#)
- TransferComponentSelector_Green
 - Camera Enumerations, [105](#)
- TransferComponentSelector_Red
 - Camera Enumerations, [105](#)
- TransferControlMode
 - quickSpin, [321](#)
- TransferControlMode_Automatic
 - Camera Enumerations, [105](#)
- TransferControlMode_Basic
 - Camera Enumerations, [105](#)
- TransferControlMode_UserControlled
 - Camera Enumerations, [105](#)
- TransferOperationMode
 - quickSpin, [321](#)
- TransferOperationMode_Continuous
 - Camera Enumerations, [105](#)
- TransferOperationMode_MultiBlock
 - Camera Enumerations, [105](#)
- TransferPause
 - quickSpin, [321](#)
- TransferQueueCurrentBlockCount
 - quickSpin, [321](#)
- TransferQueueMaxBlockCount
 - quickSpin, [321](#)
- TransferQueueMode
 - quickSpin, [321](#)
- TransferQueueMode_FirstInFirstOut
 - Camera Enumerations, [105](#)
- TransferQueueOverflowCount
 - quickSpin, [321](#)
- TransferResume
 - quickSpin, [321](#)
- TransferSelector
 - quickSpin, [321](#)
- TransferSelector_All
 - Camera Enumerations, [106](#)
- TransferSelector_Stream0
 - Camera Enumerations, [106](#)
- TransferSelector_Stream1
 - Camera Enumerations, [106](#)
- TransferSelector_Stream2
 - Camera Enumerations, [106](#)
- TransferStart
 - quickSpin, [321](#)
- TransferStatus
 - quickSpin, [322](#)
- TransferStatusSelector
 - quickSpin, [322](#)
- TransferStatusSelector_Paused
 - Camera Enumerations, [106](#)
- TransferStatusSelector_QueueOverflow
 - Camera Enumerations, [106](#)
- TransferStatusSelector_Stopped
 - Camera Enumerations, [106](#)
- TransferStatusSelector_Stopping
 - Camera Enumerations, [106](#)
- TransferStatusSelector_Streaming
 - Camera Enumerations, [106](#)
- TransferStop
 - quickSpin, [322](#)
- TransferStreamChannel
 - quickSpin, [322](#)
- TransferTriggerActivation
 - quickSpin, [322](#)
- TransferTriggerActivation_AnyEdge
 - Camera Enumerations, [106](#)
- TransferTriggerActivation_FallingEdge
 - Camera Enumerations, [106](#)
- TransferTriggerActivation_LevelHigh
 - Camera Enumerations, [106](#)
- TransferTriggerActivation_LevelLow
 - Camera Enumerations, [106](#)
- TransferTriggerActivation_RisingEdge
 - Camera Enumerations, [106](#)
- TransferTriggerMode
 - quickSpin, [322](#)
- TransferTriggerMode_Off
 - Camera Enumerations, [107](#)
- TransferTriggerMode_On
 - Camera Enumerations, [107](#)
- TransferTriggerSelector
 - quickSpin, [322](#)
- TransferTriggerSelector_TransferAbort
 - Camera Enumerations, [107](#)
- TransferTriggerSelector_TransferActive
 - Camera Enumerations, [107](#)
- TransferTriggerSelector_TransferBurstStart
 - Camera Enumerations, [107](#)
- TransferTriggerSelector_TransferBurstStop
 - Camera Enumerations, [107](#)
- TransferTriggerSelector_TransferPause
 - Camera Enumerations, [107](#)
- TransferTriggerSelector_TransferResume
 - Camera Enumerations, [107](#)
- TransferTriggerSelector_TransferStart
 - Camera Enumerations, [107](#)
- TransferTriggerSelector_TransferStop
 - Camera Enumerations, [107](#)

- Camera Enumerations, [107](#)
- TransferTriggerSource
 - quickSpin, [322](#)
- TransferTriggerSource_Action0
