

First Light Imaging FliSdk
2.9.x

Generated by Doxygen 1.8.16

1 Namespace Index	1
1.1 Namespace List	1
2 Hierarchical Index	3
2.1 Class Hierarchy	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Namespace Documentation	9
5.1 FliCblueOneEnum Namespace Reference	9
5.1.1 Enumeration Type Documentation	10
5.1.1.1 ConversionEfficiencyEnum	10
5.1.1.2 DeviceFanModeEnum	10
5.1.1.3 DeviceTecSelectorEnum	11
5.1.1.4 DeviceTemperatureSelectorEnum	11
5.1.1.5 FirmwareUpdateStatusEnum	11
5.1.1.6 GlowReductionEnum	11
5.1.1.7 IPModeEnum	12
5.1.1.8 LogCollectStatusEnum	12
5.1.1.9 SparseModeEnum	12
5.1.1.10 SparseSelectorEnum	12
5.1.1.11 TestPatternEnum	13
5.1.1.12 TestPatternGeneratorSelectorEnum	13
5.1.1.13 UserSetDefaultEnum	13
5.1.1.14 UserSetSelectorEnum	14
5.1.2 Variable Documentation	14
5.1.2.1 ConversionEfficiencyStringToValue	14
5.1.2.2 DeviceFanModeStringToValue	15
5.1.2.3 DeviceTecSelectorStringToValue	15
5.1.2.4 DeviceTemperatureSelectorStringToValue	15
5.1.2.5 FirmwareUpdateStatusStringToValue	15
5.1.2.6 GlowReductionStringToValue	16
5.1.2.7 IPModeStringToValue	16
5.1.2.8 LogCollectStatusStringToValue	16
5.1.2.9 SparseModeStringToValue	16
5.1.2.10 SparseSelectorStringToValue	17
5.1.2.11 TestPatternGeneratorSelectorStringToValue	17
5.1.2.12 TestPatternStringToValue	17
5.1.2.13 UserSetDefaultStringToValue	17
5.1.2.14 UserSetSelectorStringToValue	18

5.2 FliCblueSfncEnum Namespace Reference	18
5.2.1 Enumeration Type Documentation	19
5.2.1.1 AcquisitionModeEnum	19
5.2.1.2 BlackLevelAutoEnum	19
5.2.1.3 BlackLevelSelectorEnum	20
5.2.1.4 CxpConnectionTestModeEnum	20
5.2.1.5 CxpErrorCounterSelectorEnum	20
5.2.1.6 CxpErrorCounterStatusEnum	20
5.2.1.7 CxpLinkConfigurationEnum	21
5.2.1.8 CxpLinkConfigurationPreferredEnum	21
5.2.1.9 CxpLinkConfigurationStatusEnum	21
5.2.1.10 CxpSendReceiveSelectorEnum	21
5.2.1.11 DeviceIndicatorModeEnum	22
5.2.1.12 DeviceScanTypeEnum	22
5.2.1.13 ExposureModeEnum	22
5.2.1.14 GainSelectorEnum	22
5.2.1.15 PixelFormatEnum	23
5.2.1.16 RegionDestinationEnum	23
5.2.1.17 RegionModeEnum	23
5.2.1.18 RegionSelectorEnum	23
5.2.1.19 SensorShutterModeEnum	24
5.2.2 Variable Documentation	24
5.2.2.1 AcquisitionModeString	24
5.2.2.2 BlackLevelAutoString	24
5.2.2.3 BlackLevelSelectorString	24
5.2.2.4 CxpConnectionTestModeString	25
5.2.2.5 CxpErrorCounterSelectorString	25
5.2.2.6 CxpErrorCounterStatusString	25
5.2.2.7 CxpLinkConfigurationPreferredString	25
5.2.2.8 CxpLinkConfigurationStatusString	26
5.2.2.9 CxpLinkConfigurationString	26
5.2.2.10 CxpSendReceiveSelectorString	26
5.2.2.11 DeviceIndicatorModeString	26
5.2.2.12 DeviceScanTypeString	27
5.2.2.13 ExposureModeString	27
5.2.2.14 featuresListString	27
5.2.2.15 GainSelectorString	27
5.2.2.16 PixelFormatString	27
5.2.2.17 RegionDestinationString	28
5.2.2.18 RegionModeString	28
5.2.2.19 RegionSelectorString	28
5.2.2.20 SensorShutterModeString	28

5.3 FliCblueTwoEnum Namespace Reference	28
5.3.1 Enumeration Type Documentation	29
5.3.1.1 BinningHorizontalModeEnum	29
5.3.1.2 BinningSelectorEnum	29
5.3.1.3 BinningVerticalModeEnum	29
5.3.1.4 FirmwareUpdateStatusEnum	30
5.3.2 Variable Documentation	30
5.3.2.1 BinningHorizontalModeStringToValue	30
5.3.2.2 BinningSelectorStringToValue	30
5.3.2.3 BinningVerticalModeStringToValue	30
5.3.2.4 FirmwareUpdateStatusStringToValue	31
6 Class Documentation	33
6.1 FliCblueOne Class Reference	33
6.1.1 Constructor & Destructor Documentation	35
6.1.1.1 FliCblueOne()	36
6.1.2 Member Data Documentation	36
6.1.2.1 AcquisitionFrameRateMaxReg	36
6.1.2.2 AcquisitionFrameRateMinReg	36
6.1.2.3 ConversionEfficiency	36
6.1.2.4 CurrentIPAddress	36
6.1.2.5 CurrentSubnetMask	36
6.1.2.6 DeviceCoolingEnable	37
6.1.2.7 DeviceCoolingSetpoint	37
6.1.2.8 DeviceFanMode	37
6.1.2.9 DeviceFanSpeed	37
6.1.2.10 DeviceShutdown	37
6.1.2.11 DeviceStatus	37
6.1.2.12 DeviceStatusDetailed	38
6.1.2.13 DeviceTecCurrent	38
6.1.2.14 DeviceTecPower	38
6.1.2.15 DeviceTecSelector	38
6.1.2.16 DeviceTecVoltage	38
6.1.2.17 DeviceTemperatureSelector	38
6.1.2.18 ExposureTimeMaxReg	39
6.1.2.19 ExposureTimeMinReg	39
6.1.2.20 FirmwareUpdateAbort	39
6.1.2.21 FirmwareUpdateExecute	39
6.1.2.22 FirmwareUpdateStatus	39
6.1.2.23 FirmwareUpdateStatusRefresh	39
6.1.2.24 FirmwareUpdateUri	40
6.1.2.25 GlowReduction	40

6.1.2.26 IPMode	40
6.1.2.27 IPReconfigure	40
6.1.2.28 LogCollect	40
6.1.2.29 LogCollectAbort	40
6.1.2.30 LogCollectStatus	41
6.1.2.31 LogCollectStatusRefresh	41
6.1.2.32 LogHistoryDepth	41
6.1.2.33 LogServe	41
6.1.2.34 LogServeAbort	41
6.1.2.35 LogServeUri	41
6.1.2.36 Sparse	42
6.1.2.37 SparseHeight	42
6.1.2.38 SparseMode	42
6.1.2.39 SparseOffsetX	42
6.1.2.40 SparseOffsetY	42
6.1.2.41 SparseSelector	42
6.1.2.42 SparseWidth	43
6.1.2.43 StaticAlternateDomainNameServer	43
6.1.2.44 StaticDefaultGateway	43
6.1.2.45 StaticDomainNameServer	43
6.1.2.46 StaticIPAddress	43
6.1.2.47 StaticSubnetMask	43
6.1.2.48 TestPattern	44
6.1.2.49 TestPatternGeneratorSelector	44
6.1.2.50 UserSetDefault	44
6.1.2.51 UserSetSelector	44
6.2 FliCblueTwo Class Reference	44
6.2.1 Constructor & Destructor Documentation	45
6.2.1.1 FliCblueTwo()	45
6.2.2 Member Data Documentation	45
6.2.2.1 BinningHorizontal	45
6.2.2.2 BinningHorizontalMode	45
6.2.2.3 BinningVertical	46
6.2.2.4 BinningVerticalMode	46
6.2.2.5 FirmwareUpdateStatus	46
6.3 FliCred Class Reference	46
6.3.1 Detailed Description	48
6.3.2 Constructor & Destructor Documentation	48
6.3.2.1 FliCred()	49
6.3.3 Member Function Documentation	49
6.3.3.1 buildBias()	49
6.3.3.2 buildFlat()	49

6.3.3.3 continueStarting()	49
6.3.3.4 enableCropping()	50
6.3.3.5 enableEvents()	50
6.3.3.6 enableExtSynchro()	50
6.3.3.7 enableImageTags()	50
6.3.3.8 enableLed()	50
6.3.3.9 enableTelnet()	50
6.3.3.10 getAduOffset()	50
6.3.3.11 getBadPixelModeOnOff()	51
6.3.3.12 getBiasState()	51
6.3.3.13 getCameraType()	51
6.3.3.14 getCheckTag4by4()	51
6.3.3.15 getEventsState()	51
6.3.3.16 getExcludeBorder()	51
6.3.3.17 getExtSynchroState()	51
6.3.3.18 getFilteringModeOnOff()	52
6.3.3.19 getFlatState()	52
6.3.3.20 getHwuid()	52
6.3.3.21 getImageTagsState()	52
6.3.3.22 getIpConfig()	52
6.3.3.23 getIsSlowMode()	52
6.3.3.24 getKindOfBadPixelCorrection()	52
6.3.3.25 getLedState()	53
6.3.3.26 getLogs() [1/2]	53
6.3.3.27 getLogs() [2/2]	53
6.3.3.28 getPassword()	54
6.3.3.29 getStatus()	54
6.3.3.30 getStatusDetailed()	54
6.3.3.31 getThreholdingOnOff()	54
6.3.3.32 getUserConvolutionMatrix()	54
6.3.3.33 getUserConvolutionMatrixIndex_V2()	55
6.3.3.34 getVersionFirmware()	55
6.3.3.35 getVersionFirmwareBuild()	55
6.3.3.36 getVersionFirmwareDetailed()	55
6.3.3.37 getVersionFpga()	55
6.3.3.38 getVersionHardware()	56
6.3.3.39 getVersions()	56
6.3.3.40 restoreFactory()	56
6.3.3.41 saveCameraSettings()	56
6.3.3.42 sendBiasFile()	56
6.3.3.43 sendBiasFromUrl()	57
6.3.3.44 sendFlatFile()	57

6.3.3.45 sendFlatFromUrl()	57
6.3.3.46 setAduOffset()	58
6.3.3.47 setBadPixelModeOnOff()	58
6.3.3.48 setExcludeBorderOnOff()	58
6.3.3.49 setFilteringModeOnOff()	58
6.3.3.50 setIpAddress()	58
6.3.3.51 setIpAlternateDns()	59
6.3.3.52 setIpAutomatic()	59
6.3.3.53 setIpDns()	59
6.3.3.54 setIpGateway()	59
6.3.3.55 setIpManual()	59
6.3.3.56 setIpNetmask()	59
6.3.3.57 setIpRefresh()	59
6.3.3.58 setKindOfBadPixelCorrection()	59
6.3.3.59 setPassword()	60
6.3.3.60 setSlowMode()	60
6.3.3.61 setThresholdingLevelsValues()	60
6.3.3.62 setThresholdingOnOff()	61
6.3.3.63 setUserConvolutionMatrix()	61
6.3.3.64 setUserConvolutionMatrixIndex_V2()	61
6.3.3.65 shutDown()	61
6.3.3.66 upgradeFirmware()	62
6.4 FliCredOne Class Reference	62
6.4.1 Detailed Description	64
6.4.2 Member Enumeration Documentation	64
6.4.2.1 Mode	64
6.4.3 Constructor & Destructor Documentation	64
6.4.3.1 FliCredOne()	64
6.4.4 Member Function Documentation	64
6.4.4.1 enableCooling()	65
6.4.4.2 enableFowler()	65
6.4.4.3 enableRawImages()	65
6.4.4.4 enableRemoteMaintenance()	65
6.4.4.5 enableStandby()	65
6.4.4.6 enableTestPattern()	65
6.4.4.7 getAll()	65
6.4.4.8 getAllTemp()	66
6.4.4.9 getCoolingState()	66
6.4.4.10 getCropping()	66
6.4.4.11 getFowlerState()	66
6.4.4.12 getGain()	66
6.4.4.13 getNbReadWoReset()	66

6.4.4.14 getNbRegenGetter()	67
6.4.4.15 getNbSamplePixel()	67
6.4.4.16 getNloop()	67
6.4.4.17 getPhotoCurrent()	67
6.4.4.18 getPowerGetter()	67
6.4.4.19 getPowerPulseTube()	67
6.4.4.20 getPowers()	67
6.4.4.21 getPressure()	68
6.4.4.22 getPulseTubeReady()	68
6.4.4.23 getRawImagesState()	68
6.4.4.24 getReadOutMode()	68
6.4.4.25 getRegenRemainingTime()	68
6.4.4.26 getRemoteMaintenanceState()	68
6.4.4.27 getResetWidth()	68
6.4.4.28 getStandbyState()	69
6.4.4.29 getTelnetState()	69
6.4.4.30 getTempDiode()	69
6.4.4.31 getTempFrontEnd()	69
6.4.4.32 getTempMotherBoard()	69
6.4.4.33 getTempPowerBoard()	69
6.4.4.34 getTempPtController()	69
6.4.4.35 getTempPtMcu()	70
6.4.4.36 getTempSetpoint()	70
6.4.4.37 getTempWater()	70
6.4.4.38 getTestPatternState()	70
6.4.4.39 getVersionFpgaDetailed()	70
6.4.4.40 isCroppingValid()	70
6.4.4.41 reboot()	71
6.4.4.42 sendTestPatternFromUrl()	71
6.4.4.43 setCropping()	71
6.4.4.44 setCroppingColumns()	71
6.4.4.45 setCroppingRows()	72
6.4.4.46 setGain()	72
6.4.4.47 setMode()	72
6.4.4.48 setNbReadWoReset()	72
6.4.4.49 setNloop()	72
6.4.4.50 setNsamplePixel()	72
6.4.4.51 setResetWidth()	72
6.4.4.52 startVacuumRegen()	73
6.5 FliCredThree Class Reference	73
6.5.1 Detailed Description	76
6.5.2 Member Enumeration Documentation	77

6.5.2.1 AgcParam	77
6.5.3 Constructor & Destructor Documentation	77
6.5.3.1 FliCredThree()	77
6.5.4 Member Function Documentation	77
6.5.4.1 abortBuildNuc()	77
6.5.4.2 buildBiasNuc()	78
6.5.4.3 buildFlatHdrC1()	78
6.5.4.4 buildFlatHdrC1Nuc()	78
6.5.4.5 buildFlatHdrC2()	78
6.5.4.6 buildFlatHdrC2Nuc()	78
6.5.4.7 buildFlatNuc()	78
6.5.4.8 deleteLicense()	78
6.5.4.9 disableLicense()	79
6.5.4.10 enableAdaptBias()	79
6.5.4.11 enableAgc()	79
6.5.4.12 enableAntiBlooming()	79
6.5.4.13 enableBadPixel()	79
6.5.4.14 enableHdr()	79
6.5.4.15 enableHdrExtended()	79
6.5.4.16 enableLicense()	80
6.5.4.17 enableRawImages()	80
6.5.4.18 enableRemoteMaintenance()	80
6.5.4.19 enableSwSynchro()	80
6.5.4.20 enableTcdsAdjust()	80
6.5.4.21 enableTintGranularity()	80
6.5.4.22 enableUnsignedPixels()	80
6.5.4.23 enableVrefAdjust()	81
6.5.4.24 getAccumulatedUptime()	81
6.5.4.25 getAdaptBiasState()	81
6.5.4.26 getAgcParam()	81
6.5.4.27 getAgcPriority()	81
6.5.4.28 getAgcRoi()	81
6.5.4.29 getAgcState()	82
6.5.4.30 getAllTemp()	82
6.5.4.31 getAntiBloomingState()	82
6.5.4.32 getBadPixelState()	82
6.5.4.33 getBuildNucProgress()	82
6.5.4.34 getConversionGain()	82
6.5.4.35 getCropping()	83
6.5.4.36 getDarkOptimLevel()	83
6.5.4.37 getDate()	83
6.5.4.38 getExtMarkerSource()	83

6.5.4.39	getExtSynchroExposure()	83
6.5.4.40	getExtSynchroPolarity()	83
6.5.4.41	getFactoryBadPixelMap()	84
6.5.4.42	getHardwareFeatures()	84
6.5.4.43	getHdrCalibrationMode()	84
6.5.4.44	getHdrExtendedState()	84
6.5.4.45	getHdrState()	84
6.5.4.46	getImagePattern()	84
6.5.4.47	getIpAddress()	84
6.5.4.48	getIpAlternateDns()	85
6.5.4.49	getIpDns()	85
6.5.4.50	getIpGateway()	85
6.5.4.51	getIpMode()	85
6.5.4.52	getIpNetmask()	85
6.5.4.53	getLicenses()	85
6.5.4.54	getMaxFpsUsb()	85
6.5.4.55	getMaxSyncDelay()	86
6.5.4.56	getMaxTintItr()	86
6.5.4.57	getMinFps()	86
6.5.4.58	getMinSyncDelay()	86
6.5.4.59	getNbFramesPerSwTrig()	86
6.5.4.60	getPreset()	86
6.5.4.61	getRawImagesState()	86
6.5.4.62	getRemoteMaintenanceState()	87
6.5.4.63	getSnakeParam()	87
6.5.4.64	getSoftwareFeatures()	87
6.5.4.65	getStepSyncDelay()	87
6.5.4.66	getSwSynchroState()	87
6.5.4.67	getSyncDelay()	87
6.5.4.68	getSyncSignalSource()	87
6.5.4.69	getTcdsAdjustState()	88
6.5.4.70	getTelnetState()	88
6.5.4.71	getTempAmbiant()	88
6.5.4.72	getTempBackEnd()	88
6.5.4.73	getTempCpu()	88
6.5.4.74	getTempInterface()	88
6.5.4.75	getTempSnake()	88
6.5.4.76	getTint()	89
6.5.4.77	getTintGranularityState()	89
6.5.4.78	getTintRange()	89
6.5.4.79	getTintStep()	89
6.5.4.80	getTlsydel()	89

6.5.4.81 getTotalUptime()	89
6.5.4.82 getTriggerSource()	89
6.5.4.83 getTuning()	90
6.5.4.84 getUnsignedPixelsState()	90
6.5.4.85 getUploadFirmwareConnectionInfo()	90
6.5.4.86 getUptime()	90
6.5.4.87 getUserBadPixelMap()	90
6.5.4.88 getVrefAdjustState()	90
6.5.4.89 isCroppingValid()	91
6.5.4.90 reboot()	91
6.5.4.91 sendBadPixelFile()	91
6.5.4.92 sendBadPixelFromUrl()	91
6.5.4.93 sendBiasHdrC1File()	91
6.5.4.94 sendBiasHdrC1FromUrl()	91
6.5.4.95 sendBiasHdrC2File()	91
6.5.4.96 sendBiasHdrC2FromUrl()	92
6.5.4.97 sendFlatHdrC1File()	92
6.5.4.98 sendFlatHdrC1FromUrl()	92
6.5.4.99 sendFlatHdrC2File()	92
6.5.4.100 sendFlatHdrC2FromUrl()	92
6.5.4.101 sendLicenseFile()	92
6.5.4.102 setAgcParam()	92
6.5.4.103 setAgcPriorityNone()	93
6.5.4.104 setAgcPriorityOverExposed()	93
6.5.4.105 setAgcPriorityUnderExposed()	93
6.5.4.106 setAgcRoi()	93
6.5.4.107 setConversionGainHigh()	93
6.5.4.108 setConversionGainLow()	93
6.5.4.109 setConversionGainMedium()	93
6.5.4.110 setCropping()	94
6.5.4.111 setCroppingColumns()	94
6.5.4.112 setCroppingRows()	94
6.5.4.113 setDarkOptimLevel()	94
6.5.4.114 setExtSynchroExposureExternal()	94
6.5.4.115 setExtSynchroExposureInternal()	94
6.5.4.116 setExtSynchroPolarityInverted()	95
6.5.4.117 setExtSynchroPolarityStandard()	95
6.5.4.118 setFactoryBadPixelMap()	95
6.5.4.119 setFrameMarkerSourceCC1()	95
6.5.4.120 setFrameMarkerSourceCC2()	95
6.5.4.121 setFrameMarkerSourceCC3()	95
6.5.4.122 setFrameMarkerSourceCC4()	95

6.5.4.123 setFrameMarkerSourceExternal()	96
6.5.4.124 setHdrCalibrationC1()	96
6.5.4.125 setHdrCalibrationC2()	96
6.5.4.126 setHdrCalibrationOff()	96
6.5.4.127 setImagePatternConstant()	96
6.5.4.128 setImagePatternOff()	96
6.5.4.129 setImagePatternRamp()	96
6.5.4.130 setNbFramesPerSwTrig()	97
6.5.4.131 setPreset()	97
6.5.4.132 setPresetNumber()	97
6.5.4.133 setSnakeParam()	97
6.5.4.134 setSyncDelay()	97
6.5.4.135 setSyncSignalSourceCC1()	97
6.5.4.136 setSyncSignalSourceCC2()	97
6.5.4.137 setSyncSignalSourceCC3()	98
6.5.4.138 setSyncSignalSourceCC4()	98
6.5.4.139 setSyncSignalSourceExternal()	98
6.5.4.140 setTint()	98
6.5.4.141 setTlsyDel()	98
6.5.4.142 setTriggerSourceExternal()	98
6.5.4.143 setTriggerSourceSoftware()	98
6.5.4.144 setTuningGeneralUse()	99
6.5.4.145 setTuningLongExposure()	99
6.5.4.146 setTuningShortExposure()	99
6.5.4.147 setUserBadPixelMap()	99
6.5.4.148 setVoltageVref()	99
6.5.4.149 softwareTrig()	99
6.5.4.150 startEthernetGrabber()	99
6.5.4.151 startHttpServer()	100
6.5.4.152 stopEthernetGrabber()	100
6.5.4.153 stopHttpServer()	100
6.5.4.154 xSendBadPixelFile()	100
6.5.4.155 xSendBiasFile()	100
6.5.4.156 xSendBiasHdrC1File()	100
6.5.4.157 xSendBiasHdrC2File()	100
6.5.4.158 xSendFlatFile()	101
6.5.4.159 xSendFlatHdrC1File()	101
6.5.4.160 xSendFlatHdrC2File()	101
6.5.4.161 xSendLicenseFile()	101
6.6 FliCredTwo Class Reference	101
6.6.1 Detailed Description	105
6.6.2 Constructor & Destructor Documentation	105

6.6.2.1 FliCredTwo()	105
6.6.3 Member Function Documentation	105
6.6.3.1 abortBuildNuc()	105
6.6.3.2 buildBiasNuc()	106
6.6.3.3 buildFlatHdrC1()	106
6.6.3.4 buildFlatHdrC1Nuc()	106
6.6.3.5 buildFlatHdrC2()	106
6.6.3.6 buildFlatHdrC2Nuc()	106
6.6.3.7 buildFlatNuc()	106
6.6.3.8 deleteLicense()	106
6.6.3.9 disableLicense()	107
6.6.3.10 enableAgc()	107
6.6.3.11 enableAntiBlooming()	107
6.6.3.12 enableBadPixel()	107
6.6.3.13 enableFactoryCorrection()	107
6.6.3.14 enableHdr()	107
6.6.3.15 enableHdrExtended()	107
6.6.3.16 enableLicense()	108
6.6.3.17 enableRawImages()	108
6.6.3.18 enableRemoteMaintenance()	108
6.6.3.19 enableSwSynchro()	108
6.6.3.20 enableTcdsAdjust()	108
6.6.3.21 enableTintGranularity()	108
6.6.3.22 enableUnsignedPixels()	108
6.6.3.23 enableVrefAdjust()	109
6.6.3.24 getAccumulatedUptime()	109
6.6.3.25 getAgcPriority()	109
6.6.3.26 getAgcRoi()	109
6.6.3.27 getAgcState()	109
6.6.3.28 getAllTemp()	109
6.6.3.29 getAntiBloomingState()	110
6.6.3.30 getBadPixelState()	110
6.6.3.31 getBuildNucProgress()	110
6.6.3.32 getConversionGain()	110
6.6.3.33 getCropping()	110
6.6.3.34 getDarkOptimLevel()	110
6.6.3.35 getDate()	111
6.6.3.36 getExtMarkerSource()	111
6.6.3.37 getExtSynchroExposure()	111
6.6.3.38 getExtSynchroPolarity()	111
6.6.3.39 getFactoryBadPixelMap()	111
6.6.3.40 getFactoryCorrectionState()	111