 - Camera Enumerations, [108](#)
- TransferTriggerSource_Action1
 - Camera Enumerations, [108](#)
- TransferTriggerSource_Action2
 - Camera Enumerations, [108](#)
- TransferTriggerSource_Counter0End
 - Camera Enumerations, [107](#)
- TransferTriggerSource_Counter0Start
 - Camera Enumerations, [107](#)
- TransferTriggerSource_Counter1End
 - Camera Enumerations, [107](#)
- TransferTriggerSource_Counter1Start
 - Camera Enumerations, [107](#)
- TransferTriggerSource_Counter2End
 - Camera Enumerations, [108](#)
- TransferTriggerSource_Counter2Start
 - Camera Enumerations, [107](#)
- TransferTriggerSource_Line0
 - Camera Enumerations, [107](#)
- TransferTriggerSource_Line1
 - Camera Enumerations, [107](#)
- TransferTriggerSource_Line2
 - Camera Enumerations, [107](#)
- TransferTriggerSource_SoftwareSignal0
 - Camera Enumerations, [108](#)
- TransferTriggerSource_SoftwareSignal1
 - Camera Enumerations, [108](#)
- TransferTriggerSource_SoftwareSignal2
 - Camera Enumerations, [108](#)
- TransferTriggerSource_Timer0End
 - Camera Enumerations, [108](#)
- TransferTriggerSource_Timer0Start
 - Camera Enumerations, [108](#)
- TransferTriggerSource_Timer1End
 - Camera Enumerations, [108](#)
- TransferTriggerSource_Timer1Start
 - Camera Enumerations, [108](#)
- TransferTriggerSource_Timer2End
 - Camera Enumerations, [108](#)
- TransferTriggerSource_Timer2Start
 - Camera Enumerations, [108](#)
- Transport Layer Enumerations, [272](#)
 - DeviceAccessStatus_NoAccess, [273](#)
 - DeviceAccessStatus_ReadOnly, [273](#)
 - DeviceAccessStatus_ReadWrite, [273](#)
 - DeviceAccessStatus_Unknown, [273](#)
 - DeviceCurrentSpeed_FullSpeed, [274](#)
 - DeviceCurrentSpeed_HighSpeed, [274](#)
 - DeviceCurrentSpeed_LowSpeed, [274](#)
 - DeviceCurrentSpeed_SuperSpeed, [274](#)
 - DeviceCurrentSpeed_UnknownSpeed, [274](#)
 - DeviceEndiannessMechanism_Legacy, [274](#)
 - DeviceEndiannessMechanism_Standard, [274](#)
 - DeviceType_CLHS, [274](#)
 - DeviceType_CXP, [274](#)
 - DeviceType_CL, [274](#)
 - DeviceType_Custom, [274](#)
 - DeviceType_ETHERNET, [274](#)
 - DeviceType_GEV, [274](#)
 - DeviceType_IIDC, [274](#)
 - DeviceType_Mixed, [274](#)
 - DeviceType_PCI, [274](#)
 - DeviceType_U3V, [274](#)
 - DeviceType_UVC, [274](#)
 - GUIXMLLocation_Device, [275](#)
 - GUIXMLLocation_Host, [275](#)
 - GenICamXMLLocation_Device, [275](#)
 - GenICamXMLLocation_Host, [275](#)
 - GevCCP_EnumEntry_GevCCP_ControlAccess, [275](#)
 - GevCCP_EnumEntry_GevCCP_ExclusiveAccess, [275](#)
 - GevCCP_EnumEntry_GevCCP_OpenAccess, [275](#)
 - NUMDEVICEACCESSSTATUS, [273](#)
 - NUMDEVICECURRENTSPEED, [274](#)
 - NUMDEVICEENDIANESSMECHANISM, [274](#)
 - NUMDEVICETYPE, [274](#)
 - NUMGENICAMXMLLOCATION, [275](#)
 - NUMGEVCCP, [275](#)
 - NUMGUIXMLLOCATION, [275](#)
 - NUMPOESTATUS, [275](#)
 - NUMSTREAMBUFFERCOUNTMODE, [276](#)
 - NUMSTREAMBUFFERHANDLINGMODE, [276](#)
 - NUMSTREAMDEFAULTBUFFERCOUNTMODE, [276](#)
 - NUMSTREAMTYPE, [277](#)
 - POEStatus_NotSupported, [275](#)
 - POEStatus_PowerOff, [275](#)
 - POEStatus_PowerOn, [275](#)