6.6.3.41	getFanMode()	111
6.6.3.42	getFanSpeed()	112
6.6.3.43	getHardwareFeatures()	112
6.6.3.44	getHdrCalibrationMode()	112
6.6.3.45	getHdrExtendedState()	112
6.6.3.46	getHdrState()	112
6.6.3.47	getImagePattern()	112
6.6.3.48	getIpAddress()	112
6.6.3.49	getIpAlternateDns()	113
6.6.3.50	getIpDns()	113
6.6.3.51	getIpGateway()	113
6.6.3.52	getIpMode()	113
6.6.3.53	getIpNetmask()	113
6.6.3.54	getLicenses()	113
6.6.3.55	getMaxFpsUsb()	113
6.6.3.56	getMaxSyncDelay()	114
6.6.3.57	getMaxTintItr()	114
6.6.3.58	getMinFps()	114
6.6.3.59	getMinSyncDelay()	114
6.6.3.60	getNbFramesPerSwTrig()	114
6.6.3.61	getNbReadWoReset()	114
6.6.3.62	getPowerExternalPeltier()	114
6.6.3.63	getPowers()	115
6.6.3.64	getPowerSensor()	115
6.6.3.65	getPreset()	115
6.6.3.66	getRawImagesState()	115
6.6.3.67	getRemoteMaintenanceState()	115
6.6.3.68	getSnakeParam()	115
6.6.3.69	getSoftwareFeatures()	116
6.6.3.70	getStepSyncDelay()	116
6.6.3.71	getSwSynchroState()	116
6.6.3.72	getSyncDelay()	116
6.6.3.73	getSynchronization()	116
6.6.3.74	getSyncSignalSource()	116
6.6.3.75	getTcdsAdjustState()	116
6.6.3.76	getTelnetState()	117
6.6.3.77	getTempFrontEnd()	117
6.6.3.78	getTempHeatSink()	117
6.6.3.79	getTempMotherBoard()	117
6.6.3.80	getTempPeltier()	117
6.6.3.81	getTempPowerBoard()	117
6.6.3.82	getTempSnake()	117

6.6.3.83 getTempSnakeSetpoint()	118
6.6.3.84 getTint()	118
6.6.3.85 getTintGranularityState()	118
6.6.3.86 getTintRange()	118
6.6.3.87 getTintStep()	118
6.6.3.88 getTlSydel()	118
6.6.3.89 getTotalUptime()	118
6.6.3.90 getTriggerSource()	119
6.6.3.91 getTuning()	119
6.6.3.92 getUnsignedPixelsState()	119
6.6.3.93 getUploadFirmwareConnectionInfo()	119
6.6.3.94 getUptime()	119
6.6.3.95 getUserBadPixelMap()	119
6.6.3.96 getVoltageVref()	119
6.6.3.97 getVrefAdjustState()	120
6.6.3.98 isCroppingValid()	120
6.6.3.99 reboot()	120
6.6.3.100 sendBadPixelFile()	120
6.6.3.101 sendBadPixelFromUrl()	120
6.6.3.102 sendBiasHdrC1File()	120
6.6.3.103 sendBiasHdrC1FromUrl()	120
6.6.3.104 sendBiasHdrC2File()	121
6.6.3.105 sendBiasHdrC2FromUrl()	121
6.6.3.106 sendFlatHdrC1File()	121
6.6.3.107 sendFlatHdrC1FromUrl()	121
6.6.3.108 sendFlatHdrC2File()	121
6.6.3.109 sendFlatHdrC2FromUrl()	121
6.6.3.110 sendLicenseFile()	121
6.6.3.111 setAgcPriorityNone()	122
6.6.3.112 setAgcPriorityOverExposed()	122
6.6.3.113 setAgcPriorityUnderExposed()	122
6.6.3.114 setAgcRoi()	122
6.6.3.115 setConversionGainHigh()	122
6.6.3.116 setConversionGainLow()	122
6.6.3.117 setConversionGainMedium()	122
6.6.3.118 setCropping()	123
6.6.3.119 setCroppingColumns()	123
6.6.3.120 setCroppingRows()	123
6.6.3.121 setDarkOptimLevel()	123
6.6.3.122 setExtSynchroExposureExternal()	123
6.6.3.123 setExtSynchroExposureInternal()	123
6.6.3.124 setExtSynchroPolarityInverted()	124

6.6.3.125 setExtSynchroPolarityStandard()	124
6.6.3.126 setFactoryBadPixelMap()	124
6.6.3.127 setFanModeAutomatic()	124
6.6.3.128 setFanModeManual()	124
6.6.3.129 setFanSpeed()	124
6.6.3.130 setFrameMarkerSourceCC1()	124
6.6.3.131 setFrameMarkerSourceCC2()	125
6.6.3.132 setFrameMarkerSourceCC3()	125
6.6.3.133 setFrameMarkerSourceCC4()	125
6.6.3.134 setFrameMarkerSourceExternal()	125
6.6.3.135 setHdrCalibrationC1()	125
6.6.3.136 setHdrCalibrationC2()	125
6.6.3.137 setHdrCalibrationOff()	125
6.6.3.138 setImagePatternConstant()	125
6.6.3.139 setImagePatternOff()	126
6.6.3.140 setImagePatternRamp()	126
6.6.3.141 setNbFramesPerSwTrig()	126
6.6.3.142 setNbReadWoReset()	126
6.6.3.143 setPreset()	126
6.6.3.144 setPresetNumber()	126
6.6.3.145 setSensorTemp()	126
6.6.3.146 setSnakeParam()	127
6.6.3.147 setSyncDelay()	127
6.6.3.148 setSynchronizationCmos()	127
6.6.3.149 setSynchronizationFullCmos()	127
6.6.3.150 setSynchronizationLvds()	127
6.6.3.151 setSyncSignalSourceCC1()	127
6.6.3.152 setSyncSignalSourceCC2()	127
6.6.3.153 setSyncSignalSourceCC3()	128
6.6.3.154 setSyncSignalSourceCC4()	128
6.6.3.155 setSyncSignalSourceExternal()	128
6.6.3.156 setTint()	128
6.6.3.157 setTlSyDel()	128
6.6.3.158 setTriggerSourceExternal()	128
6.6.3.159 setTriggerSourceSoftware()	128
6.6.3.160 setTuningGeneralUse()	129
6.6.3.161 setTuningLongExposure()	129
6.6.3.162 setTuningShortExposure()	129
6.6.3.163 setUserBadPixelMap()	129
6.6.3.164 setVoltageVref()	129
6.6.3.165 softwareTrig()	129
6.6.3.166 startEthernetGrabber()	129

6.6.3.167 startHttpServer()	130
6.6.3.168 stopEthernetGrabber()	130
6.6.3.169 stopHttpServer()	130
6.6.3.170 xSendBadPixelFile()	130
6.6.3.171 xSendBiasFile()	130
6.6.3.172 xSendBiasHdrC1File()	130
6.6.3.173 xSendBiasHdrC2File()	130
6.6.3.174 xSendFlatFile()	131
6.6.3.175 xSendFlatHdrC1File()	131
6.6.3.176 xSendFlatHdrC2File()	131
6.6.3.177 xSendLicenseFile()	131
6.7 FliCredTwoLite Class Reference	131
6.7.1 Detailed Description	132
6.7.2 Member Enumeration Documentation	132
6.7.2.1 CoolingMode	132
6.7.3 Constructor & Destructor Documentation	132
6.7.3.1 FliCredTwoLite()	133
6.7.4 Member Function Documentation	133
6.7.4.1 getCoolingFirstPoint()	133
6.7.4.2 getCoolingMode()	133
6.7.4.3 getCoolingState()	133
6.7.4.4 getCoolingStepWidth()	133
6.7.4.5 getCurrentStep()	133
6.7.4.6 getSensorSetpoint()	134
6.7.4.7 getTecPower()	134
6.7.4.8 setCoolingFirstPoint()	134
6.7.4.9 setCoolingMode()	134
6.7.4.10 setCoolingState()	134
6.7.4.11 setCoolingStepWidth()	134
6.7.4.12 setSensorSetpoint()	134
6.8 FliGenicamCamera Class Reference	135
6.8.1 Detailed Description	136
6.8.2 Constructor & Destructor Documentation	136
6.8.2.1 FliGenicamCamera()	136
6.8.2.2 ~FliGenicamCamera()	137
6.8.3 Member Function Documentation	137
6.8.3.1 addCallbackAllRegisters()	137
6.8.3.2 addCallbackDimensionsRegisters()	137
6.8.3.3 executeFeature()	137
6.8.3.4 getAccessMode()	138
6.8.3.5 getAssociatedGrabber()	138
6.8.3.6 getBooleanFeature()	138

6.8.3.7	getCameraModel()	139
6.8.3.8	getDoubleFeature()	139
6.8.3.9	getDoubleIncrementFeature()	139
6.8.3.10	getDoubleMaxFeature()	140
6.8.3.11	getDoubleMinFeature()	140
6.8.3.12	getFeatureLength()	141
6.8.3.13	getFeaturesList()	141
6.8.3.14	getIntegerFeature()	141
6.8.3.15	getIntegerIncrementFeature()	142
6.8.3.16	getIntegerMaxFeature()	142
6.8.3.17	getIntegerMinFeature()	142
6.8.3.18	getPollingInterval()	143
6.8.3.19	getRawData()	143
6.8.3.20	getRepresentation()	143
6.8.3.21	getStringFeature()	144
6.8.3.22	getVisibility()	144
6.8.3.23	removeCallbackAllRegisters()	145
6.8.3.24	setBooleanFeature()	145
6.8.3.25	setDoubleFeature()	145
6.8.3.26	setIntegerFeature()	146
6.8.3.27	setRawData()	146
6.8.3.28	setStringFeature()	146
6.8.4	Member Data Documentation	147
6.8.4.1	_cameraModel	147
6.8.4.2	_grabber	147
6.8.4.3	_stringToFeature	147
6.9	FliOcam2K Class Reference	147
6.9.1	Detailed Description	148
6.9.2	Constructor & Destructor Documentation	149
6.9.2.1	FliOcam2K()	149
6.9.3	Member Function Documentation	149
6.9.3.1	disableCooling()	149
6.9.3.2	enableBias()	149
6.9.3.3	enableFlat()	149
6.9.3.4	getAllTemp()	150
6.9.3.5	getConf()	150
6.9.3.6	getCoolingState()	150
6.9.3.7	getCoolingValue()	150
6.9.3.8	getFps()	150
6.9.3.9	getFpsMax()	151
6.9.3.10	isInterface0()	151
6.9.3.11	isOldFirmware()	151

6.9.3.12	protectionReset()	151
6.9.3.13	resetCoolingAlarm()	151
6.9.3.14	sendBiasFile()	151
6.9.3.15	sendFlatFile()	152
6.9.3.16	sendInterface0Command()	152
6.9.3.17	sendInterface1Command()	152
6.9.3.18	setBiasOffset()	153
6.9.3.19	setCoolingValue()	153
6.9.3.20	setFps()	153
6.9.3.21	setFpsMax()	153
6.9.3.22	setGain()	153
6.9.3.23	setWorkMode()	153
6.9.4	Member Data Documentation	153
6.9.4.1	_conf	153
6.10	FliOcam2S Class Reference	154
6.10.1	Detailed Description	154
6.10.2	Constructor & Destructor Documentation	154
6.10.2.1	FliOcam2S()	155
6.10.3	Member Function Documentation	155
6.10.3.1	enableShutter()	155
6.10.3.2	enableShutterBlockOnRead()	155
6.10.3.3	enableShutterCorrectGlitch()	155
6.10.3.4	getShutterState()	156
6.10.3.5	sendShutterBias()	156
6.10.3.6	setShutterBlanking()	156
6.10.3.7	setShutterBurst()	156
6.10.3.8	setShutterEnd()	157
6.10.3.9	setShutterExternal()	157
6.10.3.10	setShutterInternal()	157
6.10.3.11	setShutterPulseCount()	157
6.10.3.12	setShutterPulsePosition()	157
6.10.3.13	setShutterPulseWidth()	157
6.10.3.14	setShutterSingle()	157
6.10.3.15	setShutterStep()	158
6.10.3.16	setShutterSweepMode()	158
6.11	FliRingBuffer Class Reference	158
6.11.1	Member Function Documentation	159
6.11.1.1	disableGrabN()	159
6.11.1.2	enable()	159
6.11.1.3	enableAccumulationMode()	160
6.11.1.4	enableGrabN()	160
6.11.1.5	enableSubstractMode()	160

6.11.1.6	getFilling()	160
6.11.1.7	getLastImageIndex()	161
6.11.1.8	getNbCountError()	161
6.11.1.9	getNumberOfWrap()	161
6.11.1.10	getSizeInFrames()	161
6.11.1.11	getSizeInMo()	162
6.11.1.12	isEnabled()	162
6.11.1.13	isGrabNEnabled()	162
6.11.1.14	isGrabNFinished()	162
6.11.1.15	nbFramesInAccumulation()	163
6.11.1.16	reset()	163
6.11.1.17	resetAccumulation()	163
6.11.1.18	setFowlerOffset()	163
6.11.1.19	setSizeInFrames()	163
6.11.1.20	setSizeInFramesThermo()	164
6.11.1.21	setSizeInMo()	164
6.12	FliSdk Class Reference	164
6.12.1	Detailed Description	169
6.12.2	Member Enumeration Documentation	169
6.12.2.1	Mode	169
6.12.3	Constructor & Destructor Documentation	169
6.12.3.1	FliSdk() [1/2]	169
6.12.3.2	~FliSdk()	170
6.12.3.3	FliSdk() [2/2]	170
6.12.4	Member Function Documentation	170
6.12.4.1	addEthernetCamera()	170
6.12.4.2	addFakeEthernetCamera()	170
6.12.4.3	addImageProcessing()	171
6.12.4.4	addObserver()	171
6.12.4.5	addRawImageReceivedObserver()	171
6.12.4.6	camera()	172
6.12.4.7	cblueOne()	172
6.12.4.8	cblueSfnc()	172
6.12.4.9	cblueTwo()	172
6.12.4.10	cred()	173
6.12.4.11	credOne()	173
6.12.4.12	credThree()	173
6.12.4.13	credTwo()	173
6.12.4.14	credTwoLite()	174
6.12.4.15	defineGrabOnlySlowMode()	174
6.12.4.16	detectCameras() [1/2]	175
6.12.4.17	detectCameras() [2/2]	175

6.12.4.18 detectEthernetCameras()	175
6.12.4.19 detectGrabbers()	176
6.12.4.20 detectOneCamera()	176
6.12.4.21 disableGrabN()	176
6.12.4.22 display16bImage()	176
6.12.4.23 display8bImage()	177
6.12.4.24 enableFollowUpTheRamp()	177
6.12.4.25 enableFowlerProcessing()	177
6.12.4.26 enableGrabN()	178
6.12.4.27 enableImageTagStateChanged()	179
6.12.4.28 enableIOsForCCsFrameGrabber()	179
6.12.4.29 enableMono8Pixel()	179
6.12.4.30 enableMono8PixelThermo()	179
6.12.4.31 enableObserversNotif()	180
6.12.4.32 enablePowerOverCXP()	180
6.12.4.33 enableRingBuffer()	180
6.12.4.34 enableSubstractMode()	180
6.12.4.35 enableUnsignedPixel()	181
6.12.4.36 exitAllGrabbers()	181
6.12.4.37 forceCurrentCameraModel()	181
6.12.4.38 genicamCamera()	181
6.12.4.39 getAvailableSaveFormats()	182
6.12.4.40 getBufferFilling()	182
6.12.4.41 getBufferNbTimesFull()	182
6.12.4.42 getBufferSize()	182
6.12.4.43 getBurstFilter()	182
6.12.4.44 getCroppingState()	182
6.12.4.45 getCurrentCameraModel()	183
6.12.4.46 getCurrentCameraName()	183
6.12.4.47 getCurrentGrabber()	183
6.12.4.48 getCurrentImageDimension()	183
6.12.4.49 getDetectedCameras()	184
6.12.4.50 getDetectedGrabbers()	184
6.12.4.51 getGrabberIsUSB()	184
6.12.4.52 getImage()	184
6.12.4.53 getImage16b()	185
6.12.4.54 getImagesCapacity()	185
6.12.4.55 getMode()	185
6.12.4.56 getNbCountError()	186
6.12.4.57 getOcamFrameNumber()	186
6.12.4.58 getRawImage()	186
6.12.4.59 getRealFps()	187

6.12.4.60	getSize()	187
6.12.4.61	imageProcessing()	187
6.12.4.62	initLog()	188
6.12.4.63	isCroppingDataValid() [1/2]	188
6.12.4.64	isCroppingDataValid() [2/2]	188
6.12.4.65	isCurrentCameraLink()	189
6.12.4.66	isGrabNEnabled()	189
6.12.4.67	isGrabNFinished()	189
6.12.4.68	isMono8Pixel()	189
6.12.4.69	isStarted()	189
6.12.4.70	isUnsignedPixel()	190
6.12.4.71	listAllGrabbers()	190
6.12.4.72	loadBuffer() [1/3]	190
6.12.4.73	loadBuffer() [2/3]	190
6.12.4.74	loadBuffer() [3/3]	191
6.12.4.75	log()	191
6.12.4.76	logOutside()	192
6.12.4.77	observersNotifEnabled()	192
6.12.4.78	ocam2k()	192
6.12.4.79	ocam2s()	192
6.12.4.80	openMatroxGenicamBrowser()	193
6.12.4.81	operator=()	193
6.12.4.82	removeImageProcessing()	193
6.12.4.83	removeObserver()	193
6.12.4.84	removeRawImageReceivedObserver()	193
6.12.4.85	resetBuffer()	194
6.12.4.86	ringBuffer()	194
6.12.4.87	saveBuffer() [1/2]	194
6.12.4.88	saveBuffer() [2/2]	194
6.12.4.89	serialCamera()	195
6.12.4.90	setBufferSize()	195
6.12.4.91	setBufferSizeInImages()	195
6.12.4.92	setBurstFilter()	196
6.12.4.93	setCamera()	196
6.12.4.94	setCroppingState()	196
6.12.4.95	setFowlerOffset()	197
6.12.4.96	setGrabber()	197
6.12.4.97	setImageDimension()	198
6.12.4.98	setImageDimensionImageRingBuffer()	198
6.12.4.99	setImageDimensionImageRingBufferThermo()	198
6.12.4.100	setMode()	199
6.12.4.101	setNbImagesPerBuffer()	199

6.12.4.102 setOcamFrameNumberOffset()	199
6.12.4.103 setupFixedCCsFrameGrabber()	199
6.12.4.104 sfncCamera()	200
6.12.4.105 start()	200
6.12.4.106 stop()	200
6.12.4.107 update()	200
6.12.4.108 version()	201
6.13 FliSerialCamera Class Reference	201
6.13.1 Detailed Description	202
6.13.2 Constructor & Destructor Documentation	202
6.13.2.1 FliSerialCamera()	202
6.13.2.2 ~FliSerialCamera()	202
6.13.3 Member Function Documentation	203
6.13.3.1 addObserver()	203
6.13.3.2 enableBias()	203
6.13.3.3 enableFlat()	203
6.13.3.4 getCurrentImageDimension()	203
6.13.3.5 getFps()	203
6.13.3.6 getFpsMax()	203
6.13.3.7 getModel()	204
6.13.3.8 getRawData()	204
6.13.3.9 grabberReadSerial()	204
6.13.3.10 grabberWriteSerial()	204
6.13.3.11 isCameraConnected()	204
6.13.3.12 isNumber()	204
6.13.3.13 notifyObservers()	204
6.13.3.14 purgeSerial()	205
6.13.3.15 readSerial()	205
6.13.3.16 removeObserver()	205
6.13.3.17 resynchronizeSerial()	205
6.13.3.18 sendCommand() [1/2]	205
6.13.3.19 sendCommand() [2/2]	206
6.13.3.20 setCustomSerial()	206
6.13.3.21 setFps()	206
6.13.3.22 sleep()	206
6.13.3.23 writeSerial()	207
6.13.4 Friends And Related Function Documentation	207
6.13.4.1 FliCred	207
6.13.4.2 FliCredOne	207
6.13.4.3 FliCredThree	207
6.13.4.4 FliCredTwo	207
6.13.4.5 FliCredTwoLite	207

6.13.4.6 FliOcam2K	207
6.13.4.7 FliOcam2S	208
6.13.4.8 FliSdkImpl	208
6.13.4.9 FliSdkImplCL	208
6.13.5 Member Data Documentation	208
6.13.5.1 _cameraModel	208
6.13.5.2 _croppingFromFunction	208
6.13.5.3 _customSerial	208
6.13.5.4 _grabber	208
6.13.5.5 _needEcho	209
6.13.5.6 _observers	209
6.14 FliSfncCamera Class Reference	209
6.14.1 Detailed Description	238
6.14.2 Constructor & Destructor Documentation	238
6.14.2.1 FliSfncCamera()	238
6.14.2.2 ~FliSfncCamera()	238
6.14.3 Member Data Documentation	238
6.14.3.1 AcquisitionAbort	238
6.14.3.2 AcquisitionArm	239
6.14.3.3 AcquisitionBurstFrameCount	239
6.14.3.4 AcquisitionFrameCount	239
6.14.3.5 AcquisitionFrameRate	239
6.14.3.6 AcquisitionFrameRateEnable	239
6.14.3.7 AcquisitionLineRate	239
6.14.3.8 AcquisitionLineRateEnable	240
6.14.3.9 AcquisitionMode	240
6.14.3.10 AcquisitionStart	240
6.14.3.11 AcquisitionStatus	240
6.14.3.12 AcquisitionStatusSelector	240
6.14.3.13 AcquisitionStop	240
6.14.3.14 AcquisitionStopMode	241
6.14.3.15 ActionDeviceKey	241
6.14.3.16 ActionGroupKey	241
6.14.3.17 ActionGroupMask	241
6.14.3.18 ActionQueueSize	241
6.14.3.19 ActionSelector	241
6.14.3.20 ActionUnconditionalMode	242
6.14.3.21 aPAUSEMACCtrlFramesReceived	242
6.14.3.22 aPAUSEMACCtrlFramesTransmitted	242
6.14.3.23 BalanceRatio	242
6.14.3.24 BalanceRatioSelector	242
6.14.3.25 BalanceWhiteAuto	242

6.14.3.26 BinningHorizontal	243
6.14.3.27 BinningHorizontalMode	243
6.14.3.28 BinningSelector	243
6.14.3.29 BinningVertical	243
6.14.3.30 BinningVerticalMode	243
6.14.3.31 BlackLevel	243
6.14.3.32 BlackLevelAuto	244
6.14.3.33 BlackLevelAutoBalance	244
6.14.3.34 BlackLevelSelector	244
6.14.3.35 CameraPresence	244
6.14.3.36 ChunkBinningHorizontal	244
6.14.3.37 ChunkBinningVertical	244
6.14.3.38 ChunkBlackLevel	245
6.14.3.39 ChunkBlackLevelSelector	245
6.14.3.40 ChunkComponentID	245
6.14.3.41 ChunkComponentIDValue	245
6.14.3.42 ChunkComponentSelector	245
6.14.3.43 ChunkCounterSelector	245
6.14.3.44 ChunkCounterValue	246
6.14.3.45 ChunkDecimationHorizontal	246
6.14.3.46 ChunkDecimationVertical	246
6.14.3.47 ChunkEnable	246
6.14.3.48 ChunkEncoderSelector	246
6.14.3.49 ChunkEncoderStatus	246
6.14.3.50 ChunkEncoderValue	247
6.14.3.51 ChunkExposureTime	247
6.14.3.52 ChunkExposureTimeSelector	247
6.14.3.53 ChunkFrameID	247
6.14.3.54 ChunkGain	247
6.14.3.55 ChunkGainSelector	247
6.14.3.56 ChunkGroupID	248
6.14.3.57 ChunkGroupIDValue	248
6.14.3.58 ChunkGroupSelector	248
6.14.3.59 ChunkHeight	248
6.14.3.60 ChunkLinePitch	248
6.14.3.61 ChunkLineStatusAll	248
6.14.3.62 ChunkModeActive	249
6.14.3.63 ChunkOffsetX	249
6.14.3.64 ChunkOffsetY	249
6.14.3.65 ChunkPixelDynamicRangeMax	249
6.14.3.66 ChunkPixelDynamicRangeMin	249
6.14.3.67 ChunkPixelFormat	249

6.14.3.68	ChunkRegionID	250
6.14.3.69	ChunkRegionIDValue	250
6.14.3.70	ChunkRegionSelector	250
6.14.3.71	ChunkReverseX	250
6.14.3.72	ChunkReverseY	250
6.14.3.73	ChunkScan3dAxisMax	250
6.14.3.74	ChunkScan3dAxisMin	251
6.14.3.75	ChunkScan3dBaseline	251
6.14.3.76	ChunkScan3dCoordinateOffset	251
6.14.3.77	ChunkScan3dCoordinateReferenceSelector	251
6.14.3.78	ChunkScan3dCoordinateReferenceValue	251
6.14.3.79	ChunkScan3dCoordinateScale	251
6.14.3.80	ChunkScan3dCoordinateSelector	252
6.14.3.81	ChunkScan3dCoordinateSystem	252
6.14.3.82	ChunkScan3dCoordinateSystemReference	252
6.14.3.83	ChunkScan3dCoordinateTransformSelector	252
6.14.3.84	ChunkScan3dDistanceUnit	252
6.14.3.85	ChunkScan3dFocallLength	252
6.14.3.86	ChunkScan3dInvalidDataFlag	253
6.14.3.87	ChunkScan3dInvalidDataValue	253
6.14.3.88	ChunkScan3dOutputMode	253
6.14.3.89	ChunkScan3dPrincipalPointU	253
6.14.3.90	ChunkScan3dPrincipalPointV	253
6.14.3.91	ChunkScan3dTransformValue	253
6.14.3.92	ChunkScanLineSelector	254
6.14.3.93	ChunkSelector	254
6.14.3.94	ChunkSequencerSetActive	254
6.14.3.95	ChunkSourceID	254
6.14.3.96	ChunkSourceIDValue	254
6.14.3.97	ChunkSourceSelector	254
6.14.3.98	ChunkStreamChannelID	255
6.14.3.99	ChunkTimerSelector	255
6.14.3.100	ChunkTimerValue	255
6.14.3.101	ChunkTimestamp	255
6.14.3.102	ChunkTimestampLatchValue	255
6.14.3.103	ChunkTransferBlockID	255
6.14.3.104	ChunkTransferQueueCurrentBlockCount	256
6.14.3.105	ChunkTransferStreamID	256
6.14.3.106	ChunkWidth	256
6.14.3.107	ChunkXMLEnable	256
6.14.3.108	CIConfiguration	256
6.14.3.109	CITimeSlotsCount	256

6.14.3.110 ColorTransformationEnable	257
6.14.3.111 ColorTransformationSelector	257
6.14.3.112 ColorTransformationValue	257
6.14.3.113 ColorTransformationValueSelector	257
6.14.3.114 ComponentEnable	257
6.14.3.115 ComponentIDValue	257
6.14.3.116 ComponentSelector	258
6.14.3.117 CounterDuration	258
6.14.3.118 CounterEventActivation	258
6.14.3.119 CounterEventSource	258
6.14.3.120 CounterReset	258
6.14.3.121 CounterResetActivation	258
6.14.3.122 CounterResetSource	259
6.14.3.123 CounterSelector	259
6.14.3.124 CounterStatus	259
6.14.3.125 CounterTriggerActivation	259
6.14.3.126 CounterTriggerSource	259
6.14.3.127 CounterValue	259
6.14.3.128 CounterValueAtReset	260
6.14.3.129 CxpConnectionSelector	260
6.14.3.130 CxpConnectionTestErrorCount	260
6.14.3.131 CxpConnectionTestMode	260
6.14.3.132 CxpConnectionTestPacketCount	260
6.14.3.133 CxpErrorCounterReset	260
6.14.3.134 CxpErrorCounterSelector	261
6.14.3.135 CxpErrorCounterStatus	261
6.14.3.136 CxpErrorCounterValue	261
6.14.3.137 CxpFirstLineTriggerWithFrameStart	261
6.14.3.138 CxpLinkConfiguration	261
6.14.3.139 CxpLinkConfigurationPreferred	261
6.14.3.140 CxpLinkConfigurationStatus	262
6.14.3.141 CxpLinkSharingDuplicateStripe	262
6.14.3.142 CxpLinkSharingEnable	262
6.14.3.143 CxpLinkSharingHorizontalOverlap	262
6.14.3.144 CxpLinkSharingHorizontalStripeCount	262
6.14.3.145 CxpLinkSharingStatus	262
6.14.3.146 CxpLinkSharingSubDeviceSelector	263
6.14.3.147 CxpLinkSharingSubDeviceType	263
6.14.3.148 CxpLinkSharingVerticalOverlap	263
6.14.3.149 CxpLinkSharingVerticalStripeCount	263
6.14.3.150 CxpPoCxpAuto	263
6.14.3.151 CxpPoCxpStatus	263

6.14.3.152 CxpPoCxpTripReset	264
6.14.3.153 CxpPoCxpTurnOff	264
6.14.3.154 CxpSendReceiveSelector	264
6.14.3.155 DecimationHorizontal	264
6.14.3.156 DecimationHorizontalMode	264
6.14.3.157 DecimationVertical	264
6.14.3.158 DecimationVerticalMode	265
6.14.3.159 Deinterlacing	265
6.14.3.160 DeviceCharacterSet	265
6.14.3.161 DeviceClockFrequency	265
6.14.3.162 DeviceClockSelector	265
6.14.3.163 DeviceConnectionSelector	265
6.14.3.164 DeviceConnectionSpeed	266
6.14.3.165 DeviceConnectionStatus	266
6.14.3.166 DeviceEventChannelCount	266
6.14.3.167 DeviceFamilyName	266
6.14.3.168 DeviceFeaturePersistenceEnd	266
6.14.3.169 DeviceFeaturePersistenceStart	266
6.14.3.170 DeviceFirmwareVersion	267
6.14.3.171 DeviceGenCPVersionMajor	267
6.14.3.172 DeviceGenCPVersionMinor	267
6.14.3.173 DeviceIndicatorMode	267
6.14.3.174 DeviceLinkCommandTimeout	267
6.14.3.175 DeviceLinkConnectionCount	267
6.14.3.176 DeviceLinkHeartbeatMode	268
6.14.3.177 DeviceLinkHeartbeatTimeout	268
6.14.3.178 DeviceLinkSelector	268
6.14.3.179 DeviceLinkSpeed	268
6.14.3.180 DeviceLinkThroughputLimit	268
6.14.3.181 DeviceLinkThroughputLimitMode	268
6.14.3.182 DeviceManifestEntrySelector	269
6.14.3.183 DeviceManifestPrimaryURL	269
6.14.3.184 DeviceManifestSchemaMajorVersion	269
6.14.3.185 DeviceManifestSchemaMinorVersion	269
6.14.3.186 DeviceManifestSecondaryURL	269
6.14.3.187 DeviceManifestXMLMajorVersion	269
6.14.3.188 DeviceManifestXMLMinorVersion	270
6.14.3.189 DeviceManifestXMLSubMinorVersion	270
6.14.3.190 DeviceManufacturerInfo	270
6.14.3.191 DeviceMaxThroughput	270
6.14.3.192 DeviceModelName	270
6.14.3.193 DeviceRegistersCheck	270

6.14.3.194 DeviceRegistersEndianness	271
6.14.3.195 DeviceRegistersStreamingEnd	271
6.14.3.196 DeviceRegistersStreamingStart	271
6.14.3.197 DeviceRegistersValid	271
6.14.3.198 DeviceReset	271
6.14.3.199 DeviceScanType	271
6.14.3.200 DeviceSerialNumber	272
6.14.3.201 DeviceSerialPortBaudRate	272
6.14.3.202 DeviceSerialPortSelector	272
6.14.3.203 DeviceSFNCVersionMajor	272
6.14.3.204 DeviceSFNCVersionMinor	272
6.14.3.205 DeviceSFNCVersionSubMinor	272
6.14.3.206 DeviceStreamChannelCount	273
6.14.3.207 DeviceStreamChannelEndianness	273
6.14.3.208 DeviceStreamChannelLink	273
6.14.3.209 DeviceStreamChannelPacketSize	273
6.14.3.210 DeviceStreamChannelSelector	273
6.14.3.211 DeviceStreamChannelType	273
6.14.3.212 DeviceTapGeometry	274
6.14.3.213 DeviceTemperature	274
6.14.3.214 DeviceTemperatureSelector	274
6.14.3.215 DeviceTLType	274
6.14.3.216 DeviceTLVersionMajor	274
6.14.3.217 DeviceTLVersionMinor	274
6.14.3.218 DeviceTLVersionSubMinor	275
6.14.3.219 DeviceType	275
6.14.3.220 DeviceUserID	275
6.14.3.221 DeviceVendorName	275
6.14.3.222 DeviceVersion	275
6.14.3.223 EncoderDivider	275
6.14.3.224 EncoderMode	276
6.14.3.225 EncoderOutputMode	276
6.14.3.226 EncoderReset	276
6.14.3.227 EncoderResetActivation	276
6.14.3.228 EncoderResetSource	276
6.14.3.229 EncoderResolution	276
6.14.3.230 EncoderSelector	277
6.14.3.231 EncoderSourceA	277
6.14.3.232 EncoderSourceB	277
6.14.3.233 EncoderStatus	277
6.14.3.234 EncoderTimeout	277
6.14.3.235 EncoderValue	277

6.14.3.236 EncoderValueAtReset	278
6.14.3.237 EventAcquisitionEnd	278
6.14.3.238 EventAcquisitionEndFrameID	278
6.14.3.239 EventAcquisitionEndTimestamp	278
6.14.3.240 EventAcquisitionError	278
6.14.3.241 EventAcquisitionErrorFrameID	278
6.14.3.242 EventAcquisitionErrorTimestamp	279
6.14.3.243 EventAcquisitionStart	279
6.14.3.244 EventAcquisitionStartFrameID	279
6.14.3.245 EventAcquisitionStartTimestamp	279
6.14.3.246 EventAcquisitionTransferEnd	279
6.14.3.247 EventAcquisitionTransferEndFrameID	279
6.14.3.248 EventAcquisitionTransferEndTimestamp	280
6.14.3.249 EventAcquisitionTransferStart	280
6.14.3.250 EventAcquisitionTransferStartFrameID	280
6.14.3.251 EventAcquisitionTransferStartTimestamp	280
6.14.3.252 EventAcquisitionTrigger	280
6.14.3.253 EventAcquisitionTriggerFrameID	280
6.14.3.254 EventAcquisitionTriggerMissed	281
6.14.3.255 EventAcquisitionTriggerMissedFrameID	281
6.14.3.256 EventAcquisitionTriggerMissedTimestamp	281
6.14.3.257 EventAcquisitionTriggerTimestamp	281
6.14.3.258 EventActionLate	281
6.14.3.259 EventActionLateFrameID	281
6.14.3.260 EventActionLateTimestamp	282
6.14.3.261 EventCounter0End	282
6.14.3.262 EventCounter0EndFrameID	282
6.14.3.263 EventCounter0EndTimestamp	282
6.14.3.264 EventCounter0Start	282
6.14.3.265 EventCounter0StartFrameID	282
6.14.3.266 EventCounter0StartTimestamp	283
6.14.3.267 EventCounter1End	283
6.14.3.268 EventCounter1EndFrameID	283
6.14.3.269 EventCounter1EndTimestamp	283
6.14.3.270 EventCounter1Start	283
6.14.3.271 EventCounter1StartFrameID	283
6.14.3.272 EventCounter1StartTimestamp	284
6.14.3.273 EventEncoder0Restarted	284
6.14.3.274 EventEncoder0RestartedFrameID	284
6.14.3.275 EventEncoder0RestartedTimestamp	284
6.14.3.276 EventEncoder0Stopped	284
6.14.3.277 EventEncoder0StoppedFrameID	284

6.14.3.278 EventEncoder0StoppedTimestamp	285
6.14.3.279 EventEncoder1 Restarted	285
6.14.3.280 EventEncoder1 RestartedFrameID	285
6.14.3.281 EventEncoder1 RestartedTimestamp	285
6.14.3.282 EventEncoder1 Stopped	285
6.14.3.283 EventEncoder1 StoppedFrameID	285
6.14.3.284 EventEncoder1 StoppedTimestamp	286
6.14.3.285 EventError	286
6.14.3.286 EventErrorCode	286
6.14.3.287 EventErrorFrameID	286
6.14.3.288 EventErrorTimestamp	286
6.14.3.289 EventExposureEnd	286
6.14.3.290 EventExposureEndFrameID	287
6.14.3.291 EventExposureEndTimestamp	287
6.14.3.292 EventExposureStart	287
6.14.3.293 EventExposureStartFrameID	287
6.14.3.294 EventExposureStartTimestamp	287
6.14.3.295 EventFrameBurstEnd	287
6.14.3.296 EventFrameBurstEndFrameID	288
6.14.3.297 EventFrameBurstEndTimestamp	288
6.14.3.298 EventFrameBurstStart	288
6.14.3.299 EventFrameBurstStartFrameID	288
6.14.3.300 EventFrameBurstStartTimestamp	288
6.14.3.301 EventFrameEnd	288
6.14.3.302 EventFrameEndFrameID	289
6.14.3.303 EventFrameEndTimestamp	289
6.14.3.304 EventFrameStart	289
6.14.3.305 EventFrameStartFrameID	289
6.14.3.306 EventFrameStartTimestamp	289
6.14.3.307 EventFrameTransferEnd	289
6.14.3.308 EventFrameTransferEndFrameID	290
6.14.3.309 EventFrameTransferEndTimestamp	290
6.14.3.310 EventFrameTransferStart	290
6.14.3.311 EventFrameTransferStartFrameID	290
6.14.3.312 EventFrameTransferStartTimestamp	290
6.14.3.313 EventFrameTrigger	290
6.14.3.314 EventFrameTriggerFrameID	291
6.14.3.315 EventFrameTriggerMissed	291
6.14.3.316 EventFrameTriggerMissedFrameID	291
6.14.3.317 EventFrameTriggerMissedTimestamp	291
6.14.3.318 EventFrameTriggerTimestamp	291
6.14.3.319 EventLine0AnyEdge	291

6.14.3.320 EventLine0AnyEdgeFrameID	292
6.14.3.321 EventLine0AnyEdgeTimestamp	292
6.14.3.322 EventLine0FallingEdge	292
6.14.3.323 EventLine0FallingEdgeFrameID	292
6.14.3.324 EventLine0FallingEdgeTimestamp	292
6.14.3.325 EventLine0RisingEdge	292
6.14.3.326 EventLine0RisingEdgeFrameID	293
6.14.3.327 EventLine0RisingEdgeTimestamp	293
6.14.3.328 EventLine1AnyEdge	293
6.14.3.329 EventLine1AnyEdgeFrameID	293
6.14.3.330 EventLine1AnyEdgeTimestamp	293
6.14.3.331 EventLine1FallingEdge	293
6.14.3.332 EventLine1FallingEdgeFrameID	294
6.14.3.333 EventLine1FallingEdgeTimestamp	294
6.14.3.334 EventLine1RisingEdge	294
6.14.3.335 EventLine1RisingEdgeFrameID	294
6.14.3.336 EventLine1RisingEdgeTimestamp	294
6.14.3.337 EventLineEnd	294
6.14.3.338 EventLineEndFrameID	295
6.14.3.339 EventLineEndTimestamp	295
6.14.3.340 EventLineStart	295
6.14.3.341 EventLineStartFrameID	295
6.14.3.342 EventLineStartTimestamp	295
6.14.3.343 EventLineTrigger	295
6.14.3.344 EventLineTriggerFrameID	296
6.14.3.345 EventLineTriggerMissed	296
6.14.3.346 EventLineTriggerMissedFrameID	296
6.14.3.347 EventLineTriggerMissedTimestamp	296
6.14.3.348 EventLineTriggerTimestamp	296
6.14.3.349 EventLinkSpeedChange	296
6.14.3.350 EventLinkSpeedChangeFrameID	297
6.14.3.351 EventLinkSpeedChangeTimestamp	297
6.14.3.352 EventLinkTrigger0	297
6.14.3.353 EventLinkTrigger0FrameID	297
6.14.3.354 EventLinkTrigger0Timestamp	297
6.14.3.355 EventLinkTrigger1	297
6.14.3.356 EventLinkTrigger1FrameID	298
6.14.3.357 EventLinkTrigger1Timestamp	298
6.14.3.358 EventNotification	298
6.14.3.359 EventPrimaryApplicationSwitch	298
6.14.3.360 EventPrimaryApplicationSwitchFrameID	298
6.14.3.361 EventPrimaryApplicationSwitchTimestamp	298

6.14.3.362 EventSelector	299
6.14.3.363 EventSequencerSetChange	299
6.14.3.364 EventSequencerSetChangeFrameID	299
6.14.3.365 EventSequencerSetChangeTimestamp	299
6.14.3.366 EventStream0TransferBlockEnd	299
6.14.3.367 EventStream0TransferBlockEndFrameID	299
6.14.3.368 EventStream0TransferBlockEndTimestamp	300
6.14.3.369 EventStream0TransferBlockStart	300
6.14.3.370 EventStream0TransferBlockStartFrameID	300
6.14.3.371 EventStream0TransferBlockStartTimestamp	300
6.14.3.372 EventStream0TransferBlockTrigger	300
6.14.3.373 EventStream0TransferBlockTriggerFrameID	300
6.14.3.374 EventStream0TransferBlockTriggerTimestamp	301
6.14.3.375 EventStream0TransferBurstEnd	301
6.14.3.376 EventStream0TransferBurstEndFrameID	301
6.14.3.377 EventStream0TransferBurstEndTimestamp	301
6.14.3.378 EventStream0TransferBurstStart	301
6.14.3.379 EventStream0TransferBurstStartFrameID	301
6.14.3.380 EventStream0TransferBurstStartTimestamp	302
6.14.3.381 EventStream0TransferEnd	302
6.14.3.382 EventStream0TransferEndFrameID	302
6.14.3.383 EventStream0TransferEndTimestamp	302
6.14.3.384 EventStream0TransferOverflow	302
6.14.3.385 EventStream0TransferOverflowFrameID	302
6.14.3.386 EventStream0TransferOverflowTimestamp	303
6.14.3.387 EventStream0TransferPause	303
6.14.3.388 EventStream0TransferPauseFrameID	303
6.14.3.389 EventStream0TransferPauseTimestamp	303
6.14.3.390 EventStream0TransferResume	303
6.14.3.391 EventStream0TransferResumeFrameID	303
6.14.3.392 EventStream0TransferResumeTimestamp	304
6.14.3.393 EventStream0TransferStart	304
6.14.3.394 EventStream0TransferStartFrameID	304
6.14.3.395 EventStream0TransferStartTimestamp	304
6.14.3.396 EventTest	304
6.14.3.397 EventTestTimestamp	304
6.14.3.398 EventTimer0End	305
6.14.3.399 EventTimer0EndFrameID	305
6.14.3.400 EventTimer0EndTimestamp	305
6.14.3.401 EventTimer0Start	305
6.14.3.402 EventTimer0StartFrameID	305
6.14.3.403 EventTimer0StartTimestamp	305

6.14.3.404 EventTimer1End	306
6.14.3.405 EventTimer1EndFrameID	306
6.14.3.406 EventTimer1EndTimestamp	306
6.14.3.407 EventTimer1Start	306
6.14.3.408 EventTimer1StartFrameID	306
6.14.3.409 EventTimer1StartTimestamp	306
6.14.3.410 ExposureAuto	307
6.14.3.411 ExposureMode	307
6.14.3.412 ExposureTime	307
6.14.3.413 ExposureTimeMode	307
6.14.3.414 ExposureTimeSelector	307
6.14.3.415 FileAccessLength	307
6.14.3.416 FileAccessOffset	308
6.14.3.417 FileOpenMode	308
6.14.3.418 FileOperationExecute	308
6.14.3.419 FileOperationResult	308
6.14.3.420 FileOperationSelector	308
6.14.3.421 FileOperationStatus	308
6.14.3.422 FileSelector	309
6.14.3.423 FileSize	309
6.14.3.424 Gain	309
6.14.3.425 GainAuto	309
6.14.3.426 GainAutoBalance	309
6.14.3.427 GainSelector	309
6.14.3.428 Gamma	310
6.14.3.429 GenDCStreamingMode	310
6.14.3.430 GenDCStreamingStatus	310
6.14.3.431 GevActiveLinkCount	310
6.14.3.432 GevCCP	310
6.14.3.433 GevCurrentDefaultGateway	310
6.14.3.434 GevCurrentIPAddress	311
6.14.3.435 GevCurrentIPConfigurationDHCP	311
6.14.3.436 GevCurrentIPConfigurationLLA	311
6.14.3.437 GevCurrentIPConfigurationPersistentIP	311
6.14.3.438 GevCurrentPhysicalLinkConfiguration	311
6.14.3.439 GevCurrentSubnetMask	311
6.14.3.440 GevDiscoveryAckDelay	312
6.14.3.441 GevFirstURL	312
6.14.3.442 GevGVCPExtendedStatusCodes	312
6.14.3.443 GevGVCPExtendedStatusCodesSelector	312
6.14.3.444 GevGVCPPendingAck	312
6.14.3.445 GevGVSPExtendedIDMode	312

6.14.3.446	GevInterfaceSelector	313
6.14.3.447	GevIPConfigurationStatus	313
6.14.3.448	GevMACAddress	313
6.14.3.449	GevMCDA	313
6.14.3.450	GevMCPHostPort	313
6.14.3.451	GevMCRC	313
6.14.3.452	GevMCSP	314
6.14.3.453	GevMCTT	314
6.14.3.454	GevPAUSEFrameReception	314
6.14.3.455	GevPAUSEFrameTransmission	314
6.14.3.456	GevPersistentDefaultGateway	314
6.14.3.457	GevPersistentIPAddress	314
6.14.3.458	GevPersistentSubnetMask	315
6.14.3.459	GevPhysicalLinkConfiguration	315
6.14.3.460	GevPrimaryApplicationIPAddress	315
6.14.3.461	GevPrimaryApplicationSocket	315
6.14.3.462	GevPrimaryApplicationSwitchoverKey	315
6.14.3.463	GevSCCFGAllInTransmission	315
6.14.3.464	GevSCCFGExtendedChunkData	316
6.14.3.465	GevSCCFGPacketResendDestination	316
6.14.3.466	GevSCCFGUnconditionalStreaming	316
6.14.3.467	GevSCDA	316
6.14.3.468	GevSCPD	316
6.14.3.469	GevSCPHostPort	317
6.14.3.470	GevSCPIInterfaceIndex	317
6.14.3.471	GevSCPSDoNotFragment	317
6.14.3.472	GevSCPSFireTestPacket	317
6.14.3.473	GevSCPSPacketSize	317
6.14.3.474	GevSCSP	317
6.14.3.475	GevSCZoneConfigurationLock	318
6.14.3.476	GevSCZoneCount	318
6.14.3.477	GevSCZoneDirectionAll	318
6.14.3.478	GevSecondURL	318
6.14.3.479	GevStreamChannelSelector	318
6.14.3.480	GevSupportedOption	318
6.14.3.481	GevSupportedOptionSelector	319
6.14.3.482	GroupIDValue	319
6.14.3.483	GroupSelector	319
6.14.3.484	Height	319
6.14.3.485	HeightMax	319
6.14.3.486	ImageCompressionBitrate	319
6.14.3.487	ImageCompressionJPEGFormatOption	320

6.14.3.488 ImageCompressionMode	320
6.14.3.489 ImageCompressionQuality	320
6.14.3.490 ImageCompressionRateOption	320
6.14.3.491 LightBrightness	320
6.14.3.492 LightConnectionStatus	321
6.14.3.493 LightControllerSelector	321
6.14.3.494 LightControllerSource	321
6.14.3.495 LightCurrentMeasured	321
6.14.3.496 LightCurrentRating	321
6.14.3.497 LightVoltageMeasured	321
6.14.3.498 LightVoltageRating	322
6.14.3.499 LineFormat	322
6.14.3.500 LineInverter	322
6.14.3.501 LineMode	322
6.14.3.502 LinePitch	322
6.14.3.503 LinePitchEnable	322
6.14.3.504 LineSelector	323
6.14.3.505 LineSource	323
6.14.3.506 LineStatus	323
6.14.3.507 LineStatusAll	323
6.14.3.508 LogicBlockFunction	323
6.14.3.509 LogicBlockInputInverter	323
6.14.3.510 LogicBlockInputNumber	324
6.14.3.511 LogicBlockInputSelector	324
6.14.3.512 LogicBlockInputSource	324
6.14.3.513 LogicBlockLUTIndex	324
6.14.3.514 LogicBlockLUTSelector	324
6.14.3.515 LogicBlockLUTValue	324
6.14.3.516 LogicBlockLUTValueAll	325
6.14.3.517 LogicBlockSelector	325
6.14.3.518 LUTEnable	325
6.14.3.519 LUTIndex	325
6.14.3.520 LUTSelector	325
6.14.3.521 LUTValue	325
6.14.3.522 MultiSlopeExposureGradient	326
6.14.3.523 MultiSlopeExposureLimit	326
6.14.3.524 MultiSlopeIntensityLimit	326
6.14.3.525 MultiSlopeKneePointCount	326
6.14.3.526 MultiSlopeKneePointSelector	326
6.14.3.527 MultiSlopeMode	326
6.14.3.528 MultiSlopeSaturationThreshold	327
6.14.3.529 OffsetX	327