 - spinTLDeviceAccessStatusEnums, [273](#)
 - spinTLDeviceCurrentSpeedEnums, [273](#)
 - spinTLDeviceEndiannessMechanismEnums, [274](#)
 - spinTLDeviceTypeEnums, [274](#)
 - spinTLGUIXMLLocationEnums, [275](#)
 - spinTLGenICamXMLLocationEnums, [274](#)
 - spinTLGevCCPEnums, [275](#)
 - spinTLPOEStatusEnums, [275](#)
 - spinTLStreamBufferCountModeEnums, [275](#)
 - spinTLStreamBufferHandlingModeEnums, [276](#)
 - spinTLStreamDefaultBufferCountModeEnums, [276](#)
 - spinTLStreamTypeEnums, [276](#)
 - StreamBufferCountMode_Auto, [276](#)
 - StreamBufferCountMode_Manual, [276](#)
 - StreamBufferHandlingMode_NewestFirst, [276](#)
 - StreamBufferHandlingMode_NewestFirstOverwrite, [276](#)
 - StreamBufferHandlingMode_NewestOnly, [276](#)
 - StreamBufferHandlingMode_OldestFirst, [276](#)
 - StreamBufferHandlingMode_OldestFirstOverwrite, [276](#)
 - StreamDefaultBufferCountMode_Auto, [276](#)
 - StreamDefaultBufferCountMode_Manual, [276](#)

- StreamType_CLHS, [277](#)
- StreamType_CXP, [277](#)
- StreamType_CL, [277](#)
- StreamType_Custom, [277](#)
- StreamType_ETHERNET, [277](#)
- StreamType_GEV, [277](#)
- StreamType_IIDC, [277](#)
- StreamType_Mixed, [277](#)
- StreamType_PCI, [277](#)
- StreamType_U3V, [277](#)
- StreamType_UVC, [277](#)
- TriggerActivation
 - quickSpin, [322](#)
- TriggerActivation_AnyEdge
 - Camera Enumerations, [108](#)
- TriggerActivation_FallingEdge
 - Camera Enumerations, [108](#)
- TriggerActivation_LevelHigh
 - Camera Enumerations, [108](#)
- TriggerActivation_LevelLow
 - Camera Enumerations, [108](#)
- TriggerActivation_RisingEdge
 - Camera Enumerations, [108](#)
- TriggerDelay
 - quickSpin, [322](#)
- TriggerDivider
 - quickSpin, [322](#)
- TriggerEventTest
 - quickSpin, [322](#)
- TriggerMode
 - quickSpin, [322](#)
- TriggerMode_Off
 - Camera Enumerations, [108](#)
- TriggerMode_On
 - Camera Enumerations, [108](#)
- TriggerMultiplier
 - quickSpin, [322](#)
- TriggerOverlap
 - quickSpin, [322](#)
- TriggerOverlap_Off
 - Camera Enumerations, [109](#)
- TriggerOverlap_PreviousFrame
 - Camera Enumerations, [109](#)
- TriggerOverlap_ReadOut
 - Camera Enumerations, [109](#)
- TriggerSelector
 - quickSpin, [322](#)
- TriggerSelector_AcquisitionStart
 - Camera Enumerations, [109](#)
- TriggerSelector_FrameBurstStart
 - Camera Enumerations, [109](#)
- TriggerSelector_FrameStart
 - Camera Enumerations, [109](#)
- TriggerSoftware
 - quickSpin, [322](#)
- TriggerSource
 - quickSpin, [322](#)
- TriggerSource_Action0
 - Camera Enumerations, [109](#)
- TriggerSource_Counter0End
 - Camera Enumerations, [109](#)
- TriggerSource_Counter0Start
 - Camera Enumerations, [109](#)
- TriggerSource_Counter1End
 - Camera Enumerations, [109](#)
- TriggerSource_Counter1Start
 - Camera Enumerations, [109](#)
- TriggerSource_Line0
 - Camera Enumerations, [109](#)
- TriggerSource_Line1
 - Camera Enumerations, [109](#)
- TriggerSource_Line2
 - Camera Enumerations, [109](#)
- TriggerSource_Line3
 - Camera Enumerations, [109](#)
- TriggerSource_LogicBlock0
 - Camera Enumerations, [109](#)
- TriggerSource_LogicBlock1