6.14.3.530 OffsetY	327
6.14.3.531 PayloadSize	327
6.14.3.532 PixelColorFilter	327
6.14.3.533 PixelDynamicRangeMax	327
6.14.3.534 PixelDynamicRangeMin	328
6.14.3.535 PixelFormat	328
6.14.3.536 PixelFormatInfoID	328
6.14.3.537 PixelFormatInfoSelector	328
6.14.3.538 PixelSize	328
6.14.3.539 PtpClockAccuracy	328
6.14.3.540 PtpClockID	329
6.14.3.541 PtpDataSetLatch	329
6.14.3.542 PtpEnable	329
6.14.3.543 PtpGrandmasterClockID	329
6.14.3.544 PtpOffsetFromMaster	329
6.14.3.545 PtpParentClockID	329
6.14.3.546 PtpServoStatus	330
6.14.3.547 PtpStatus	330
6.14.3.548 RegionDestination	330
6.14.3.549 RegionIDValue	330
6.14.3.550 RegionMode	330
6.14.3.551 RegionSelector	330
6.14.3.552 ReverseX	331
6.14.3.553 ReverseY	331
6.14.3.554 Scan3dAxisMax	331
6.14.3.555 Scan3dAxisMin	331
6.14.3.556 Scan3dBaseline	331
6.14.3.557 Scan3dCoordinateOffset	331
6.14.3.558 Scan3dCoordinateReferenceSelector	332
6.14.3.559 Scan3dCoordinateReferenceValue	332
6.14.3.560 Scan3dCoordinateScale	332
6.14.3.561 Scan3dCoordinateSelector	332
6.14.3.562 Scan3dCoordinateSystem	332
6.14.3.563 Scan3dCoordinateSystemReference	332
6.14.3.564 Scan3dCoordinateTransformSelector	333
6.14.3.565 Scan3dDistanceUnit	333
6.14.3.566 Scan3dExtractionMethod	333
6.14.3.567 Scan3dExtractionSelector	333
6.14.3.568 Scan3dExtractionSource	333
6.14.3.569 Scan3dFocalLength	333
6.14.3.570 Scan3dInvalidDataFlag	334
6.14.3.571 Scan3dInvalidDataValue	334

6.14.3.572 Scan3dOutputMode	334
6.14.3.573 Scan3dPrincipalPointU	334
6.14.3.574 Scan3dPrincipalPointV	334
6.14.3.575 Scan3dTransformValue	335
6.14.3.576 SensorDigitizationTaps	335
6.14.3.577 SensorHeight	335
6.14.3.578 SensorName	335
6.14.3.579 SensorPixelHeight	335
6.14.3.580 SensorPixelWidth	335
6.14.3.581 SensorShutterMode	336
6.14.3.582 SensorTaps	336
6.14.3.583 SensorWidth	336
6.14.3.584 SequencerConfigurationMode	336
6.14.3.585 SequencerFeatureEnable	336
6.14.3.586 SequencerFeatureSelector	336
6.14.3.587 SequencerMode	337
6.14.3.588 SequencerPathSelector	337
6.14.3.589 SequencerSetActive	337
6.14.3.590 SequencerSetLoad	337
6.14.3.591 SequencerSetNext	337
6.14.3.592 SequencerSetSave	337
6.14.3.593 SequencerSetSelector	338
6.14.3.594 SequencerSetStart	338
6.14.3.595 SequencerTriggerActivation	338
6.14.3.596 SequencerTriggerSource	338
6.14.3.597 SoftwareSignalPulse	338
6.14.3.598 SoftwareSignalSelector	338
6.14.3.599 SourceCount	339
6.14.3.600 SourceIDValue	339
6.14.3.601 SourceSelector	339
6.14.3.602 TestEventGenerate	339
6.14.3.603 TestPattern	339
6.14.3.604 TestPatternGeneratorSelector	339
6.14.3.605 TestPayloadFormatMode	340
6.14.3.606 TestPendingAck	340
6.14.3.607 TimerDelay	340
6.14.3.608 TimerDuration	340
6.14.3.609 TimerReset	340
6.14.3.610 TimerSelector	340
6.14.3.611 TimerStatus	341
6.14.3.612 TimerTriggerActivation	341
6.14.3.613 TimerTriggerArmDelay	341

6.14.3.614 TimerTriggerSource	341
6.14.3.615 TimerValue	341
6.14.3.616 Timestamp	341
6.14.3.617 TimestampLatch	342
6.14.3.618 TimestampLatchValue	342
6.14.3.619 TimestampReset	342
6.14.3.620 TLParamsLocked	342
6.14.3.621 TLParamsLockedSelector	342
6.14.3.622 TLParamsLockedState	342
6.14.3.623 TransferAbort	343
6.14.3.624 TransferBlockCount	343
6.14.3.625 TransferBurstCount	343
6.14.3.626 TransferComponentSelector	343
6.14.3.627 TransferControlMode	343
6.14.3.628 TransferOperationMode	343
6.14.3.629 TransferPause	344
6.14.3.630 TransferQueueCurrentBlockCount	344
6.14.3.631 TransferQueueMaxBlockCount	344
6.14.3.632 TransferQueueMode	344
6.14.3.633 TransferResume	344
6.14.3.634 TransferSelector	344
6.14.3.635 TransferStart	345
6.14.3.636 TransferStatus	345
6.14.3.637 TransferStatusSelector	345
6.14.3.638 TransferStop	345
6.14.3.639 TransferStreamChannel	345
6.14.3.640 TransferTriggerActivation	345
6.14.3.641 TransferTriggerMode	346
6.14.3.642 TransferTriggerSelector	346
6.14.3.643 TransferTriggerSource	346
6.14.3.644 TriggerActivation	346
6.14.3.645 TriggerDelay	346
6.14.3.646 TriggerDivider	346
6.14.3.647 TriggerMode	347
6.14.3.648 TriggerMultiplier	347
6.14.3.649 TriggerOverlap	347
6.14.3.650 TriggerSelector	347
6.14.3.651 TriggerSoftware	347
6.14.3.652 TriggerSource	347
6.14.3.653 UserOutputSelector	348
6.14.3.654 UserOutputValue	348
6.14.3.655 UserOutputValueAll	348

6.14.3.656 UserOutputValueAllMask	348
6.14.3.657 UserSetDefault	348
6.14.3.658 UserSetDescription	348
6.14.3.659 UserSetFeatureEnable	349
6.14.3.660 UserSetFeatureSelector	349
6.14.3.661 UserSetLoad	349
6.14.3.662 UserSetSave	349
6.14.3.663 UserSetSelector	349
6.14.3.664 WhiteClip	349
6.14.3.665 WhiteClipSelector	350
6.14.3.666 Width	350
6.14.3.667 WidthMax	350
6.15 IFliSdkObserver Class Reference	350
6.15.1 Detailed Description	350
6.15.2 Member Function Documentation	350
6.15.2.1 onCameraChanged()	351
6.15.2.2 onFowlerProcessingStateChanged()	351
6.15.2.3 onGrabNStateChanged()	351
6.15.2.4 onResetBufferTriggered()	351
6.15.2.5 onStartedStateChanged()	351
6.16 IImageProcessing Class Reference	351
6.16.1 Member Enumeration Documentation	354
6.16.1.1 BadPixelsAlgo	354
6.16.1.2 ClippingType	354
6.16.1.3 ThermoUnit	354
6.16.2 Member Function Documentation	355
6.16.2.1 autoClipEnabled()	355
6.16.2.2 badPixelsCartoLoaded()	355
6.16.2.3 clip()	355
6.16.2.4 enable8bitsPixel()	355
6.16.2.5 enableAutoClip()	355
6.16.2.6 enableAutoExposure()	356
6.16.2.7 enableBadPixelsCarto()	356
6.16.2.8 enableClahe()	356
6.16.2.9 enableDenoising()	356
6.16.2.10 enableDisplayInfos()	356
6.16.2.11 enableFilters()	356
6.16.2.12 enableImagesAccumulation()	357
6.16.2.13 enableIndependentMode()	357
6.16.2.14 enableManualClippingCoeff()	357
6.16.2.15 enableSharpen()	357
6.16.2.16 enableSmoothImage()	357

6.16.2.17 enableThermo()	357
6.16.2.18 flipHorizontally()	358
6.16.2.19 flipVertically()	358
6.16.2.20 forceCalcMeanStdDevAndHist16b()	358
6.16.2.21 getClipBlack()	358
6.16.2.22 getClippingType()	358
6.16.2.23 getClippingTypeList()	358
6.16.2.24 getClipWhite()	359
6.16.2.25 getCoeffA()	359
6.16.2.26 getCoeffB()	359
6.16.2.27 getColorMapImage()	359
6.16.2.28 getColorMapList()	359
6.16.2.29 getGamma()	359
6.16.2.30 getHistogram16b()	360
6.16.2.31 getHistogram16bNegative()	360
6.16.2.32 getHistogram16bNegativeNoCompute()	360
6.16.2.33 getHistogram16bNoCompute()	360
6.16.2.34 getHistogram8b()	360
6.16.2.35 getMaxVal()	360
6.16.2.36 getMean16b()	361
6.16.2.37 getMean16bNoCompute()	361
6.16.2.38 getMean8b()	361
6.16.2.39 getMinVal()	361
6.16.2.40 getNumThreads()	361
6.16.2.41 getNumThreadsMax()	361
6.16.2.42 getPercentOfMean()	362
6.16.2.43 getProcessedImage()	362
6.16.2.44 getRawThermolImage() [1/2]	362
6.16.2.45 getRawThermolImage() [2/2]	362
6.16.2.46 getSize()	362
6.16.2.47 getSpatialStdDev16b()	362
6.16.2.48 getSpatialStdDev16bNoCompute()	363
6.16.2.49 getSpatialStdDev8b()	363
6.16.2.50 getStdDevAndMeanSelection()	363
6.16.2.51 getThermoCalibrationData()	363
6.16.2.52 getThermoUnit()	363
6.16.2.53 isFlippedHorizontally()	363
6.16.2.54 isFlippedVertically()	364
6.16.2.55 isIndependent()	364
6.16.2.56 isThermoEnabled()	364
6.16.2.57 setBadPixelsCarto()	364
6.16.2.58 setClaheCliplimit()	364

6.16.2.59 setClahTileGridSize()	364
6.16.2.60 setClipBlack()	365
6.16.2.61 setClipDepth()	365
6.16.2.62 setClipLimit()	365
6.16.2.63 setClippingAlpha()	365
6.16.2.64 setClippingBeta()	365
6.16.2.65 setClippingType() [1/2]	365
6.16.2.66 setClippingType() [2/2]	366
6.16.2.67 setClipWhite()	366
6.16.2.68 setColorMapping()	366
6.16.2.69 setDenoisingH()	366
6.16.2.70 setDenoisingSearchWindowSize()	366
6.16.2.71 setDenoisingTemplateWindowSize()	366
6.16.2.72 setDimension()	367
6.16.2.73 setDragoBias()	367
6.16.2.74 setDragoGamma()	367
6.16.2.75 setDragoMultiplier()	367
6.16.2.76 setDragoSaturation()	367
6.16.2.77 setGamma()	368
6.16.2.78 setIlsThermoThrRaw()	368
6.16.2.79 setMantiukGamma()	368
6.16.2.80 setMantiukMultiplier()	368
6.16.2.81 setMantiukSaturation()	368
6.16.2.82 setMantiukScale()	368
6.16.2.83 setnblImagesAccumulation()	369
6.16.2.84 setNumThreads()	369
6.16.2.85 setPercentOfMean()	369
6.16.2.86 setPixelSign()	369
6.16.2.87 setReinhardColorAdapt()	369
6.16.2.88 setReinhardGamma()	369
6.16.2.89 setReinhardIntensity()	370
6.16.2.90 setReinhardLightAdapt()	370
6.16.2.91 setRotationAngle()	370
6.16.2.92 setRotationAngleText()	370
6.16.2.93 setSharpenAlpha()	370
6.16.2.94 setSharpenBeta()	370
6.16.2.95 setSharpenGamma()	371
6.16.2.96 setSharpenKsize()	371
6.16.2.97 setSharpenSigmaX()	371
6.16.2.98 setSharpenSigmaY()	371
6.16.2.99 setStdDevAndMeanSelection()	371
6.16.2.100 setThermoCalibrationData()	372

6.16.2.101 setThermoUnit()	372
6.16.2.102 setToneMappingDrago()	372
6.16.2.103 setToneMappingMantiuk()	372
6.16.2.104 setToneMappingNormal()	372
6.16.2.105 setToneMappingReinhard()	372
6.16.2.106 updateAutoExposureParam()	373
6.16.3 Member Data Documentation	373
6.16.3.1 processMutex	373
6.17 ImageProcessing Class Reference	373
6.17.1 Detailed Description	376
6.17.2 Constructor & Destructor Documentation	376
6.17.2.1 ImageProcessing() [1/3]	376
6.17.2.2 ImageProcessing() [2/3]	376
6.17.2.3 ImageProcessing() [3/3]	376
6.17.2.4 ~ImageProcessing()	376
6.17.3 Member Function Documentation	376
6.17.3.1 aduToDegrees() [1/2]	377
6.17.3.2 aduToDegrees() [2/2]	377
6.17.3.3 autoClipEnabled()	377
6.17.3.4 badPixelsCartoLoaded()	377
6.17.3.5 clip()	377
6.17.3.6 enable8bitsPixel()	378
6.17.3.7 enableAutoClip()	378
6.17.3.8 enableAutoExposure()	378
6.17.3.9 enableBadPixelsCarto()	378
6.17.3.10 enableClahe()	378
6.17.3.11 enableDenoising()	378
6.17.3.12 enableDisplayInfos()	379
6.17.3.13 enableFilters()	379
6.17.3.14 enableImagesAccumulation()	379
6.17.3.15 enableIndependentMode()	379
6.17.3.16 enableManualClippingCoeff()	379
6.17.3.17 enableSharpen()	379
6.17.3.18 enableSmoothImage()	380
6.17.3.19 enableThermo()	380
6.17.3.20 flipHorizontally()	380
6.17.3.21 flipVertically()	380
6.17.3.22 forceCalcMeanStdDevAndHist16b()	380
6.17.3.23 getClipBlack()	380
6.17.3.24 getClippingType()	381
6.17.3.25 getClippingTypeList()	381
6.17.3.26 getClipWhite()	381

6.17.3.27 getCoeffA()	381
6.17.3.28 getCoeffB()	381
6.17.3.29 getColorMapImage()	381
6.17.3.30 getColorMapList()	382
6.17.3.31 getGamma()	382
6.17.3.32 getHistogram16b()	382
6.17.3.33 getHistogram16bNegative()	382
6.17.3.34 getHistogram16bNegativeNoCompute()	382
6.17.3.35 getHistogram16bNoCompute()	382
6.17.3.36 getHistogram8b()	383
6.17.3.37 getMaxVal()	383
6.17.3.38 getMean16b()	383
6.17.3.39 getMean16bNoCompute()	383
6.17.3.40 getMean8b()	383
6.17.3.41 getMinVal()	383
6.17.3.42 getNumThreads()	384
6.17.3.43 getNumThreadsMax()	384
6.17.3.44 getPercentOfMean()	384
6.17.3.45 getProcessedImage() [1/2]	384
6.17.3.46 getProcessedImage() [2/2]	384
6.17.3.47 getProcessedImage16b()	384
6.17.3.48 getRawThermolImage() [1/2]	385
6.17.3.49 getRawThermolImage() [2/2]	385
6.17.3.50 getRingBuffer()	385
6.17.3.51 getSize()	385
6.17.3.52 getSpatialStdDev16b()	385
6.17.3.53 getSpatialStdDev16bNoCompute()	385
6.17.3.54 getSpatialStdDev8b()	386
6.17.3.55 getStdDevAndMeanSelection()	386
6.17.3.56 getThermoCalibrationData()	386
6.17.3.57 getThermoUnit()	386
6.17.3.58 isFlippedHorizontally()	386
6.17.3.59 isFlippedVertically()	386
6.17.3.60 isIndependent()	387
6.17.3.61 isThermoEnabled()	387
6.17.3.62 setBadPixelsCarto()	387
6.17.3.63 setCamera() [1/2]	387
6.17.3.64 setCamera() [2/2]	387
6.17.3.65 setClaheClipLimit()	387
6.17.3.66 setClaheTileGridSize()	388
6.17.3.67 setClipBlack()	388
6.17.3.68 setClipDepth()	388

6.17.3.69 setClipLimit()	388
6.17.3.70 setClippingAlpha()	388
6.17.3.71 setClippingBeta()	389
6.17.3.72 setClippingType() [1/2]	389
6.17.3.73 setClippingType() [2/2]	389
6.17.3.74 setClipWhite()	389
6.17.3.75 setColorMapping()	389
6.17.3.76 setDenoisingH()	389
6.17.3.77 setDenoisingSearchWindowSize()	390
6.17.3.78 setDenoisingTemplateWindowSize()	390
6.17.3.79 setDimension()	390
6.17.3.80 setDragoBias()	390
6.17.3.81 setDragoGamma()	390
6.17.3.82 setDragoMultiplier()	391
6.17.3.83 setDragoSaturation()	391
6.17.3.84 setGamma()	391
6.17.3.85 setIsThermoThrRaw()	391
6.17.3.86 setMantiukGamma()	391
6.17.3.87 setMantiukMultiplier()	391
6.17.3.88 setMantiukSaturation()	392
6.17.3.89 setMantiukScale()	392
6.17.3.90 setnblImagesAccumulation()	392
6.17.3.91 setNumThreads()	392
6.17.3.92 setPercentOfMean()	392
6.17.3.93 setPixelSign()	392
6.17.3.94 setReinhardColorAdapt()	393
6.17.3.95 setReinhardGamma()	393
6.17.3.96 setReinhardIntensity()	393
6.17.3.97 setReinhardLightAdapt()	393
6.17.3.98 setRingBuffer()	393
6.17.3.99 setRotationAngle()	393
6.17.3.100 setRotationAngleText()	394
6.17.3.101 setSharpenAlpha()	394
6.17.3.102 setSharpenBeta()	394
6.17.3.103 setSharpenGamma()	394
6.17.3.104 setSharpenKsize()	394
6.17.3.105 setSharpenSigmaX()	395
6.17.3.106 setSharpenSigmaY()	395
6.17.3.107 setStdDevAndMeanSelection()	395
6.17.3.108 setThermoCalibrationData()	395
6.17.3.109 setThermoUnit()	395
6.17.3.110 setToneMappingDrago()	396

6.17.3.111 setToneMappingMantiuk()	396
6.17.3.112 setToneMappingNormal()	396
6.17.3.113 setToneMappingReinhard()	396
6.17.3.114 updateAutoExposureParam()	396
6.18 ImageRingBuffer Class Reference	396
6.18.1 Detailed Description	398
6.18.2 Constructor & Destructor Documentation	398
6.18.2.1 ImageRingBuffer()	398
6.18.2.2 ~ImageRingBuffer()	398
6.18.3 Member Function Documentation	399
6.18.3.1 disableGrabN()	399
6.18.3.2 enable()	399
6.18.3.3 enable8BitsPixel()	399
6.18.3.4 enable8BitsPixelThermo()	399
6.18.3.5 enableAccumulationMode()	399
6.18.3.6 enableFollowUpTheRamp()	400
6.18.3.7 enableGrabN()	400
6.18.3.8 enableModelmro()	400
6.18.3.9 enableObserversNotif()	400
6.18.3.10 enableSubstractMode()	400
6.18.3.11 getFilling()	401
6.18.3.12 getFps()	401
6.18.3.13 getImage()	401
6.18.3.14 getImageDimension()	401
6.18.3.15 getLastImageIndex()	401
6.18.3.16 getNbCountError()	402
6.18.3.17 getNumberOfWrap()	402
6.18.3.18 getOcamFrameNumber()	402
6.18.3.19 getSizeInFrames()	402
6.18.3.20 getSizeInMo()	403
6.18.3.21 isEnabled()	403
6.18.3.22 isGrabNEnabled()	403
6.18.3.23 isGrabNFinished()	403
6.18.3.24 nbFramesInAccumulation()	404
6.18.3.25 put() [1/2]	404
6.18.3.26 put() [2/2]	405
6.18.3.27 putFollowUpTheRamp()	405
6.18.3.28 putFowler()	405
6.18.3.29 putlota()	406
6.18.3.30 putNro()	406
6.18.3.31 reset()	406
6.18.3.32 resetAccumulation()	406

6.18.3.33 resetCountError()	407
6.18.3.34 resetGrabN()	407
6.18.3.35 resetNbSecondsFps()	407
6.18.3.36 setCameraModel()	407
6.18.3.37 setDefaultCapacity()	407
6.18.3.38 setFowlerOffset()	407
6.18.3.39 setImageDimension()	407
6.18.3.40 setImageDimensionThermo()	408
6.18.3.41 setImageTagState()	408
6.18.3.42 setNbLoop()	408
6.18.3.43 setNbRead()	408
6.18.3.44 setNbReadImro()	408
6.18.3.45 setNbSampPix()	409
6.18.3.46 setObserverList()	409
6.18.3.47 setOcamFrameNumberOffset()	409
6.18.3.48 setSizeInFrames()	409
6.18.3.49 setSizeInFramesThermo()	409
6.18.3.50 setSizeInMo()	410
6.19 IRawImageReceivedObserver Class Reference	410
6.19.1 Detailed Description	410
6.19.2 Member Function Documentation	411
6.19.2.1 fpsTrigger()	411
6.19.2.2 imageReceived()	411
6.19.2.3 imageReceivedAfterBuffer()	411
6.19.2.4 imageReceivedBeforeBuffer()	412
6.19.2.5 useDeprecatedFunction()	412
6.20 Ocam2Conf Struct Reference	412
6.20.1 Member Data Documentation	413
6.20.1.1 binningOffset	413
6.20.1.2 configFile	413
6.20.1.3 fpsMax	413
6.20.1.4 height	413
6.20.1.5 nbIdenticalPixels	413
6.20.1.6 nbPixels	414
6.20.1.7 rawHeight	414
6.20.1.8 rawNbPixels	414
6.20.1.9 rawWidth	414
6.20.1.10 width	414
6.20.1.11 wmode	414
7 File Documentation	415
7.1 FliCblueOne.h File Reference	415

7.2 FliCblueOneEnum.h File Reference	415
7.3 FliCblueSfncEnum.h File Reference	417
7.4 FliCblueTwo.h File Reference	418
7.5 FliCblueTwoEnum.h File Reference	418
7.6 FliCred.h File Reference	419
7.6.1 Typedef Documentation	419
7.6.1.1 FliCamera	419
7.7 FliCredOne.h File Reference	419
7.8 FliCredThree.h File Reference	420
7.9 FliCredTwo.h File Reference	420
7.10 FliCredTwoLite.h File Reference	420
7.11 FliGenicamCamera.h File Reference	420
7.12 FliOcam2K.h File Reference	420
7.12.1 Enumeration Type Documentation	421
7.12.1.1 Ocam2CoolingState	421
7.12.1.2 Ocam2Mode	421
7.13 FliOcam2S.h File Reference	421
7.14 FliRingBuffer.h File Reference	421
7.15 FliSdk.h File Reference	422
7.16 FliSdk_C_V2.h File Reference	422
7.16.1 Typedef Documentation	425
7.16.1.1 callbackHandler	425
7.16.1.2 newImageAvailableCallBack	425
7.16.1.3 saveBufferProgressionCallback	426
7.16.2 Function Documentation	426
7.16.2.1 FliSdk_addCallbackNewImage_V2()	426
7.16.2.2 FliSdk_addEthernetCamera_V2()	426
7.16.2.3 FliSdk_addImageProcessing_V2()	427
7.16.2.4 FliSdk_detectCameras_V2()	427
7.16.2.5 FliSdk_detectGrabbers_V2()	427
7.16.2.6 FliSdk_disableGrabN_V2()	428
7.16.2.7 FliSdk_display16bImage_V2()	428
7.16.2.8 FliSdk_display8bImage_V2()	428
7.16.2.9 FliSdk_enableFollowUpTheRamp_V2()	429
7.16.2.10 FliSdk_enableFowlerProcessing_V2()	429
7.16.2.11 FliSdk_enableGrabN_V2()	429
7.16.2.12 FliSdk_enableRingBuffer_V2()	431
7.16.2.13 FliSdk_enableSubstractMode_V2()	431
7.16.2.14 FliSdk_enableUnsignedPixel_V2()	431
7.16.2.15 FliSdk_exit_V2()	432
7.16.2.16 FliSdk_forceCurrentCameraModel_V2()	432
7.16.2.17 FliSdk_getAvailableSaveFormats_V2()	432

7.16.2.18 FliSdk_getBufferFilling_V2()	433
7.16.2.19 FliSdk_getBufferSize_V2()	433
7.16.2.20 FliSdk_getBufferWithInfo_V2()	433
7.16.2.21 FliSdk_getBurstFilter_V2()	434
7.16.2.22 FliSdk_getCameraModelAsString_V2()	434
7.16.2.23 FliSdk_getCroppingState_V2()	434
7.16.2.24 FliSdk_getCurrentCameraModel_V2()	435
7.16.2.25 FliSdk_getCurrentCameraName_V2()	435
7.16.2.26 FliSdk_getCurrentImageDimension_V2()	435
7.16.2.27 FliSdk_getDetectedCameras_V2()	437
7.16.2.28 FliSdk_getDetectedGrabbers_V2()	437
7.16.2.29 FliSdk_getFps_V2()	437
7.16.2.30 FliSdk_getImagesCapacity_V2()	438
7.16.2.31 FliSdk_getNbCountError_V2()	438
7.16.2.32 FliSdk_getOcamFrameNumber_V2()	438
7.16.2.33 FliSdk_getProcessedImage16b_lv_V2()	439
7.16.2.34 FliSdk_getProcessedImage16b_V2()	439
7.16.2.35 FliSdk_getProcessedImage_lv_V2()	440
7.16.2.36 FliSdk_getProcessedImage_V2()	440
7.16.2.37 FliSdk_getRawImage_lv_V2()	440
7.16.2.38 FliSdk_getRawImage_V2()	441
7.16.2.39 FliSdk_init_V2()	441
7.16.2.40 FliSdk_initLog_V2()	441
7.16.2.41 FliSdk_isCblueOne_V2()	442
7.16.2.42 FliSdk_isCblueSfnc_V2()	442
7.16.2.43 FliSdk_isCblueTwo_V2()	442
7.16.2.44 FliSdk_isCred_V2()	442
7.16.2.45 FliSdk_isCredOne_V2()	443
7.16.2.46 FliSdk_isCredThree_V2()	443
7.16.2.47 FliSdk_isCredTwo_V2()	443
7.16.2.48 FliSdk_isCredTwoLite_V2()	444
7.16.2.49 FliSdk_isCroppingDataValid_V2()	444
7.16.2.50 FliSdk_isGrabNEnabled_V2()	444
7.16.2.51 FliSdk_isGrabNFinished_V2()	446
7.16.2.52 FliSdk_isMono8Pixel_V2()	446
7.16.2.53 FliSdk_isOcam2k_V2()	446
7.16.2.54 FliSdk_isOcam2s_V2()	447
7.16.2.55 FliSdk_isSerialCamera_V2()	447
7.16.2.56 FliSdk_isStarted_V2()	447
7.16.2.57 FliSdk_isUnsignedPixel_V2()	448
7.16.2.58 FliSdk_loadBufferFromFile_V2()	448
7.16.2.59 FliSdk_loadBufferRaw_V2()	448

7.16.2.60 FliSdk_removeCallbackNewImage_V2()	449
7.16.2.61 FliSdk_removeImageProcessing_V2()	449
7.16.2.62 FliSdk_resetBuffer_V2()	449
7.16.2.63 FliSdk_saveBuffer_V2()	450
7.16.2.64 FliSdk_saveBufferWithOptions_V2()	450
7.16.2.65 FliSdk_setBufferSize_V2()	451
7.16.2.66 FliSdk_setBufferSizeInImages_V2()	451
7.16.2.67 FliSdk_setBurstFilter_V2()	451
7.16.2.68 FliSdk_setCamera_V2()	451
7.16.2.69 FliSdk_setCroppingState_V2()	452
7.16.2.70 FliSdk_setFowlerOffset_V2()	452
7.16.2.71 FliSdk_setFpsTrigger_V2()	453
7.16.2.72 FliSdk_setGrabber_V2()	453
7.16.2.73 FliSdk_setImageDimension_V2()	453
7.16.2.74 FliSdk_setMode_V2()	454
7.16.2.75 FliSdk_setNbImagesPerBuffer_V2()	454
7.16.2.76 FliSdk_setOcamFrameNumberOffset_V2()	454
7.16.2.77 FliSdk_start_V2()	455
7.16.2.78 FliSdk_stop_V2()	455
7.16.2.79 FliSdk_update_V2()	455
7.17 FliSerialCamera.h File Reference	456
7.18 FliSfncCamera.h File Reference	456
7.19 IFliSdkObserver.h File Reference	456
7.20 IImageProcessing.h File Reference	456
7.21 ImageProcessing.h File Reference	456
7.22 ImageRingBuffer.h File Reference	456
Index	457

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

FliCblueOneEnum	9
FliCblueSfncEnum	18
FliCblueTwoEnum	28

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

FliGenicamCamera	135
FliSfncCamera	209
FliCblueOne	33
FliCblueTwo	44
FliRingBuffer	158
ImageRingBuffer	396
FliSdk	164
FliSerialCamera	201
FliCred	46
FliCredOne	62
FliCredThree	73
FliCredTwoLite	131
FliCredTwo	101
FliOcam2K	147
FliOcam2S	154
IFliSdkObserver	350
IImageProcessing	351
ImageProcessing	373
IRawImageReceivedObserver	410
Ocam2Conf	412

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

FliCblueOne	33
FliCblueTwo	44
FliCred	
This class manages the methods common to all the C-RED camera (C-RED One, C-RED 2, C-RED 2 Lite, C-RED 2 ER, C-RED 3)	46
FliCredOne	
This class manages the methods specific to the C-RED One camera	62
FliCredThree	
This class manages the methods specific to the C-RED 3 camera	73
FliCredTwo	
This class manages the methods specific to the C-RED 2 and C-RED 2 ER cameras	101
FliCredTwoLite	
This class manages the methods specific to the C-RED 2 Lite camera	131
FliGenicamCamera	
This is the base class of all genicam camera (C-BLUE)	135
FliOcam2K	
This class manages the methods specific to the OCAM2K camera	147
FliOcam2S	
This class manages the methods specific to the OCAM2S camera	154
FliRingBuffer	158
FliSdk	
This class manages the interface with the camera and the grabber	164
FliSerialCamera	
This class is the base class of all serial camera (CameraLink). It implements common C-RED and Ocam functions	201
FliSfncCamera	
This class defined all the register of an SFNC compliant camera	209
IFliSdkObserver	
This interface defines an observer to observe some SDK states	350
IImageProcessing	351
ImageProcessing	
This class manages all the processing of the images such as : statistics (mean stddev), clipping, flipping, sharpen etc..	373
ImageRingBuffer	
This class derive from pure virtual FliRingBuffer and manages the images inside a ring buffer with some put methods to add one or several images or nro or iota buffers into the ring buffer	396

IRawImageReceivedObserver	
This can be herited to be an observer of the reception of a raw image	410
Ocam2Conf	412

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

FliCblueOne.h	415
FliCblueOneEnum.h	415
FliCblueSfncEnum.h	417
FliCblueTwo.h	418
FliCblueTwoEnum.h	418
FliCred.h	419
FliCredOne.h	419
FliCredThree.h	420
FliCredTwo.h	420
FliCredTwoLite.h	420
FliGenicamCamera.h	420
FliOcam2K.h	420
FliOcam2S.h	421
FliRingBuffer.h	421
FliSdk.h	422
FliSdk_C_V2.h	422
FliSerialCamera.h	456
FliSfncCamera.h	456
IFliSdkObserver.h	456
IImageProcessing.h	456
ImageProcessing.h	456
ImageRingBuffer.h	456

Chapter 5

Namespace Documentation

5.1 FliCblueOneEnum Namespace Reference

Enumerations

- enum `DeviceTemperatureSelectorEnum` : `int64_t` {
 `DeviceTemperatureSelectorEnum::Sensor` = 0, `DeviceTemperatureSelectorEnum::CPU` = 1, `DeviceTemperatureSelectorEnum::GPU` = 2, `DeviceTemperatureSelectorEnum::Frontend` = 3,
 `DeviceTemperatureSelectorEnum::Heatsink` = 4, `DeviceTemperatureSelectorEnum::Case` = 5 }
- enum `DeviceTecSelectorEnum` : `int64_t` { `DeviceTecSelectorEnum::TEC1` = 0 }
- enum `DeviceFanModeEnum` : `int64_t` { `DeviceFanModeEnum::Automatic` = 0, `DeviceFanModeEnum::Manual` = 1 }
- enum `FirmwareUpdateStatusEnum` : `int64_t` { `FirmwareUpdateStatusEnum::Idle` = 0, `FirmwareUpdateStatusEnum::InProgress` = 1, `FirmwareUpdateStatusEnum::Done` = 2, `FirmwareUpdateStatusEnum::Failed` = 3 }
- enum `LogCollectStatusEnum` : `int64_t` { `LogCollectStatusEnum::Idle` = 0, `LogCollectStatusEnum::InProgress` = 1, `LogCollectStatusEnum::Done` = 2, `LogCollectStatusEnum::Failed` = 3 }
- enum `IPModeEnum` : `int64_t` { `IPModeEnum::Automatic` = 0, `IPModeEnum::Manual` = 1 }
- enum `SparseSelectorEnum` : `int64_t` {
 `SparseSelectorEnum::Region0` = 0, `SparseSelectorEnum::Region1` = 1, `SparseSelectorEnum::Region2` = 2,
 `SparseSelectorEnum::Region3` = 3,
 `SparseSelectorEnum::Region4` = 4, `SparseSelectorEnum::Region5` = 5, `SparseSelectorEnum::Region6` = 6,
 `SparseSelectorEnum::Region7` = 7 }
- enum `SparseModeEnum` : `int64_t` { `SparseModeEnum::Off` = 0, `SparseModeEnum::On` = 1 }
- enum `TestPatternGeneratorSelectorEnum` : `int64_t` { `TestPatternGeneratorSelectorEnum::Sensor` = 0, `TestPatternGeneratorSelectorEnum::Simulator` = 1 }
- enum `TestPatternEnum` : `int64_t` {
 `TestPatternEnum::Off` = 0, `TestPatternEnum::Black` = 1, `TestPatternEnum::White` = 2, `TestPatternEnum::GreyHorizontalRamp` = 3,
 `TestPatternEnum::SimulatorGreyHorizontalRamp` = 10, `TestPatternEnum::SimulatorGreyHorizontalRampMoving` = 11 }
- enum `GlowReductionEnum` : `int64_t` { `GlowReductionEnum::Off` = 0, `GlowReductionEnum::On` = 1 }
- enum `ConversionEfficiencyEnum` : `int64_t` { `ConversionEfficiencyEnum::Low` = 0, `ConversionEfficiencyEnum::High` = 1 }
- enum `UserSetSelectorEnum` : `int64_t` {
 `UserSetSelectorEnum::Default8bits` = 30, `UserSetSelectorEnum::Default12bits` = 32, `UserSetSelectorEnum::HighSensitivity8bits` = 40,
 `UserSetSelectorEnum::HighSensitivity12bits` = 42,
 `UserSetSelectorEnum::UserSet0` = 0, `UserSetSelectorEnum::UserSet1` = 1, `UserSetSelectorEnum::UserSet2` = 2, `UserSetSelectorEnum::UserSet3` = 3,
 `UserSetSelectorEnum::UserSet4` = 4, `UserSetSelectorEnum::UserSet5` = 5, `UserSetSelectorEnum::UserSet6` = 6, `UserSetSelectorEnum::UserSet7` = 7,
 `UserSetSelectorEnum::UserSet8` = 8, `UserSetSelectorEnum::UserSet9` = 9 }

- enum `UserSetDefaultEnum` : `int64_t` {
`UserSetDefaultEnum::Default8bits = 30`, `UserSetDefaultEnum::Default12bits = 32`, `UserSetDefaultEnum::HighSensitivity8bits = 40`, `UserSetDefaultEnum::HighSensitivity12bits = 42`,
`UserSetDefaultEnum::UserSet0 = 0`, `UserSetDefaultEnum::UserSet1 = 1`, `UserSetDefaultEnum::UserSet2 = 2`, `UserSetDefaultEnum::UserSet3 = 3`,
`UserSetDefaultEnum::UserSet4 = 4`, `UserSetDefaultEnum::UserSet5 = 5`, `UserSetDefaultEnum::UserSet6 = 6`, `UserSetDefaultEnum::UserSet7 = 7`,
`UserSetDefaultEnum::UserSet8 = 8`, `UserSetDefaultEnum::UserSet9 = 9` }

Variables

- const `std::map< std::string, int64_t >` `DeviceTemperatureSelectorStringToValue`
- const `std::map< std::string, int64_t >` `DeviceTecSelectorStringToValue`
- const `std::map< std::string, int64_t >` `DeviceFanModeStringToValue`
- const `std::map< std::string, int64_t >` `FirmwareUpdateStatusStringToValue`
- const `std::map< std::string, int64_t >` `LogCollectStatusStringToValue`
- const `std::map< std::string, int64_t >` `IPModeStringToValue`
- const `std::map< std::string, int64_t >` `SparseSelectorStringToValue`
- const `std::map< std::string, int64_t >` `SparseModeStringToValue`
- const `std::map< std::string, int64_t >` `TestPatternGeneratorSelectorStringToValue`
- const `std::map< std::string, int64_t >` `TestPatternStringToValue`
- const `std::map< std::string, int64_t >` `GlowReductionStringToValue`
- const `std::map< std::string, int64_t >` `ConversionEfficiencyStringToValue`
- const `std::map< std::string, int64_t >` `UserSetSelectorStringToValue`
- const `std::map< std::string, int64_t >` `UserSetDefaultStringToValue`

5.1.1 Enumeration Type Documentation

5.1.1.1 ConversionEfficiencyEnum

```
enum FliCblueOneEnum::ConversionEfficiencyEnum : int64_t [strong]
```

Enumerator

Low	
High	

5.1.1.2 DeviceFanModeEnum

```
enum FliCblueOneEnum::DeviceFanModeEnum : int64_t [strong]
```

Enumerator

Automatic	
Manual	

5.1.1.3 DeviceTecSelectorEnum

```
enum FliCblueOneEnum::DeviceTecSelectorEnum : int64_t [strong]
```

Enumerator

TEC1	
------	--

5.1.1.4 DeviceTemperatureSelectorEnum

```
enum FliCblueOneEnum::DeviceTemperatureSelectorEnum : int64_t [strong]
```

Enumerator

Sensor	
CPU	
Power	
Frontend	
Heatsink	
Case	

5.1.1.5 FirmwareUpdateStatusEnum

```
enum FliCblueOneEnum::FirmwareUpdateStatusEnum : int64_t [strong]
```

Enumerator

Idle	
InProgress	
Done	
Failed	

5.1.1.6 GlowReductionEnum

```
enum FliCblueOneEnum::GlowReductionEnum : int64_t [strong]
```

Enumerator

Off	
On	

5.1.1.7 IPModeEnum

```
enum FliCblueOneEnum::IPModeEnum : int64_t [strong]
```

Enumerator

Automatic	
Manual	

5.1.1.8 LogCollectStatusEnum

```
enum FliCblueOneEnum::LogCollectStatusEnum : int64_t [strong]
```

Enumerator

Idle	
InProgress	
Done	
Failed	

5.1.1.9 SparseModeEnum

```
enum FliCblueOneEnum::SparseModeEnum : int64_t [strong]
```

Enumerator

Off	
On	

5.1.1.10 SparseSelectorEnum

```
enum FliCblueOneEnum::SparseSelectorEnum : int64_t [strong]
```


Enumerator

Region0	
Region1	
Region2	
Region3	
Region4	
Region5	
Region6	
Region7	

5.1.1.11 TestPatternEnum

```
enum FliCblueOneEnum::TestPatternEnum : int64_t [strong]
```

Enumerator

Off	
Black	
White	
GreyHorizontalRamp	
SimulatorGreyHorizontalRamp	
SimulatorGreyHorizontalRampMoving	

5.1.1.12 TestPatternGeneratorSelectorEnum

```
enum FliCblueOneEnum::TestPatternGeneratorSelectorEnum : int64_t [strong]
```

Enumerator

Sensor	
Simulator	

5.1.1.13 UserSetDefaultEnum

```
enum FliCblueOneEnum::UserSetDefaultEnum : int64_t [strong]
```

Enumerator

Default8bits	
--------------	--

Enumerator

Default12bits	
HighSensitivity8bits	
HighSensitivity12bits	
UserSet0	
UserSet1	
UserSet2	
UserSet3	
UserSet4	
UserSet5	
UserSet6	
UserSet7	
UserSet8	
UserSet9	

5.1.1.14 UserSetSelectorEnum

```
enum FliCblueOneEnum::UserSetSelectorEnum : int64_t [strong]
```

Enumerator

Default8bits	
Default12bits	
HighSensitivity8bits	
HighSensitivity12bits	
UserSet0	
UserSet1	
UserSet2	
UserSet3	
UserSet4	
UserSet5	
UserSet6	
UserSet7	
UserSet8	
UserSet9	

5.1.2 Variable Documentation

5.1.2.1 ConversionEfficiencyStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::ConversionEfficiencyStringToValue
```

Initial value:

```
=
{
    {"Low", 0},
    {"High", 1}
}
```

5.1.2.2 DeviceFanModeStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::DeviceFanModeStringToValue
```

Initial value:

```
=
{
    {"Automatic", 0},
    {"Manual", 1}
}
```

5.1.2.3 DeviceTecSelectorStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::DeviceTecSelectorStringToValue
```

Initial value:

```
=
{
    {"TEC1", 0}
}
```

5.1.2.4 DeviceTemperatureSelectorStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::DeviceTemperatureSelectorStringToValue
```

Initial value:

```
=
{
    {"Sensor", 0},
    {"CPU", 1},
    {"Power", 2},
    {"Frontend", 3},
    {"Heatsink", 4},
    {"Case", 5}
}
```

5.1.2.5 FirmwareUpdateStatusStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::FirmwareUpdateStatusStringToValue
```

Initial value:

```
=
{
    {"Idle", 0},
    {"InProgress", 1},
    {"Done", 2},
    {"Failed", 3}
}
```

5.1.2.6 GlowReductionStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::GlowReductionStringToValue
```

Initial value:

```
=  
{  
    {"Off", 0},  
    {"On", 1}  
}
```

5.1.2.7 IPModeStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::IPModeStringToValue
```

Initial value:

```
=  
{  
    {"Automatic", 0},  
    {"Manual", 1}  
}
```

5.1.2.8 LogCollectStatusStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::LogCollectStatusStringToValue
```

Initial value:

```
=  
{  
    {"Idle", 0},  
    {"InProgress", 1},  
    {"Done", 2},  
    {"Failed", 3}  
}
```

5.1.2.9 SparseModeStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::SparseModeStringToValue
```

Initial value:

```
=  
{  
    {"Off", 0},  
    {"On", 1}  
}
```

5.1.2.10 SparseSelectorStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::SparseSelectorStringToValue
```

Initial value:

```
=
{
    {"Region0", 0},
    {"Region1", 1},
    {"Region2", 2},
    {"Region3", 3},
    {"Region4", 4},
    {"Region5", 5},
    {"Region6", 6},
    {"Region7", 7}
}
```

5.1.2.11 TestPatternGeneratorSelectorStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::TestPatternGeneratorSelectorStringToValue
```

Initial value:

```
=
{
    {"Sensor", 0},
    {"Simulator", 1}
}
```

5.1.2.12 TestPatternStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::TestPatternStringToValue
```

Initial value:

```
=
{
    {"Off", 0},
    {"Black", 1},
    {"White", 2},
    {"GreyHorizontalRamp", 3},
    {"SimulatorGreyHorizontalRamp", 10},
    {"SimulatorGreyHorizontalRampMoving", 11}
}
```

5.1.2.13 UserSetDefaultStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::UserSetDefaultStringToValue
```

Initial value:

```
=
{
    {"Default8bits", 30},
    {"Default12bits", 32},
    {"HighSensitivity8bits", 40},
    {"HighSensitivity12bits", 42},
    {"UserSet0", 0},
    {"UserSet1", 1},
    {"UserSet2", 2},
    {"UserSet3", 3},
    {"UserSet4", 4},
    {"UserSet5", 5},
    {"UserSet6", 6},
    {"UserSet7", 7},
    {"UserSet8", 8},
    {"UserSet9", 9}
}
```

5.1.2.14 UserSetSelectorStringToValue

```
const std::map<std::string, int64_t> FliCblueOneEnum::UserSetSelectorStringToValue
```

Initial value:

```
=
{
    {"Default8bits", 30},
    {"Default12bits", 32},
    {"HighSensitivity8bits", 40},
    {"HighSensitivity12bits", 42},
    {"UserSet0", 0},
    {"UserSet1", 1},
    {"UserSet2", 2},
    {"UserSet3", 3},
    {"UserSet4", 4},
    {"UserSet5", 5},
    {"UserSet6", 6},
    {"UserSet7", 7},
    {"UserSet8", 8},
    {"UserSet9", 9}
}
```

5.2 FliCblueSfncEnum Namespace Reference

Enumerations

- enum [DeviceScanTypeEnum](#) : int64_t { [DeviceScanTypeEnum::Areascan](#) = 0 }
- enum [DeviceIndicatorModeEnum](#) : int64_t { [DeviceIndicatorModeEnum::Inactive](#) = 0, [DeviceIndicatorModeEnum::Active](#) = 1, [DeviceIndicatorModeEnum::ErrorStatus](#) = 2 }
- enum [SensorShutterModeEnum](#) : int64_t { [SensorShutterModeEnum::Global](#) = 0, [SensorShutterModeEnum::Rolling](#) = 1, [SensorShutterModeEnum::GlobalReset](#) = 2 }
- enum [RegionSelectorEnum](#) : int64_t { [RegionSelectorEnum::Region0](#) = 0 }
- enum [RegionModeEnum](#) : int64_t { [RegionModeEnum::Off](#) = 0, [RegionModeEnum::On](#) = 1 }
- enum [RegionDestinationEnum](#) : int64_t { [RegionDestinationEnum::Stream0](#) = 0 }
- enum [PixelFormatEnum](#) : int64_t { [PixelFormatEnum::Mono8](#) = 0, [PixelFormatEnum::Mono10](#) = 1, [PixelFormatEnum::Mono12](#) = 2 }
- enum [AcquisitionModeEnum](#) : int64_t { [AcquisitionModeEnum::Continuous](#) = 0 }
- enum [ExposureModeEnum](#) : int64_t { [ExposureModeEnum::Timed](#) = 0 }
- enum [GainSelectorEnum](#) : int64_t { [GainSelectorEnum::AnalogAll](#) = 0, [GainSelectorEnum::DigitalAll](#) = 1 }
- enum [BlackLevelSelectorEnum](#) : int64_t { [BlackLevelSelectorEnum::All](#) = 0 }
- enum [BlackLevelAutoEnum](#) : int64_t { [BlackLevelAutoEnum::Off](#) = 0, [BlackLevelAutoEnum::Continuous](#) = 1 }
- enum [CxpLinkConfigurationStatusEnum](#) : int64_t { [CxpLinkConfigurationStatusEnum::CXP1_X1](#) = 0, [CxpLinkConfigurationStatusEnum::CXP12_X1](#) = 1, [CxpLinkConfigurationStatusEnum::CXP1_X2](#) = 2, [CxpLinkConfigurationStatusEnum::CXP6_X2](#) = 3, [CxpLinkConfigurationStatusEnum::CXP10_X2](#) = 4, [CxpLinkConfigurationStatusEnum::CXP12_X2](#) = 5 }
- enum [CxpLinkConfigurationPreferredEnum](#) : int64_t { [CxpLinkConfigurationPreferredEnum::CXP12_X1](#) = 0, [CxpLinkConfigurationPreferredEnum::CXP6_X2](#) = 1, [CxpLinkConfigurationPreferredEnum::CXP10_X2](#) = 2, [CxpLinkConfigurationPreferredEnum::CXP12_X2](#) = 3 }
- enum [CxpLinkConfigurationEnum](#) : int64_t { [CxpLinkConfigurationEnum::CXP10_X2](#) = 0 }
- enum [CxpConnectionTestModeEnum](#) : int64_t { [CxpConnectionTestModeEnum::Off](#) = 0, [CxpConnectionTestModeEnum::Mode](#) = 1 }
- enum [CxpSendReceiveSelectorEnum](#) : int64_t { [CxpSendReceiveSelectorEnum::Send](#) = 0, [CxpSendReceiveSelectorEnum::Re](#) = 1 }
- enum [CxpErrorCounterSelectorEnum](#) : int64_t { [CxpErrorCounterSelectorEnum::ConnectionLockLoss](#) = 0, [CxpErrorCounterSelectorEnum::Encoding](#) = 1, [CxpErrorCounterSelectorEnum::StreamDataPacketCrc](#) = 2, [CxpErrorCounterSelectorEnum::ControlPacketCrc](#) = 3, [CxpErrorCounterSelectorEnum::EventPacketCrc](#) = 4, [CxpErrorCounterSelectorEnum::DuplicatedCharactersCorrected](#) = 5, [CxpErrorCounterSelectorEnum::DuplicatedCharactersUncorrected](#) = 6 }
- enum [CxpErrorCounterStatusEnum](#) : int64_t { [CxpErrorCounterStatusEnum::CounterActive](#) = 0, [CxpErrorCounterStatusEnum::CounterOverflow](#) = 1 }

Variables

- const std::vector< std::string > [DeviceScanTypeString](#)
- const std::vector< std::string > [DeviceIndicatorModeString](#)
- const std::vector< std::string > [SensorShutterModeString](#)
- const std::vector< std::string > [RegionSelectorString](#)
- const std::vector< std::string > [RegionModeString](#)
- const std::vector< std::string > [RegionDestinationString](#)
- const std::vector< std::string > [PixelFormatString](#)
- const std::vector< std::string > [AcquisitionModeString](#)
- const std::vector< std::string > [ExposureModeString](#)
- const std::vector< std::string > [GainSelectorString](#)
- const std::vector< std::string > [BlackLevelSelectorString](#)
- const std::vector< std::string > [BlackLevelAutoString](#)
- const std::vector< std::string > [CxpLinkConfigurationStatusString](#)
- const std::vector< std::string > [CxpLinkConfigurationPreferredString](#)
- const std::vector< std::string > [CxpLinkConfigurationString](#)
- const std::vector< std::string > [CxpConnectionTestModeString](#)
- const std::vector< std::string > [CxpSendReceiveSelectorString](#)
- const std::vector< std::string > [CxpErrorCounterSelectorString](#)
- const std::vector< std::string > [CxpErrorCounterStatusString](#)
- const std::vector< std::string > [featuresListString](#)

5.2.1 Enumeration Type Documentation

5.2.1.1 AcquisitionModeEnum