 - Camera Enumerations, [109](#)
- TriggerSource_Software
 - Camera Enumerations, [109](#)
- TriggerSource_UserOutput0
 - Camera Enumerations, [109](#)
- TriggerSource_UserOutput1
 - Camera Enumerations, [109](#)
- TriggerSource_UserOutput2
 - Camera Enumerations, [109](#)
- TriggerSource_UserOutput3
 - Camera Enumerations, [109](#)
- True
 - Spinnaker C Definitions, [12](#)
- type
 - spinLibraryVersion, [337](#)
- UNKNOWN_PIXELFORMAT
 - Camera Enumerations, [88](#)
- USB
 - Spinnaker C GenICam Enumerations, [268](#)
- UnknownNode
 - Spinnaker C GenICam Enumerations, [267](#)
- Unsigned
 - Spinnaker C GenICam Enumerations, [268](#)
- UserOutputSelector
 - quickSpin, [322](#)
- UserOutputSelector_UserOutput0
 - Camera Enumerations, [110](#)
- UserOutputSelector_UserOutput1
 - Camera Enumerations, [110](#)
- UserOutputSelector_UserOutput2
 - Camera Enumerations, [110](#)
- UserOutputSelector_UserOutput3
 - Camera Enumerations, [110](#)
- UserOutputValue
 - quickSpin, [322](#)
- UserOutputValueAll
 - quickSpin, [322](#)
- UserOutputValueAllMask

- quickSpin, [322](#)
- UserSetDefault
 - quickSpin, [322](#)
- UserSetDefault_Default
 - Camera Enumerations, [110](#)
- UserSetDefault_UserSet0
 - Camera Enumerations, [110](#)
- UserSetDefault_UserSet1
 - Camera Enumerations, [110](#)
- UserSetFeatureEnable
 - quickSpin, [323](#)
- UserSetLoad
 - quickSpin, [323](#)
- UserSetSave
 - quickSpin, [323](#)
- UserSetSelector
 - quickSpin, [323](#)
- UserSetSelector_Default
 - Camera Enumerations, [110](#)
- UserSetSelector_UserSet0
 - Camera Enumerations, [110](#)
- UserSetSelector_UserSet1
 - Camera Enumerations, [110](#)
- V3_3Enable
 - quickSpin, [323](#)
- ValueNode
 - Spinnaker C GenICam Enumerations, [267](#)
- Varying
 - Spinnaker C GenICam Enumerations, [268](#)
- WEIGHTED_DIRECTIONAL_FILTER
 - Spinnaker C Enumerations, [209](#)
- WhiteClip
 - quickSpin, [323](#)
- WhiteClipSelector
 - quickSpin, [323](#)
- WhiteClipSelector_All
 - Camera Enumerations, [110](#)
- WhiteClipSelector_Blue
 - Camera Enumerations, [110](#)
- WhiteClipSelector_Green
 - Camera Enumerations, [110](#)
- WhiteClipSelector_Red
 - Camera Enumerations, [110](#)
- WhiteClipSelector_Tap1
 - Camera Enumerations, [110](#)
- WhiteClipSelector_Tap2
 - Camera Enumerations, [110](#)
- WhiteClipSelector_U
 - Camera Enumerations, [110](#)
- WhiteClipSelector_V
 - Camera Enumerations, [110](#)
- WhiteClipSelector_Y
 - Camera Enumerations, [110](#)
- Width
 - quickSpin, [323](#)
- width
 - spinH264Option, [334](#)
- WidthMax
 - quickSpin, [323](#)
- WO
 - Spinnaker C GenICam Enumerations, [264](#)
- WriteAround
 - Spinnaker C GenICam Enumerations, [265](#)
- WriteThrough
 - Spinnaker C GenICam Enumerations, [265](#)
- xvAll
 - Spinnaker C GenICam Enumerations, [269](#)
- xvCycles
 - Spinnaker C GenICam Enumerations, [269](#)
- xvDefault
 - Spinnaker C GenICam Enumerations, [269](#)
- xvLoad
 - Spinnaker C GenICam Enumerations, [269](#)
- xvSFNC
 - Spinnaker C GenICam Enumerations, [269](#)
- Yes
 - Spinnaker C GenICam Enumerations, [269](#)