```
enum FliCblueSfncEnum::AcquisitionModeEnum : int64_t [strong]
```

Enumerator

Continuous	
------------	--

5.2.1.2 BlackLevelAutoEnum

```
enum FliCblueSfncEnum::BlackLevelAutoEnum : int64_t [strong]
```

Enumerator

Off	
Continuous	

5.2.1.3 BlackLevelSelectorEnum

```
enum FliCblueSfncEnum::BlackLevelSelectorEnum : int64_t [strong]
```

Enumerator

All	
-----	--

5.2.1.4 CxpConnectionTestModeEnum

```
enum FliCblueSfncEnum::CxpConnectionTestModeEnum : int64_t [strong]
```

Enumerator

Off	
Mode1	

5.2.1.5 CxpErrorCounterSelectorEnum

```
enum FliCblueSfncEnum::CxpErrorCounterSelectorEnum : int64_t [strong]
```

Enumerator

ConnectionLockLoss	
Encoding	
StreamDataPacketCrc	
ControlPacketCrc	
EventPacketCrc	
DuplicatedCharactersCorrected	
DuplicatedCharactersUncorrected	

5.2.1.6 CxpErrorCounterStatusEnum

```
enum FliCblueSfncEnum::CxpErrorCounterStatusEnum : int64_t [strong]
```

Enumerator

CounterActive	
CounterOverflow	

5.2.1.7 CxpLinkConfigurationEnum

```
enum FliCblueSfncEnum::CxpLinkConfigurationEnum : int64_t [strong]
```

Enumerator

CXP10_X2	
----------	--

5.2.1.8 CxpLinkConfigurationPreferredEnum

```
enum FliCblueSfncEnum::CxpLinkConfigurationPreferredEnum : int64_t [strong]
```

Enumerator

CXP12_X1	
CXP6_X2	
CXP10_X2	
CXP12_X2	

5.2.1.9 CxpLinkConfigurationStatusEnum

```
enum FliCblueSfncEnum::CxpLinkConfigurationStatusEnum : int64_t [strong]
```

Enumerator

CXP1_X1	
CXP12_X1	
CXP1_X2	
CXP6_X2	
CXP10_X2	
CXP12_X2	

5.2.1.10 CxpSendReceiveSelectorEnum

```
enum FliCblueSfncEnum::CxpSendReceiveSelectorEnum : int64_t [strong]
```

Enumerator

Send	
Receive	

5.2.1.11 DeviceIndicatorModeEnum

```
enum FliCblueSfncEnum::DeviceIndicatorModeEnum : int64_t [strong]
```

Enumerator

Inactive	
Active	
ErrorStatus	

5.2.1.12 DeviceScanTypeEnum

```
enum FliCblueSfncEnum::DeviceScanTypeEnum : int64_t [strong]
```

Enumerator

Areascan	
----------	--

5.2.1.13 ExposureModeEnum

```
enum FliCblueSfncEnum::ExposureModeEnum : int64_t [strong]
```

Enumerator

Timed	
-------	--

5.2.1.14 GainSelectorEnum

```
enum FliCblueSfncEnum::GainSelectorEnum : int64_t [strong]
```

Enumerator

AnalogAll	
DigitalAll	

5.2.1.15 PixelFormatEnum

```
enum FliCblueSfncEnum::PixelFormatEnum : int64_t [strong]
```

Enumerator

Mono8	
Mono10	
Mono12	

5.2.1.16 RegionDestinationEnum

```
enum FliCblueSfncEnum::RegionDestinationEnum : int64_t [strong]
```

Enumerator

Stream0	
---------	--

5.2.1.17 RegionModeEnum

```
enum FliCblueSfncEnum::RegionModeEnum : int64_t [strong]
```

Enumerator

Off	
On	

5.2.1.18 RegionSelectorEnum

```
enum FliCblueSfncEnum::RegionSelectorEnum : int64_t [strong]
```

Enumerator

Region0	
---------	--

5.2.1.19 SensorShutterModeEnum

```
enum FliCblueSfncEnum::SensorShutterModeEnum : int64_t [strong]
```

Enumerator

Global	
Rolling	
GlobalReset	

5.2.2 Variable Documentation

5.2.2.1 AcquisitionModeString

```
const std::vector<std::string> FliCblueSfncEnum::AcquisitionModeString
```

Initial value:

```
=
{
    "Continuous",
}
```

5.2.2.2 BlackLevelAutoString

```
const std::vector<std::string> FliCblueSfncEnum::BlackLevelAutoString
```

Initial value:

```
=
{
    "Off",
    "Continuous",
}
```

5.2.2.3 BlackLevelSelectorString

```
const std::vector<std::string> FliCblueSfncEnum::BlackLevelSelectorString
```

Initial value:

```
=
{
    "All",
}
```

5.2.2.4 CxpConnectionTestModeString

```
const std::vector<std::string> FliCblueSfncEnum::CxpConnectionTestModeString
```

Initial value:

```
=  
{  
    "Off",  
    "Model",  
}
```

5.2.2.5 CxpErrorCounterSelectorString

```
const std::vector<std::string> FliCblueSfncEnum::CxpErrorCounterSelectorString
```

Initial value:

```
=  
{  
    "ConnectionLockLoss",  
    "Encoding",  
    "StreamDataPacketCrc",  
    "ControlPacketCrc",  
    "EventPacketCrc",  
    "DuplicatedCharactersCorrected",  
    "DuplicatedCharactersUncorrected",  
}
```

5.2.2.6 CxpErrorCounterStatusString

```
const std::vector<std::string> FliCblueSfncEnum::CxpErrorCounterStatusString
```

Initial value:

```
=  
{  
    "CounterActive",  
    "CounterOverflow",  
}
```

5.2.2.7 CxpLinkConfigurationPreferredString

```
const std::vector<std::string> FliCblueSfncEnum::CxpLinkConfigurationPreferredString
```

Initial value:

```
=  
{  
    "CXP12_X1",  
    "CXP6_X2",  
    "CXP10_X2",  
    "CXP12_X2",  
}
```

5.2.2.8 CxpLinkConfigurationStatusString

```
const std::vector<std::string> FliCblueSfncEnum::CxpLinkConfigurationStatusString
```

Initial value:

```
=  
{  
    "CXP1_X1",  
    "CXP12_X1",  
    "CXP1_X2",  
    "CXP6_X2",  
    "CXP10_X2",  
    "CXP12_X2",  
}
```

5.2.2.9 CxpLinkConfigurationString

```
const std::vector<std::string> FliCblueSfncEnum::CxpLinkConfigurationString
```

Initial value:

```
=  
{  
    "CXP10_X2",  
}
```

5.2.2.10 CxpSendReceiveSelectorString

```
const std::vector<std::string> FliCblueSfncEnum::CxpSendReceiveSelectorString
```

Initial value:

```
=  
{  
    "Send",  
    "Receive",  
}
```

5.2.2.11 DeviceIndicatorModeString

```
const std::vector<std::string> FliCblueSfncEnum::DeviceIndicatorModeString
```

Initial value:

```
=  
{  
    "Inactive",  
    "Active",  
    "ErrorStatus",  
}
```

5.2.2.12 DeviceScanTypeString

```
const std::vector<std::string> FliCblueSfncEnum::DeviceScanTypeString
```

Initial value:

```
=  
{  
    "Areascan",  
}
```

5.2.2.13 ExposureModeString

```
const std::vector<std::string> FliCblueSfncEnum::ExposureModeString
```

Initial value:

```
=  
{  
    "Timed",  
}
```

5.2.2.14 featuresListString

```
const std::vector<std::string> FliCblueSfncEnum::featuresListString
```

5.2.2.15 GainSelectorString

```
const std::vector<std::string> FliCblueSfncEnum::GainSelectorString
```

Initial value:

```
=  
{  
    "AnalogAll",  
    "DigitalAll",  
}
```

5.2.2.16 PixelFormatString

```
const std::vector<std::string> FliCblueSfncEnum::PixelFormatString
```

Initial value:

```
=  
{  
    "Mono8",  
    "Mono10",  
    "Mono12",  
}
```

5.2.2.17 RegionDestinationString

```
const std::vector<std::string> FliCblueSfncEnum::RegionDestinationString
```

Initial value:

```
=
{
    "Stream0",
}
```

5.2.2.18 RegionModeString

```
const std::vector<std::string> FliCblueSfncEnum::RegionModeString
```

Initial value:

```
=
{
    "Off",
    "On",
}
```

5.2.2.19 RegionSelectorString

```
const std::vector<std::string> FliCblueSfncEnum::RegionSelectorString
```

Initial value:

```
=
{
    "Region0",
}
```

5.2.2.20 SensorShutterModeString

```
const std::vector<std::string> FliCblueSfncEnum::SensorShutterModeString
```

Initial value:

```
=
{
    "Global",
    "Rolling",
    "GlobalReset",
}
```

5.3 FliCblueTwoEnum Namespace Reference

Enumerations

- enum [BinningSelectorEnum](#) : int64_t { [BinningSelectorEnum::Sensor](#) = 0 }
Selects which binning engine is controlled by the BinningHorizontal and BinningVertical features.
- enum [BinningHorizontalModeEnum](#) : int64_t { [BinningHorizontalModeEnum::Sum](#) = 0, [BinningHorizontalModeEnum::Average](#) = 1 }
Sets the mode to use to combine horizontal photo-sensitive cells together when BinningHorizontal is used.
- enum [BinningVerticalModeEnum](#) : int64_t { [BinningVerticalModeEnum::Sum](#) = 0, [BinningVerticalModeEnum::Average](#) = 1 }
Sets the mode to use to combine vertical photo-sensitive cells together when BinningVertical is used.
- enum [FirmwareUpdateStatusEnum](#) : int64_t { [FirmwareUpdateStatusEnum::Idle](#) = 0, [FirmwareUpdateStatusEnum::InProgress](#) = 1, [FirmwareUpdateStatusEnum::Done](#) = 2, [FirmwareUpdateStatusEnum::Failed](#) = 3 }

Variables

- `const std::map< std::string, int64_t >` [BinningSelectorStringToValue](#)
- `const std::map< std::string, int64_t >` [BinningHorizontalModeStringToValue](#)
- `const std::map< std::string, int64_t >` [BinningVerticalModeStringToValue](#)
- `const std::map< std::string, int64_t >` [FirmwareUpdateStatusStringToValue](#)

5.3.1 Enumeration Type Documentation

5.3.1.1 BinningHorizontalModeEnum

```
enum FliCblueTwoEnum::BinningHorizontalModeEnum : int64_t [strong]
```

Sets the mode to use to combine horizontal photo-sensitive cells together when BinningHorizontal is used.

Enumerator

Sum	The response from the combined cells will be added, resulting in increased sensitivity.
Average	The response from the combined cells will be averaged, resulting in increased signal/noise ratio.

5.3.1.2 BinningSelectorEnum

```
enum FliCblueTwoEnum::BinningSelectorEnum : int64_t [strong]
```

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

Enumerator

Sensor	Selected features will control the sensor binning.
--------	--

5.3.1.3 BinningVerticalModeEnum

```
enum FliCblueTwoEnum::BinningVerticalModeEnum : int64_t [strong]
```

Sets the mode to use to combine vertical photo-sensitive cells together when BinningVertical is used.

Enumerator

Sum	The response from the combined cells will be added, resulting in increased sensitivity.
Average	The response from the combined cells will be averaged, resulting in increased signal/noise ratio.

5.3.1.4 FirmwareUpdateStatusEnum

```
enum FliCblueTwoEnum::FirmwareUpdateStatusEnum : int64_t [strong]
```

Enumerator

Idle	
InProgress	
Done	
Failed	

5.3.2 Variable Documentation

5.3.2.1 BinningHorizontalModeStringToValue

```
const std::map<std::string, int64_t> FliCblueTwoEnum::BinningHorizontalModeStringToValue
```

Initial value:

```
=
{
    {"Sum", 0},
    {"Average", 1}
}
```

5.3.2.2 BinningSelectorStringToValue

```
const std::map<std::string, int64_t> FliCblueTwoEnum::BinningSelectorStringToValue
```

Initial value:

```
=
{
    {"Sensor", 0}
}
```

5.3.2.3 BinningVerticalModeStringToValue

```
const std::map<std::string, int64_t> FliCblueTwoEnum::BinningVerticalModeStringToValue
```

Initial value:

```
=
{
    {"Sum", 0},
    {"Average", 1}
}
```

5.3.2.4 FirmwareUpdateStatusStringToValue

```
const std::map<std::string, int64_t> FliCblueTwoEnum::FirmwareUpdateStatusStringToValue
```

Initial value:

```
=  
{  
    {"Idle", 0},  
    {"InProgress", 1},  
    {"Done", 2},  
    {"Failed", 3}  
}
```

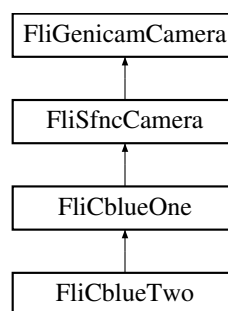

Chapter 6

Class Documentation

6.1 FliCblueOne Class Reference

```
#include <FliCblueOne.h>
```

Inheritance diagram for FliCblueOne:



Public Member Functions

- [FliCblueOne](#) (IFrameGrabberGenicam *grabber)

Public Attributes

- GenicamFeature * [DeviceShutdown](#)
Turns the device off.
- GenicamFeature< [FliCblueOneEnum::DeviceTemperatureSelectorEnum](#) > * [DeviceTemperatureSelector](#)
Selects the location within the device, where the temperature will be measured.
- GenicamFeature< [FliCblueOneEnum::DeviceTecSelectorEnum](#) > * [DeviceTecSelector](#)
Selects the TEC module within the device, where voltage, current and power will be measured.
- GenicamFeature< double > * [DeviceTecVoltage](#)
Voltage applied to TEC in Volts (V). It is measured at the TEC selected by DeviceTecSelector.
- GenicamFeature< double > * [DeviceTecCurrent](#)
Current consumed by the TEC in Amperes (A). It is measured at the TEC selected by DeviceTecSelector.
- GenicamFeature< double > * [DeviceTecPower](#)

- TEC power consumption in Watts (W). It is measured at the TEC selected by DeviceTecSelector.*
- GenicamFeature< [FliCblueOneEnum::DeviceFanModeEnum](#) > * [DeviceFanMode](#)
Selects the mode of operation of the device fan.
 - GenicamFeature< int64_t > * [DeviceFanSpeed](#)
Selects the speed of the fan in manual mode.
 - GenicamFeature< bool > * [DeviceCoolingEnable](#)
Controls if the sensor cooling is enabled.
 - GenicamFeature< double > * [DeviceCoolingSetpoint](#)
Specifies the sensor temperature target when cooling is enabled.
 - GenicamFeature< std::string > * [DeviceStatus](#)
Status of the device.
 - GenicamFeature< std::string > * [DeviceStatusDetailed](#)
Detailed status of the device.
 - GenicamFeature< std::string > * [FirmwareUpdateUri](#)
Specifies location of firmware update (max 255 bytes).
 - GenicamFeature * [FirmwareUpdateExecute](#)
Launches the firmware update procedure. See.
 - GenicamFeature * [FirmwareUpdateAbort](#)
Aborts the firmware update procedure in progress.
 - GenicamFeature * [FirmwareUpdateStatusRefresh](#)
Forces reload of firmware update status. This is only needed for implementation that do not handle IsSelfClearing properly.
 - GenicamFeature< [FliCblueOneEnum::FirmwareUpdateStatusEnum](#) > * [FirmwareUpdateStatus](#)
Returns firmware update status.
 - GenicamFeature< int64_t > * [LogHistoryDepth](#)
Specifies the log history depth in days.
 - GenicamFeature * [LogCollect](#)
Collects the logs.
 - GenicamFeature * [LogCollectAbort](#)
Aborts collecting of the logs.
 - GenicamFeature< [FliCblueOneEnum::LogCollectStatusEnum](#) > * [LogCollectStatus](#)
Returns log collect status.
 - GenicamFeature * [LogCollectStatusRefresh](#)
Forces reload of log collecting status. This is only needed for implementation that do not handle IsSelfClearing properly.
 - GenicamFeature * [LogServe](#)
Serves the logs previously collected.
 - GenicamFeature * [LogServeAbort](#)
Aborts serving of the logs.
 - GenicamFeature< std::string > * [LogServeUri](#)
Specifies location of firmware update.
 - GenicamFeature< std::string > * [CurrentIPAddress](#)
Reports the IP address of the camera Ethernet link.
 - GenicamFeature< std::string > * [CurrentSubnetMask](#)
Reports the subnet mask of the camera Ethernet link.
 - GenicamFeature< [FliCblueOneEnum::IPModeEnum](#) > * [IPMode](#)
Configures how the camera Ethernet link is configured.
 - GenicamFeature * [IPReconfigure](#)
Reconfigures Network.
 - GenicamFeature< std::string > * [StaticIPAddress](#)

- Controls the static IP address of the camera ethernet link. It is only used when the DHCP configuration scheme is disabled.*
- GenicamFeature< std::string > * [StaticSubnetMask](#)
Controls the static subnet mask associated with the static IP address of the camera ethernet link. It is only used when the DHCP configuration scheme is disabled.
 - GenicamFeature< std::string > * [StaticDefaultGateway](#)
Controls the static default gateway associated with the static IP address of the camera ethernet link. It is only used when the DHCP configuration scheme is disabled.
 - GenicamFeature< std::string > * [StaticDomainNameServer](#)
Controls the static domain name server associated with the static IP address of the camera ethernet link. It is only used when the DHCP configuration scheme is disabled.
 - GenicamFeature< std::string > * [StaticAlternateDomainNameServer](#)
Controls the static alternate domain name server associated with the static IP address of the camera ethernet link. It is only used when the DHCP configuration scheme is disabled.
 - GenicamFeature< bool > * [Sparse](#)
This feature controls whether the region is contiguous or split in different areas.
 - GenicamFeature< [FliCblueOneEnum::SparseSelectorEnum](#) > * [SparseSelector](#)
Selects the sparse area to be configured.
 - GenicamFeature< int64_t > * [SparseWidth](#)
Width of the sparse area (in pixels).
 - GenicamFeature< int64_t > * [SparseHeight](#)
Height of the sparse area (in pixels).
 - GenicamFeature< int64_t > * [SparseOffsetX](#)
Horizontal offset from the origin to the sparse area (in pixels).
 - GenicamFeature< int64_t > * [SparseOffsetY](#)
Vertical offset from the origin to the sparse area (in pixels).
 - GenicamFeature< [FliCblueOneEnum::SparseModeEnum](#) > * [SparseMode](#)
Controls if the selected spare area is active.
 - GenicamFeature< [FliCblueOneEnum::TestPatternGeneratorSelectorEnum](#) > * [TestPatternGeneratorSelector](#)
Selects which test pattern generator is controlled by the TestPattern feature.
 - GenicamFeature< [FliCblueOneEnum::TestPatternEnum](#) > * [TestPattern](#)
Selects the type of test pattern that is generated by the device as image source.
 - GenicamFeature< double > * [AcquisitionFrameRateMinReg](#)
Minimum acquisition rate (in Hertz) at which the frames are captured.
 - GenicamFeature< double > * [AcquisitionFrameRateMaxReg](#)
Maximum acquisition rate (in Hertz) at which the frames are captured.
 - GenicamFeature< double > * [ExposureTimeMinReg](#)
 - GenicamFeature< double > * [ExposureTimeMaxReg](#)
 - GenicamFeature< [FliCblueOneEnum::GlowReductionEnum](#) > * [GlowReduction](#)
Controls the glow reduction scheme in use.
 - GenicamFeature< [FliCblueOneEnum::ConversionEfficiencyEnum](#) > * [ConversionEfficiency](#)
Controls the conversion efficiency.
 - GenicamFeature< [FliCblueOneEnum::UserSetSelectorEnum](#) > * [UserSetSelector](#)
Selects the feature User Set to load, save or configure.
 - GenicamFeature< [FliCblueOneEnum::UserSetDefaultEnum](#) > * [UserSetDefault](#)
Selects the feature User Set to load and make active by default when the device is reset.

Additional Inherited Members

6.1.1 Constructor & Destructor Documentation

6.1.1.1 FliCblueOne()

```
FliCblueOne::FliCblueOne (
    IFrameGrabberGenicam * grabber )
```

6.1.2 Member Data Documentation

6.1.2.1 AcquisitionFrameRateMaxReg

```
GenicamFeature<double>* FliCblueOne::AcquisitionFrameRateMaxReg
```

Maximum acquisition rate (in Hertz) at which the frames are captured.

6.1.2.2 AcquisitionFrameRateMinReg

```
GenicamFeature<double>* FliCblueOne::AcquisitionFrameRateMinReg
```

Minimum acquisition rate (in Hertz) at which the frames are captured.

6.1.2.3 ConversionEfficiency

```
GenicamFeature<FliCblueOneEnum::ConversionEfficiencyEnum>* FliCblueOne::ConversionEfficiency
```

Controls the conversion efficiency.

6.1.2.4 CurrentIPAddress

```
GenicamFeature<std::string>* FliCblueOne::CurrentIPAddress
```

Reports the IP address of the camera Ethernet link.

6.1.2.5 CurrentSubnetMask

```
GenicamFeature<std::string>* FliCblueOne::CurrentSubnetMask
```

Reports the subnet mask of the camera Ethernet link.

6.1.2.6 DeviceCoolingEnable

```
GenicamFeature<bool>* FliCblueOne::DeviceCoolingEnable
```

Controls if the sensor cooling is enabled.

6.1.2.7 DeviceCoolingSetpoint

```
GenicamFeature<double>* FliCblueOne::DeviceCoolingSetpoint
```

Specifies the sensor temperature target when cooling is enabled.

6.1.2.8 DeviceFanMode

```
GenicamFeature<FliCblueOneEnum::DeviceFanModeEnum>* FliCblueOne::DeviceFanMode
```

Selects the mode of operation of the device fan.

6.1.2.9 DeviceFanSpeed

```
GenicamFeature<int64_t>* FliCblueOne::DeviceFanSpeed
```

Selects the speed of the fan in manual mode.

6.1.2.10 DeviceShutdown

```
GenicamFeature* FliCblueOne::DeviceShutdown
```

Turns the device off.

6.1.2.11 DeviceStatus

```
GenicamFeature<std::string>* FliCblueOne::DeviceStatus
```

Status of the device.

6.1.2.12 DeviceStatusDetailed

```
GenicamFeature<std::string>* FliCblueOne::DeviceStatusDetailed
```

Detailed status of the device.

6.1.2.13 DeviceTecCurrent

```
GenicamFeature<double>* FliCblueOne::DeviceTecCurrent
```

Current consumed by the TEC in Amperes (A). It is measured at the TEC selected by DeviceTecSelector.

6.1.2.14 DeviceTecPower

```
GenicamFeature<double>* FliCblueOne::DeviceTecPower
```

TEC power consumption in Watts (W). It is measured at the TEC selected by DeviceTecSelector.

6.1.2.15 DeviceTecSelector

```
GenicamFeature<FliCblueOneEnum::DeviceTecSelectorEnum>* FliCblueOne::DeviceTecSelector
```

Selects the TEC module within the device, where voltage, current and power will be measured.

6.1.2.16 DeviceTecVoltage

```
GenicamFeature<double>* FliCblueOne::DeviceTecVoltage
```

Voltage applied to TEC in Volts (V). It is measured at the TEC selected by DeviceTecSelector.

6.1.2.17 DeviceTemperatureSelector

```
GenicamFeature<FliCblueOneEnum::DeviceTemperatureSelectorEnum>* FliCblueOne::DeviceTemperatureSelector
```

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

6.1.2.18 ExposureTimeMaxReg

GenicamFeature<double>* FliCblueOne::ExposureTimeMaxReg

6.1.2.19 ExposureTimeMinReg

GenicamFeature<double>* FliCblueOne::ExposureTimeMinReg

6.1.2.20 FirmwareUpdateAbort

GenicamFeature* FliCblueOne::FirmwareUpdateAbort

Aborts the firmware update procedure in progress.

6.1.2.21 FirmwareUpdateExecute

GenicamFeature* FliCblueOne::FirmwareUpdateExecute

Launches the firmware update procedure. See.

6.1.2.22 FirmwareUpdateStatus

GenicamFeature<FliCblueOneEnum::FirmwareUpdateStatusEnum>* FliCblueOne::FirmwareUpdateStatus

Returns firmware update status.

6.1.2.23 FirmwareUpdateStatusRefresh

GenicamFeature* FliCblueOne::FirmwareUpdateStatusRefresh

Forces reload of firmware update status. This is only needed for implementation that do not handle IsSelfClearing properly.

6.1.2.24 FirmwareUpdateUri

```
GenicamFeature<std::string>* FliCblueOne::FirmwareUpdateUri
```

Specifies location of firmware update (max 255 bytes).

6.1.2.25 GlowReduction

```
GenicamFeature<FliCblueOneEnum::GlowReductionEnum>* FliCblueOne::GlowReduction
```

Controls the glow reduction scheme in use.

6.1.2.26 IPMode

```
GenicamFeature<FliCblueOneEnum::IPModeEnum>* FliCblueOne::IPMode
```

Configures how the camera Ethernet link is configured.

6.1.2.27 IPReconfigure

```
GenicamFeature* FliCblueOne::IPReconfigure
```

Reconfigures Network.

6.1.2.28 LogCollect

```
GenicamFeature* FliCblueOne::LogCollect
```

Collects the logs.

6.1.2.29 LogCollectAbort

```
GenicamFeature* FliCblueOne::LogCollectAbort
```

Aborts collecting of the logs.

6.1.2.30 LogCollectStatus

```
GenicamFeature<FliCblueOneEnum::LogCollectStatusEnum>* FliCblueOne::LogCollectStatus
```

Returns log collect status.

6.1.2.31 LogCollectStatusRefresh

```
GenicamFeature* FliCblueOne::LogCollectStatusRefresh
```

Forces reload of log collecting status. This is only needed for implementation that do not handle IsSelfClearing properly.

6.1.2.32 LogHistoryDepth

```
GenicamFeature<int64_t>* FliCblueOne::LogHistoryDepth
```

Specifies the log history depth in days.

6.1.2.33 LogServe

```
GenicamFeature* FliCblueOne::LogServe
```

Serves the logs previously collected.

6.1.2.34 LogServeAbort

```
GenicamFeature* FliCblueOne::LogServeAbort
```

Aborts serving of the logs.

6.1.2.35 LogServeUri

```
GenicamFeature<std::string>* FliCblueOne::LogServeUri
```

Specifies location of firmware update.

6.1.2.36 Sparse

```
GenicamFeature<bool>* FliCblueOne::Sparse
```

This feature controls whether the region is contiguous or split in different areas.

6.1.2.37 SparseHeight

```
GenicamFeature<int64_t>* FliCblueOne::SparseHeight
```

Height of the sparse area (in pixels).

6.1.2.38 SparseMode

```
GenicamFeature<FliCblueOneEnum::SparseModeEnum>* FliCblueOne::SparseMode
```

Controls if the selected spare area is active.

6.1.2.39 SparseOffsetX

```
GenicamFeature<int64_t>* FliCblueOne::SparseOffsetX
```

Horizontal offset from the origin to the sparse area (in pixels).

6.1.2.40 SparseOffsetY

```
GenicamFeature<int64_t>* FliCblueOne::SparseOffsetY
```

Vertical offset from the origin to the sparse area (in pixels).

6.1.2.41 SparseSelector

```
GenicamFeature<FliCblueOneEnum::SparseSelectorEnum>* FliCblueOne::SparseSelector
```

Selects the sparse area to be configured.

6.1.2.42 SparseWidth

```
GenicamFeature<int64_t>* FliCblueOne::SparseWidth
```

Width of the sparse area (in pixels).

6.1.2.43 StaticAlternateDomainNameServer

```
GenicamFeature<std::string>* FliCblueOne::StaticAlternateDomainNameServer
```

Controls the static alternate domain name server associated with the static IP address of the camera ethernet link. It is only used when the DHCP configuration scheme is disabled.

6.1.2.44 StaticDefaultGateway

```
GenicamFeature<std::string>* FliCblueOne::StaticDefaultGateway
```

Controls the static default gateway associated with the static IP address of the camera ethernet link. It is only used when the DHCP configuration scheme is disabled.

6.1.2.45 StaticDomainNameServer

```
GenicamFeature<std::string>* FliCblueOne::StaticDomainNameServer
```

Controls the static domain name server associated with the static IP address of the camera ethernet link. It is only used when the DHCP configuration scheme is disabled.

6.1.2.46 StaticIPAddress

```
GenicamFeature<std::string>* FliCblueOne::StaticIPAddress
```

Controls the static IP address of the camera ethernet link. It is only used when the DHCP configuration scheme is disabled.

6.1.2.47 StaticSubnetMask

```
GenicamFeature<std::string>* FliCblueOne::StaticSubnetMask
```

Controls the static subnet mask associated with the static IP address of the camera ethernet link. It is only used when the DHCP configuration scheme is disabled.

6.1.2.48 TestPattern

```
GenicamFeature<FliCblueOneEnum::TestPatternEnum>* FliCblueOne::TestPattern
```

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

6.1.2.49 TestPatternGeneratorSelector

```
GenicamFeature<FliCblueOneEnum::TestPatternGeneratorSelectorEnum>* FliCblueOne::TestPatternGeneratorSelector
```

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

6.1.2.50 UserSetDefault

```
GenicamFeature<FliCblueOneEnum::UserSetDefaultEnum>* FliCblueOne::UserSetDefault
```

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

6.1.2.51 UserSetSelector

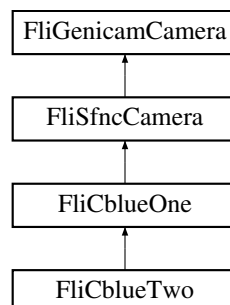
```
GenicamFeature<FliCblueOneEnum::UserSetSelectorEnum>* FliCblueOne::UserSetSelector
```

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

6.2 FliCblueTwo Class Reference

```
#include <FliCblueTwo.h>
```

Inheritance diagram for FliCblueTwo:



Public Member Functions

- [FliCblueTwo](#) (IFrameGrabberGenicam *grabber)

Public Attributes

- GenicamFeature< int64_t > * [BinningHorizontal](#)
Number of horizontal photo-sensitive cells to combine together. This reduces the horizontal resolution (width) of the image.
- GenicamFeature< int64_t > * [BinningVertical](#)
Number of vertical photo - sensitive cells to combine together. This reduces the vertical resolution (height) of the image.
- GenicamFeature< [FliCblueTwoEnum::BinningHorizontalModeEnum](#) > * [BinningHorizontalMode](#)
Sets the mode to use to combine horizontal photo-sensitive cells together when BinningHorizontal is used.
- GenicamFeature< [FliCblueTwoEnum::BinningVerticalModeEnum](#) > * [BinningVerticalMode](#)
Sets the mode to use to combine vertical photo-sensitive cells together when BinningVertical is used.
- GenicamFeature< [FliCblueTwoEnum::FirmwareUpdateStatusEnum](#) > * [FirmwareUpdateStatus](#)
Returns firmware update status.

Additional Inherited Members

6.2.1 Constructor & Destructor Documentation

6.2.1.1 FliCblueTwo()

```
FliCblueTwo::FliCblueTwo (
    IFrameGrabberGenicam * grabber )
```

6.2.2 Member Data Documentation

6.2.2.1 BinningHorizontal

```
GenicamFeature<int64_t>* FliCblueTwo::BinningHorizontal
```

Number of horizontal photo-sensitive cells to combine together. This reduces the horizontal resolution (width) of the image.

6.2.2.2 BinningHorizontalMode

```
GenicamFeature<FliCblueTwoEnum::BinningHorizontalModeEnum>* FliCblueTwo::BinningHorizontalMode
```

Sets the mode to use to combine horizontal photo-sensitive cells together when BinningHorizontal is used.

6.2.2.3 BinningVertical

```
GenicamFeature<int64_t>* FliCblueTwo::BinningVertical
```

Number of vertical photo - sensitive cells to combine together. This reduces the vertical resolution (height) of the image.

6.2.2.4 BinningVerticalMode

```
GenicamFeature<FliCblueTwoEnum::BinningVerticalModeEnum>* FliCblueTwo::BinningVerticalMode
```

Sets the mode to use to combine vertical photo-sensitive cells together when BinningVertical is used.

6.2.2.5 FirmwareUpdateStatus

```
GenicamFeature<FliCblueTwoEnum::FirmwareUpdateStatusEnum>* FliCblueTwo::FirmwareUpdateStatus
```

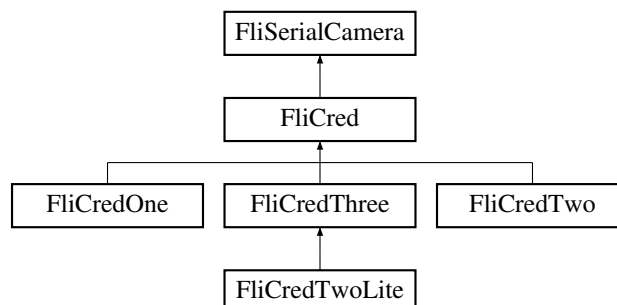
Returns firmware update status.

6.3 FliCred Class Reference

This class manages the methods common to all the C-RED camera (C-RED One, C-RED 2, C-RED 2 Lite, C-RED 2 ER, C-RED 3)

```
#include <FliCred.h>
```

Inheritance diagram for FliCred:



Public Member Functions

- [FliCred](#) (IFrameGrabberCL *grabber)
- FliSdkError [getAduOffset](#) (int &aduOffset)
- FliSdkError [getBiasState](#) (bool &enabled)
- FliSdkError [getFlatState](#) (bool &enabled)
- FliSdkError [getEventsState](#) (bool &enabled)
- FliSdkError [getCameraType](#) (std::string &info)
- FliSdkError [getHwuid](#) (std::string &hwuid)
- FliSdkError [getImageTagsState](#) (bool &enabled)
- FliSdkError [getLedState](#) (bool &enabled)
- FliSdkError [getPassword](#) (std::string &password)
- FliSdkError [getExtSynchroState](#) (bool &enabled)
- FliSdkError [getIpConfig](#) (std::string &macAddress, std::string &ipAddress, std::string &mask, bool &established)
- FliSdkError [getStatusDetailed](#) (std::string &status, std::string &diag)
- FliSdkError [getStatus](#) (std::string &status)
- FliSdkError [getVersions](#) (std::string &firmware, std::string &fpga, std::string &hardware)
- FliSdkError [getVersionFirmware](#) (std::string &version)
- FliSdkError [getVersionFirmwareBuild](#) (std::string &build)
- FliSdkError [getVersionFirmwareDetailed](#) (std::string &detailed)
- FliSdkError [getVersionFpga](#) (std::string &version)
- FliSdkError [getVersionHardware](#) (std::string &version)
- FliSdkError [getIsSlowMode](#) (bool &slowmode)
- FliSdkError [getCheckTag4by4](#) (bool &tag4by4)
- FliSdkError [getExcludeBorder](#) (bool &exclude)
- FliSdkError [getBadPixelModeOnOff](#) (bool &checked)
- FliSdkError [getKindOfBadPixelCorrection](#) (int &correction)
 - getKindOfBadPixelCorrection* : get the kind of correction to apply to bad pixels
- FliSdkError [getFilteringModeOnOff](#) (bool &checked)
- FliSdkError [getUserConvolutionMatrixIndex_V2](#) (int &index)
 - getUserConvolutionMatrixIndex_V2* : get the current convolution matrix index
- FliSdkError [getUserConvolutionMatrix](#) (std::vector< std::vector< double > > &matrixBadPixels, double &divisor, std::string &description)
 - FliCred_getUserConvolutionMatrix_V2* : method to get the convolution matrix from the camera.
- FliSdkError [enableExtSynchro](#) (bool enable)
- FliSdkError [enableImageTags](#) (bool enable)
- FliSdkError [enableEvents](#) (bool enable)
- FliSdkError [enableLed](#) (bool enable)
- FliSdkError [enableTelnet](#) (bool enable)
- FliSdkError [enableCropping](#) (bool enable)
- FliSdkError [setIpAddress](#) (std::string ip)
- FliSdkError [setIpAlternateDns](#) (std::string dns)
- FliSdkError [setIpDns](#) (std::string dns)
- FliSdkError [setIpGateway](#) (std::string gateway)
- FliSdkError [setIpAutomatic](#) ()
- FliSdkError [setIpManual](#) ()
- FliSdkError [setIpRefresh](#) ()
- FliSdkError [setIpNetmask](#) (std::string netmask)
- FliSdkError [setPassword](#) (std::string password)
- FliSdkError [setAduOffset](#) (int aduOffset)
- FliSdkError [setSlowMode](#) (bool slowMode)
- FliSdkError [setExcludeBorderOnOff](#) (bool exclude)
- FliSdkError [setBadPixelModeOnOff](#) (bool checked)

- FliSdkError [setFilteringModeOnOff](#) (bool checked)
- FliSdkError [setKindOfBadPixelCorrection](#) (int correction)
 - setKindOfBadPixelCorrection* : set the kind of correction to apply to bad pixels
- FliSdkError [setUserConvolutionMatrixIndex_V2](#) (const int index)
 - setUserConvolutionMatrixIndex_V2* : set the index of the current convolution matrix
- FliSdkError [setUserConvolutionMatrix](#) (std::vector< std::vector< double > > &matrixBadPixels, double divisor, std::string &description)
 - FliCred_setUserConvolutionMatrix_V2* : method to set the convolution matrix inside the camera.
- FliSdkError [saveCameraSettings](#) ()
 - saveCameraSettings* save all the user (not the factory) settings inside the camera
- FliSdkError [continueStarting](#) ()
 - continueStarting* let the camera continue after a reboot for instance
- FliSdkError [shutDown](#) ()
 - shutDown* will shut down the camera
- FliSdkError [getLogs](#) (std::string &url)
 - getLogs* start the logs and return the url of them
- FliSdkError [getLogs](#) (uint16_t nbDays, std::string &url)
 - getLogs* start the logs and return the number of days last ones the url of them
- FliSdkError [buildFlat](#) ()
 - buildFlat* will execute the build of the flat
- FliSdkError [buildBias](#) ()
 - buildBias* will execute the build of the bias
- FliSdkError [restoreFactory](#) ()
 - restoreFactory* will restore the factory settings inside the camera as the user settings
- FliSdkError [upgradeFirmware](#) (std::string url)
 - upgradeFirmware* will upgrade the Firmware inside the camera from the url of a firmware file
- FliSdkError [sendBiasFromUrl](#) (std::string url)
 - sendBiasFromUrl* send to the camera a bias file to the camera from an url
- FliSdkError [sendFlatFromUrl](#) (std::string url)
 - sendFlatFromUrl* send to the camera a flat file to the camera from an url
- FliSdkError [sendBiasFile](#) (std::string filePath)
 - sendBiasFile* send to the camera a bias file to the camera
- FliSdkError [sendFlatFile](#) (std::string filePath)
 - sendFlatFile* send to the camera a flat file to the camera
- FliSdkError [setThresholdingOnOff](#) (bool checked)
- FliSdkError [getThresholdingOnOff](#) (bool &checked)
- FliSdkError [setThresholdingLevelsValues](#) (int lowLevel, int highLevel, int lowValue, int middleValue, int high↔Value)
 - setThresholdingLevelsValues* set the levels and values of the thresholding

Additional Inherited Members

6.3.1 Detailed Description

This class manages the methods common to all the C-RED camera (C-RED One, C-RED 2, C-RED 2 Lite, C-RED 2 ER, C-RED 3)

6.3.2 Constructor & Destructor Documentation

6.3.2.1 FliCred()

```
FliCred::FliCred (
    IFrameGrabberCL * grabber )
```

6.3.3 Member Function Documentation

6.3.3.1 buildBias()

```
FliSdkError FliCred::buildBias ( )
```

buildBias will execute the build of the bias

Returns

a FliSdkError or FLISDK_NO_ERROR

6.3.3.2 buildFlat()

```
FliSdkError FliCred::buildFlat ( )
```

buildFlat will execute the build of the flat

Returns

a FliSdkError or FLISDK_NO_ERROR

6.3.3.3 continueStarting()

```
FliSdkError FliCred::continueStarting ( )
```

continueStarting let the camera continue after a reboot for instance

Returns

a FliSdkError or FLISDK_NO_ERROR

6.3.3.4 enableCropping()

```
FliSdkError FliCred::enableCropping (
    bool enable )
```

6.3.3.5 enableEvents()

```
FliSdkError FliCred::enableEvents (
    bool enable )
```

6.3.3.6 enableExtSynchro()

```
FliSdkError FliCred::enableExtSynchro (
    bool enable )
```

6.3.3.7 enableImageTags()

```
FliSdkError FliCred::enableImageTags (
    bool enable )
```

6.3.3.8 enableLed()

```
FliSdkError FliCred::enableLed (
    bool enable )
```

6.3.3.9 enableTelnet()

```
FliSdkError FliCred::enableTelnet (
    bool enable )
```

6.3.3.10 getAduOffset()

```
FliSdkError FliCred::getAduOffset (
    int & aduOffset )
```

6.3.3.11 getBadPixelModeOnOff()

```
FliSdkError FliCred::getBadPixelModeOnOff (
    bool & checked )
```

6.3.3.12 getBiasState()

```
FliSdkError FliCred::getBiasState (
    bool & enabled )
```

6.3.3.13 getCameraType()

```
FliSdkError FliCred::getCameraType (
    std::string & info )
```

6.3.3.14 getCheckTag4by4()

```
FliSdkError FliCred::getCheckTag4by4 (
    bool & tag4by4 )
```

6.3.3.15 getEventsState()

```
FliSdkError FliCred::getEventsState (
    bool & enabled )
```

6.3.3.16 getExcludeBorder()

```
FliSdkError FliCred::getExcludeBorder (
    bool & exclude )
```

6.3.3.17 getExtSynchroState()

```
FliSdkError FliCred::getExtSynchroState (
    bool & enabled )
```

6.3.3.18 getFilteringModeOnOff()

```
FliSdkError FliCred::getFilteringModeOnOff (
    bool & checked )
```

6.3.3.19 getFlatState()

```
FliSdkError FliCred::getFlatState (
    bool & enabled )
```

6.3.3.20 getHwuid()

```
FliSdkError FliCred::getHwuid (
    std::string & hwuid )
```

6.3.3.21 getImageTagsState()

```
FliSdkError FliCred::getImageTagsState (
    bool & enabled )
```

6.3.3.22 getIpConfig()

```
FliSdkError FliCred::getIpConfig (
    std::string & macAddress,
    std::string & ipAddress,
    std::string & mask,
    bool & established )
```

6.3.3.23 getIsSlowMode()

```
FliSdkError FliCred::getIsSlowMode (
    bool & slowmode )
```

6.3.3.24 getKindOfBadPixelCorrection()

```
FliSdkError FliCred::getKindOfBadPixelCorrection (
    int & correction )
```

getKindOfBadPixelCorrection : get the kind of correction to apply to bad pixels

Parameters

<i>correction</i>	: 1 = Low latency bad pixels correction; 2 = Convolution bad pixels correction
-------------------	--

Returns

a FliSdkError or FLISDK_NO_ERROR

6.3.3.25 getLedState()

```
FliSdkError FliCred::getLedState (
    bool & enabled )
```

6.3.3.26 getLogs() [1/2]

```
FliSdkError FliCred::getLogs (
    std::string & url )
```

getLogs start the logs and return the url of them

Parameters

<i>url</i>	the url of the log files if ip is the adress of the camera it will be : "http://" + ip + ":8080/tmp/logs.bin"
------------	---

Returns

a FliSdkError or FLISDK_NO_ERROR

6.3.3.27 getLogs() [2/2]

```
FliSdkError FliCred::getLogs (
    uint16_t nbDays,
    std::string & url )
```

getLogs start the logs and return the number of days last ones the url of them

Parameters

<i>nbDays</i>	the number of days to get the more recent logs
<i>url</i>	the url of the log files if ip is the adress of the camera it will be : "http://" + ip + ":8080/tmp/logs.bin"

Returns

a `FliSdkError` or `FLISDK_NO_ERROR`

6.3.3.28 getPassword()

```
FliSdkError FliCred::getPassword (
    std::string & password )
```

6.3.3.29 getStatus()

```
FliSdkError FliCred::getStatus (
    std::string & status )
```

6.3.3.30 getStatusDetailed()

```
FliSdkError FliCred::getStatusDetailed (
    std::string & status,
    std::string & diag )
```

6.3.3.31 getThresholdingOnOff()

```
FliSdkError FliCred::getThresholdingOnOff (
    bool & checked )
```

6.3.3.32 getUserConvolutionMatrix()

```
FliSdkError FliCred::getUserConvolutionMatrix (
    std::vector< std::vector< double > > & matrixBadPixels,
    double & divisor,
    std::string & description )
```

`FliCred_getUserConvolutionMatrix_V2` : method to get the convolution matrix from the camera.

Parameters

<i>matrixBadPixels</i>	: a vector of doubles to return the coefficients of the convolution matrix it will have a <code>Fli::MATRIX_FILTERING_SIZE * Fli::MATRIX_FILTERING_SIZE</code> size
<i>divisor</i>	: return the value of the total divisor coefficient as a double
<i>description</i>	: return the text that describe the matrix as a <code>char*</code>

Returns

a FliSdkError or FLISDK_NO_ERROR

6.3.3.33 getUserConvolutionMatrixIndex_V2()

```
FliSdkError FliCred::getUserConvolutionMatrixIndex_V2 (
    int & index )
```

getUserConvolutionMatrixIndex_V2 : get the current convolution matrix index

Parameters

<i>index</i>	: the current index
--------------	---------------------

Returns

a FliSdkError or FLISDK_NO_ERROR

6.3.3.34 getVersionFirmware()

```
FliSdkError FliCred::getVersionFirmware (
    std::string & version )
```

6.3.3.35 getVersionFirmwareBuild()

```
FliSdkError FliCred::getVersionFirmwareBuild (
    std::string & build )
```

6.3.3.36 getVersionFirmwareDetailed()

```
FliSdkError FliCred::getVersionFirmwareDetailed (
    std::string & detailed )
```

6.3.3.37 getVersionFpga()

```
FliSdkError FliCred::getVersionFpga (
    std::string & version )
```

6.3.3.38 getVersionHardware()

```
FliSdkError FliCred::getVersionHardware (
    std::string & version )
```

6.3.3.39 getVersions()

```
FliSdkError FliCred::getVersions (
    std::string & firmware,
    std::string & fpga,
    std::string & hardware )
```

6.3.3.40 restoreFactory()

```
FliSdkError FliCred::restoreFactory ( )
```

restoreFactory will restore the factory settings inside the camera as the user settings

Returns

a FliSdkError or FLISDK_NO_ERROR

6.3.3.41 saveCameraSettings()

```
FliSdkError FliCred::saveCameraSettings ( )
```

saveCameraSettings save all the user (not the factory) settings inside the camera

Returns

a FliSdkError or FLISDK_NO_ERROR

6.3.3.42 sendBiasFile()

```
FliSdkError FliCred::sendBiasFile (
    std::string filePath )
```

sendBiasFile send to the camera a bias file to the camera

Parameters

<i>filePath</i>	: the file name and path of the bias file to send to the camera
-----------------	---

Returns

a FliSdkError or FLISDK_NO_ERROR

6.3.3.43 sendBiasFromUrl()

```
FliSdkError FliCred::sendBiasFromUrl (
    std::string url )
```

sendBiasFromUrl send to the camera a bias file to the camera from an url

Parameters

<i>url</i>	: the url of the bias file to send to the camera
------------	--

Returns

a FliSdkError or FLISDK_NO_ERROR

6.3.3.44 sendFlatFile()

```
FliSdkError FliCred::sendFlatFile (
    std::string filePath )
```

sendFlatFile send to the camera a flat file to the camera

Parameters

<i>filePath</i>	: the file name and path of the flat file to send to the camera
-----------------	---

Returns**6.3.3.45 sendFlatFromUrl()**

```
FliSdkError FliCred::sendFlatFromUrl (
    std::string url )
```

sendFlatFromUrl send to the camera a flat file to the camera from an url

Parameters

<i>url</i>	: the url of the flat file to send to the camera
------------	--

Returns

a `FliSdkError` or `FLISDK_NO_ERROR`

6.3.3.46 setAduOffset()

```
FliSdkError FliCred::setAduOffset (
    int aduOffset )
```

6.3.3.47 setBadPixelModeOnOff()

```
FliSdkError FliCred::setBadPixelModeOnOff (
    bool checked )
```

6.3.3.48 setExcludeBorderOnOff()

```
FliSdkError FliCred::setExcludeBorderOnOff (
    bool exclude )
```

6.3.3.49 setFilteringModeOnOff()

```
FliSdkError FliCred::setFilteringModeOnOff (
    bool checked )
```

6.3.3.50 setIpAddress()

```
FliSdkError FliCred::setIpAddress (
    std::string ip )
```

6.3.3.51 setIpAlternateDns()

```
FliSdkError FliCred::setIpAlternateDns (
    std::string dns )
```

6.3.3.52 setIpAutomatic()

```
FliSdkError FliCred::setIpAutomatic ( )
```

6.3.3.53 setIpDns()

```
FliSdkError FliCred::setIpDns (
    std::string dns )
```

6.3.3.54 setIpGateway()

```
FliSdkError FliCred::setIpGateway (
    std::string gateway )
```

6.3.3.55 setIpManual()

```
FliSdkError FliCred::setIpManual ( )
```

6.3.3.56 setIpNetmask()

```
FliSdkError FliCred::setIpNetmask (
    std::string netmask )
```

6.3.3.57 setIpRefresh()

```
FliSdkError FliCred::setIpRefresh ( )
```

6.3.3.58 setKindOfBadPixelCorrection()

```
FliSdkError FliCred::setKindOfBadPixelCorrection (
    int correction )
```

setKindOfBadPixelCorrection : set the kind of correction to apply to bad pixels

Parameters

<i>correction</i>	: 1 = Low latency bad pixels correction; 2 = Convolution bad pixels correction
-------------------	--

Returns

a `FliSdkError` or `FLISDK_NO_ERROR`

6.3.3.59 setPassword()

```
FliSdkError FliCred::setPassword (
    std::string password )
```

6.3.3.60 setSlowMode()

```
FliSdkError FliCred::setSlowMode (
    bool slowMode )
```

6.3.3.61 setThresholdingLevelsValues()

```
FliSdkError FliCred::setThresholdingLevelsValues (
    int lowLevel,
    int highLevel,
    int lowValue,
    int middleValue,
    int highValue )
```

`setThresholdingLevelsValues` set the levels and values of the thresholding

Parameters

<i>lowLevel</i>	the low level of ADU
<i>highLevel</i>	the high level of ADU
<i>lowValue</i>	the value to be set for all the values below the low level
<i>middleValue</i>	the value to be set for all the values between the low level and the high level
<i>highValue</i>	the value to be set for all the values above the high level

Returns

a `FliSdkError` or `FLISDK_NO_ERROR`

6.3.3.62 setThresholdingOnOff()

```
FliSdkError FliCred::setThresholdingOnOff (
    bool checked )
```

6.3.3.63 setUserConvolutionMatrix()

```
FliSdkError FliCred::setUserConvolutionMatrix (
    std::vector< std::vector< double > > & matrixBadPixels,
    double divisor,
    std::string & description )
```

FliCred_setUserConvolutionMatrix_V2 : method to set the convolution matrix inside the camera.

Parameters

<i>matrixBadPixels</i>	: a vector of doubles to give the coefficients of the convolution matrix it will have a Fli::MATRIX_FILTERING_SIZE * Fli::MATRIX_FILTERING_SIZE size
<i>divisor</i>	: the value of the total divisor coefficient
<i>description</i>	: the text that describe the matrix

Returns

a FliSdkError or FLISDK_NO_ERROR

6.3.3.64 setUserConvolutionMatrixIndex_V2()

```
FliSdkError FliCred::setUserConvolutionMatrixIndex_V2 (
    const int index )
```

setUserConvolutionMatrixIndex_V2 : set the index of the current convolution matrix

Parameters

<i>index</i>	: the index to set
--------------	--------------------

Returns

a FliSdkError or FLISDK_NO_ERROR

6.3.3.65 shutDown()

```
FliSdkError FliCred::shutDown ( )
```

shutDown will shut down the camera

Returns

a `FliSdkError` or `FLISDK_NO_ERROR`

6.3.3.66 upgradeFirmware()

```
FliSdkError FliCred::upgradeFirmware (
    std::string url )
```

`upgradeFirmware` will upgrade the Firmware inside the camera from the url of a firmware file

Parameters

<i>url</i>	: the url of the firmware file to be updated
------------	--

Returns

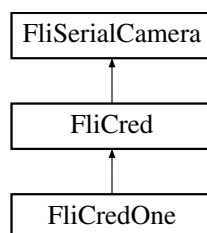
a `FliSdkError` or `FLISDK_NO_ERROR`

6.4 FliCredOne Class Reference

This class manages the methods specific to the C-RED One camera.

```
#include <FliCredOne.h>
```

Inheritance diagram for `FliCredOne`:

**Public Types**

- enum `Mode` {
 - `undefined`, `globalResetSingle`, `globalResetBursts`, `globalResetCds`, `rollingResetlota`, `rollingResetNro`, `rollingResetSingle` }

Public Member Functions

- [FliCredOne](#) (IFrameGrabberCL *grabber)
- FliSdkError [getCropping](#) (bool &enabled, std::string &columns, std::string &rows)
- FliSdkError [getAllTemp](#) (double &mb, double &fe, double &pw, double &cryod, double &cryopt, double &water, double &peltier, double &ptmcu)
- FliSdkError [getNbReadWoReset](#) (int &nbRead)
- FliSdkError [getRawImagesState](#) (bool &enabled)
- FliSdkError [getTempFrontEnd](#) (double &temp)
- FliSdkError [getTempMotherBoard](#) (double &temp)
- FliSdkError [getTempPowerBoard](#) (double &temp)
- FliSdkError [getTempDiode](#) (double &temp)
- FliSdkError [getTempPtController](#) (double &temp)
- FliSdkError [getTempSetpoint](#) (double &temp)
- FliSdkError [getTempPtMcu](#) (double &temp)
- FliSdkError [getTempWater](#) (double &temp)
- FliSdkError [getVersionFpgaDetailed](#) (std::string &detailed)
- FliSdkError [getAll](#) (std::string &all)
- FliSdkError [getCoolingState](#) (bool &enabled)
- FliSdkError [getGain](#) (double &gain)
- FliSdkError [getNbRegenGetter](#) (std::string ®enInfo)
- FliSdkError [getRegenRemainingTime](#) (int &time)
- FliSdkError [getReadOutMode](#) ([Mode](#) &mode)
- FliSdkError [getNloop](#) (int &nLoop)
- FliSdkError [getNbSamplePixel](#) (int &nSample)
- FliSdkError [getPhotoCurrent](#) (double &photocurrent)
- FliSdkError [getPowers](#) (double &getter, double &peltier, double &pulseTube)
- FliSdkError [getPowerGetter](#) (double &getter)
- FliSdkError [getPowerPulseTube](#) (double &pulseTube)
- FliSdkError [getPressure](#) (std::string &pressure)
- FliSdkError [getPulseTubeReady](#) (std::string &info)
- FliSdkError [getRemoteMaintenanceState](#) (std::string &status)
- FliSdkError [getResetWidth](#) (int &width)
- FliSdkError [getStandbyState](#) (bool &enabled)
- FliSdkError [getTestPatternState](#) (bool &enabled)
- FliSdkError [getTelnetState](#) (bool &enabled)
- FliSdkError [getFowlerState](#) (bool &enabled)
- FliSdkError [setCropping](#) (bool enable, std::string columns, std::string rows)
- FliSdkError [setCroppingColumns](#) (std::string columns)
- FliSdkError [setCroppingRows](#) (std::string rows)
- FliSdkError [setNbReadWoReset](#) (int nbRead)
- FliSdkError [setGain](#) (double gain)
- FliSdkError [setMode](#) ([Mode](#) mode)
- FliSdkError [setNloop](#) (int nLoop)
- FliSdkError [setNsamplePixel](#) (int nSample)
- FliSdkError [setResetWidth](#) (int resetWidth)
- FliSdkError [enableRawImages](#) (bool enable)
- FliSdkError [enableRemoteMaintenance](#) (bool enable)
- FliSdkError [enableCooling](#) (bool enable)
- FliSdkError [enableStandby](#) (bool enable)
- FliSdkError [enableTestPattern](#) (bool enable)
- FliSdkError [enableFowler](#) (bool enable)
- FliSdkError [startVacuumRegen](#) ()
 - *startVacuumRegen the camera will try to redo a vacuum*
- FliSdkError [sendTestPatternFromUrl](#) (std::string url)

sendTestPatternFromUrl send the test patterns to the camera from a url

- FliSdkError [reboot](#) ()

reboot will force the camera to reboot

- FliSdkError [isCroppingValid](#) (std::string columns, std::string rows)

isCroppingValid check against a given *Regex* pattern, the values of the columns and rows of a cropping

Additional Inherited Members

6.4.1 Detailed Description

This class manages the methods specific to the C-RED One camera.

6.4.2 Member Enumeration Documentation

6.4.2.1 Mode

```
enum FliCredOne::Mode
```

Enumerator

undefined	
globalResetSingle	
globalResetBursts	
globalResetCds	
rollingResetlota	
rollingResetNro	
rollingResetSingle	

6.4.3 Constructor & Destructor Documentation

6.4.3.1 FliCredOne()

```
FliCredOne::FliCredOne (
    IFrameGrabberCL * grabber )
```

6.4.4 Member Function Documentation

6.4.4.1 enableCooling()

```
FliSdkError FliCredOne::enableCooling (
    bool enable )
```

6.4.4.2 enableFowler()

```
FliSdkError FliCredOne::enableFowler (
    bool enable )
```

6.4.4.3 enableRawImages()

```
FliSdkError FliCredOne::enableRawImages (
    bool enable )
```

6.4.4.4 enableRemoteMaintenance()

```
FliSdkError FliCredOne::enableRemoteMaintenance (
    bool enable )
```

6.4.4.5 enableStandby()

```
FliSdkError FliCredOne::enableStandby (
    bool enable )
```

6.4.4.6 enableTestPattern()

```
FliSdkError FliCredOne::enableTestPattern (
    bool enable )
```

6.4.4.7 getAll()

```
FliSdkError FliCredOne::getAll (
    std::string & all )
```

6.4.4.8 getAllTemp()

```
FliSdkError FliCredOne::getAllTemp (
    double & mb,
    double & fe,
    double & pw,
    double & cryod,
    double & cryopt,
    double & water,
    double & peltier,
    double & ptmcu )
```

6.4.4.9 getCoolingState()

```
FliSdkError FliCredOne::getCoolingState (
    bool & enabled )
```

6.4.4.10 getCropping()

```
FliSdkError FliCredOne::getCropping (
    bool & enabled,
    std::string & columns,
    std::string & rows )
```

6.4.4.11 getFowlerState()

```
FliSdkError FliCredOne::getFowlerState (
    bool & enabled )
```

6.4.4.12 getGain()

```
FliSdkError FliCredOne::getGain (
    double & gain )
```

6.4.4.13 getNbReadWoReset()

```
FliSdkError FliCredOne::getNbReadWoReset (
    int & nbRead )
```

6.4.4.14 getNbRegenGetter()

```
FliSdkError FliCredOne::getNbRegenGetter (
    std::string & regenInfo )
```

6.4.4.15 getNbSamplePixel()

```
FliSdkError FliCredOne::getNbSamplePixel (
    int & nSample )
```

6.4.4.16 getNloop()

```
FliSdkError FliCredOne::getNloop (
    int & nLoop )
```

6.4.4.17 getPhotoCurrent()

```
FliSdkError FliCredOne::getPhotoCurrent (
    double & photocurrent )
```

6.4.4.18 getPowerGetter()

```
FliSdkError FliCredOne::getPowerGetter (
    double & getter )
```

6.4.4.19 getPowerPulseTube()

```
FliSdkError FliCredOne::getPowerPulseTube (
    double & pulseTube )
```

6.4.4.20 getPowers()

```
FliSdkError FliCredOne::getPowers (
    double & getter,
    double & peltier,
    double & pulseTube )
```

6.4.4.21 getPressure()

```
FliSdkError FliCredOne::getPressure (
    std::string & pressure )
```

6.4.4.22 getPulseTubeReady()

```
FliSdkError FliCredOne::getPulseTubeReady (
    std::string & info )
```

6.4.4.23 getRawImagesState()

```
FliSdkError FliCredOne::getRawImagesState (
    bool & enabled )
```

6.4.4.24 getReadOutMode()

```
FliSdkError FliCredOne::getReadOutMode (
    Mode & mode )
```

6.4.4.25 getRegenRemainingTime()

```
FliSdkError FliCredOne::getRegenRemainingTime (
    int & time )
```

6.4.4.26 getRemoteMaintenanceState()

```
FliSdkError FliCredOne::getRemoteMaintenanceState (
    std::string & status )
```

6.4.4.27 getResetWidth()

```
FliSdkError FliCredOne::getResetWidth (
    int & width )
```


6.4.4.28 getStandbyState()

```
FliSdkError FliCredOne::getStandbyState (
    bool & enabled )
```

6.4.4.29 getTelnetState()

```
FliSdkError FliCredOne::getTelnetState (
    bool & enabled )
```

6.4.4.30 getTempDiode()

```
FliSdkError FliCredOne::getTempDiode (
    double & temp )
```

6.4.4.31 getTempFrontEnd()

```
FliSdkError FliCredOne::getTempFrontEnd (
    double & temp )
```

6.4.4.32 getTempMotherBoard()

```
FliSdkError FliCredOne::getTempMotherBoard (
    double & temp )
```

6.4.4.33 getTempPowerBoard()

```
FliSdkError FliCredOne::getTempPowerBoard (
    double & temp )
```

6.4.4.34 getTempPtController()

```
FliSdkError FliCredOne::getTempPtController (
    double & temp )
```

6.4.4.35 getTempPtMcu()

```
FliSdkError FliCredOne::getTempPtMcu (
    double & temp )
```

6.4.4.36 getTempSetpoint()

```
FliSdkError FliCredOne::getTempSetpoint (
    double & temp )
```

6.4.4.37 getTempWater()

```
FliSdkError FliCredOne::getTempWater (
    double & temp )
```

6.4.4.38 getTestPatternState()

```
FliSdkError FliCredOne::getTestPatternState (
    bool & enabled )
```

6.4.4.39 getVersionFpgaDetailed()

```
FliSdkError FliCredOne::getVersionFpgaDetailed (
    std::string & detailed )
```

6.4.4.40 isCroppingValid()

```
FliSdkError FliCredOne::isCroppingValid (
    std::string columns,
    std::string rows )
```

isCroppingValid check against a given Regex pattern, the values of the columns and rows of a cropping

Parameters

<i>columns</i>	the values of the columns
<i>rows</i>	the values of the rows

Returns

a `FliSdkError` the state if columns and/or rows are bad or `FLISDK_NO_ERROR`

6.4.4.41 reboot()

```
FliSdkError FliCredOne::reboot ( )
```

`reboot` will force the camera to reboot

Returns

a `FliSdkError` or `FLISDK_NO_ERROR`

6.4.4.42 sendTestPatternFromUrl()

```
FliSdkError FliCredOne::sendTestPatternFromUrl (
    std::string url )
```

`sendTestPatternFromUrl` send the test patterns to the camera from a url

Parameters

<code>url</code>	: the url where are the test patterns file
------------------	--

Returns

a `FliSdkError` or `FLISDK_NO_ERROR`

6.4.4.43 setCropping()

```
FliSdkError FliCredOne::setCropping (
    bool enable,
    std::string columns,
    std::string rows )
```

6.4.4.44 setCroppingColumns()

```
FliSdkError FliCredOne::setCroppingColumns (
    std::string columns )
```

6.4.4.45 setCroppingRows()

```
FliSdkError FliCredOne::setCroppingRows (
    std::string rows )
```

6.4.4.46 setGain()

```
FliSdkError FliCredOne::setGain (
    double gain )
```

6.4.4.47 setMode()

```
FliSdkError FliCredOne::setMode (
    Mode mode )
```

6.4.4.48 setNbReadWoReset()

```
FliSdkError FliCredOne::setNbReadWoReset (
    int nbRead )
```

6.4.4.49 setNloop()

```
FliSdkError FliCredOne::setNloop (
    int nLoop )
```

6.4.4.50 setNsamplePixel()

```
FliSdkError FliCredOne::setNsamplePixel (
    int nSample )
```

6.4.4.51 setResetWidth()

```
FliSdkError FliCredOne::setResetWidth (
    int resetWidth )
```

6.4.4.52 startVacuumRegen()

```
FliSdkError FliCredOne::startVacuumRegen ( )
```

startVacuumRegen the camera will try to redo a vacuum

Returns

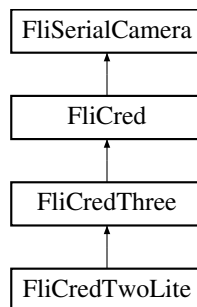
a FliSdkError or FLISDK_NO_ERROR

6.5 FliCredThree Class Reference

This class manages the methods specific to the C-RED 3 camera.

```
#include <FliCredThree.h>
```

Inheritance diagram for FliCredThree:



Public Types

- enum [AgcParam](#) {
[level_pix_high_hg](#), [level_pix_high_mg](#), [level_pix_low_lg](#), [level_pix_low_mg](#),
[trigger_nb_frames_hg_to_mg](#), [trigger_nb_frames_lg_to_mg](#), [trigger_nb_frames_mg_to_hg](#), [trigger_nb_frames_mg_to_lg](#),
[trigger_ratio_pixels_hg_to_mg](#), [trigger_ratio_pixels_lg_to_mg](#), [trigger_ratio_pixels_mg_to_hg](#), [trigger_ratio_pixels_mg_to_lg](#)
}

Public Member Functions

- [FliCredThree](#) (IFrameGrabberCL *grabber, bool isOem=false)
- FliSdkError [getCropping](#) (bool &enabled, uint16_t &col1, uint16_t &col2, uint16_t &row1, uint16_t &row2)
- FliSdkError [getAllTemp](#) (double &cpu, double &backend, double &interfaceTemp, double &ambient, double &sensor)
- FliSdkError [getTempAmbiant](#) (double &temp)
- FliSdkError [getTempBackEnd](#) (double &temp)
- FliSdkError [getTempCpu](#) (double &temp)
- FliSdkError [getTempInterface](#) (double &temp)
- FliSdkError [getTempSnake](#) (double &temp)
- FliSdkError [getTint](#) (double &tint)
- FliSdkError [getTintRange](#) (double &tintMin, double &tintMax)
- FliSdkError [getBadPixelState](#) (bool &enabled)

- FliSdkError [getAdaptBiasState](#) (bool &enabled)
- FliSdkError [getAgcState](#) (bool &enabled)
- FliSdkError [getConversionGain](#) (std::string &conversionGain)
- FliSdkError [getAntiBloomingState](#) (bool &enabled)
- FliSdkError [getAgcPriority](#) (std::string &priority)
- FliSdkError [getAgcRoi](#) (uint16_t &col1, uint16_t &row1, uint16_t &col2, uint16_t &row2)
- FliSdkError [getDarkOptimLevel](#) (int &level)
- FliSdkError [getExtSynchroExposure](#) (std::string &exposure)
- FliSdkError [getExtSynchroPolarity](#) (std::string &polarity)
- FliSdkError [getTuning](#) (std::string &tuning)
- FliSdkError [getHdrState](#) (bool &enabled)
- FliSdkError [getHdrCalibrationMode](#) (std::string &mode)
- FliSdkError [getHdrExtendedState](#) (bool &enabled)
- FliSdkError [getLicenses](#) (std::vector< std::string > &licenses)
- FliSdkError [getMaxFpsUsb](#) (double &maxFpsUsb)
- FliSdkError [getMaxSyncDelay](#) (double &maxSyncDelay)
- FliSdkError [getMinSyncDelay](#) (double &minSyncDelay)
- FliSdkError [getMaxTintItr](#) (double &maxTintItr)
- FliSdkError [getMinFps](#) (double &minFps)
- FliSdkError [getNbFramesPerSwTrig](#) (int &nbFrames)
- FliSdkError [getPreset](#) (int &preset)
- FliSdkError [getRemoteMaintenanceState](#) (bool &enabled)
- FliSdkError [getStepSyncDelay](#) (double &delay)
- FliSdkError [getSwSynchroState](#) (bool &enabled)
- FliSdkError [getSyncDelay](#) (double &delay)
- FliSdkError [getTcadsAdjustState](#) (bool &enabled)
- FliSdkError [getTelnetState](#) (bool &enabled)
- FliSdkError [getTintGranularityState](#) (bool &enabled)
- FliSdkError [getTlsydel](#) (int &val)
- FliSdkError [getVrefAdjustState](#) (bool &enabled)
- FliSdkError [getAgcParam](#) ([AgcParam](#) param, double &value)
- FliSdkError [getIpAlternateDns](#) (std::string &dns)
- FliSdkError [getIpDns](#) (std::string &dns)
- FliSdkError [getIpGateway](#) (std::string &gateway)
- FliSdkError [getIpMode](#) (std::string &mode)
- FliSdkError [getIpNetmask](#) (std::string &netmask)
- FliSdkError [getIpAddress](#) (std::string &ip)
- FliSdkError [getSnakeParam](#) (std::string parameter, uint16_t &value)
- FliSdkError [getUploadFirmwareConnectionInfo](#) (std::string &ip, uint16_t &port)
- FliSdkError [getTriggerSource](#) (std::string &source)
- FliSdkError [getSyncSignalSource](#) (std::string &source)
- FliSdkError [getExtMarkerSource](#) (std::string &source)
- FliSdkError [getHardwareFeatures](#) (int &features)
- FliSdkError [getSoftwareFeatures](#) (int &features)
- FliSdkError [getFactoryBadPixelMap](#) (std::vector< bool > &map, std::function< void(int)> getProgress=nullptr)
- FliSdkError [getUserBadPixelMap](#) (std::vector< bool > &map, std::function< void(int)> getProgress=nullptr)
- FliSdkError [getTintStep](#) (double &step)
- FliSdkError [getImagePattern](#) (std::string &pattern)
- FliSdkError [getDate](#) (std::string &date)
- FliSdkError [getUptime](#) (std::string &uptime)
- FliSdkError [getAccumulatedUptime](#) (std::string &uptime)
- FliSdkError [getTotalUptime](#) (std::string &uptime)
- FliSdkError [getRawImagesState](#) (bool &enabled)
- FliSdkError [getUnsignedPixelsState](#) (bool &enabled)

- FliSdkError [setCropping](#) (bool enable, uint16_t col1, uint16_t col2, uint16_t row1, uint16_t row2)
- FliSdkError [setCroppingColumns](#) (uint16_t col1, uint16_t col2)
- FliSdkError [setCroppingRows](#) (uint16_t row1, uint16_t row2)
- FliSdkError [setTint](#) (double tint)
- FliSdkError [setConversionGainHigh](#) ()
- FliSdkError [setConversionGainMedium](#) ()
- FliSdkError [setConversionGainLow](#) ()
- FliSdkError [setAgcPriorityNone](#) ()
- FliSdkError [setAgcPriorityOverExposed](#) ()
- FliSdkError [setAgcPriorityUnderExposed](#) ()
- FliSdkError [setDarkOptimLevel](#) (int level)
- FliSdkError [setExtSynchroExposureInternal](#) ()
- FliSdkError [setExtSynchroExposureExternal](#) ()
- FliSdkError [setExtSynchroPolarityInverted](#) ()
- FliSdkError [setExtSynchroPolarityStandard](#) ()
- FliSdkError [setTuningGeneralUse](#) ()
- FliSdkError [setTuningShortExposure](#) ()
- FliSdkError [setTuningLongExposure](#) ()
- FliSdkError [setHdrCalibrationC1](#) ()
- FliSdkError [setHdrCalibrationC2](#) ()
- FliSdkError [setHdrCalibrationOff](#) ()
- FliSdkError [setNbFramesPerSwTrig](#) (uint16_t nbFrames)
- FliSdkError [setSyncDelay](#) (int delay)
- FliSdkError [setTlSyDel](#) (int val)
- FliSdkError [setVoltageVref](#) (double vref)
- FliSdkError [setAgcRoi](#) (uint16_t col1, uint16_t row1, uint16_t col2, uint16_t row2)
- FliSdkError [setAgcParam](#) (AgcParam param, double value)
- FliSdkError [setPreset](#) ()
- FliSdkError [setPresetNumber](#) (uint8_t presetNumber)
- FliSdkError [setSnakeParam](#) (std::string parameter, uint16_t value)
- FliSdkError [setTriggerSourceSoftware](#) ()
- FliSdkError [setTriggerSourceExternal](#) ()
- FliSdkError [setSyncSignalSourceExternal](#) ()
- FliSdkError [setSyncSignalSourceCC1](#) ()
- FliSdkError [setSyncSignalSourceCC2](#) ()
- FliSdkError [setSyncSignalSourceCC3](#) ()
- FliSdkError [setSyncSignalSourceCC4](#) ()
- FliSdkError [setFrameMarkerSourceExternal](#) ()
- FliSdkError [setFrameMarkerSourceCC1](#) ()
- FliSdkError [setFrameMarkerSourceCC2](#) ()
- FliSdkError [setFrameMarkerSourceCC3](#) ()
- FliSdkError [setFrameMarkerSourceCC4](#) ()
- FliSdkError [setFactoryBadPixelMap](#) (std::vector< bool > &map)
- FliSdkError [setUserBadPixelMap](#) (std::vector< bool > &map)
- FliSdkError [setImagePatternRamp](#) ()
- FliSdkError [setImagePatternConstant](#) (uint16_t val)
- FliSdkError [setImagePatternOff](#) ()
- FliSdkError [enableBadPixel](#) (bool enable)
- FliSdkError [enableAdaptBias](#) (bool enable)
- FliSdkError [enableAgc](#) (bool enable)
- FliSdkError [enableAntiBlooming](#) (bool enable)
- FliSdkError [enableHdrExtended](#) (bool enable)
- FliSdkError [enableHdr](#) (bool enable)
- FliSdkError [enableRemoteMaintenance](#) (bool enable)
- FliSdkError [enableSwSynchro](#) (bool enable)

- FliSdkError [enableTcdsAdjust](#) (bool enable)
- FliSdkError [enableTintGranularity](#) (bool enable)
- FliSdkError [enableVrefAdjust](#) (bool enable)
- FliSdkError [enableRawImages](#) (bool enable)
- FliSdkError [enableUnsignedPixels](#) (bool enable)
- FliSdkError [reboot](#) ()
- FliSdkError [buildFlatHdrC1](#) ()
- FliSdkError [buildFlatHdrC2](#) ()
- FliSdkError [sendBiasHdrC1FromUrl](#) (std::string url)
- FliSdkError [sendBiasHdrC2FromUrl](#) (std::string url)
- FliSdkError [sendFlatHdrC1FromUrl](#) (std::string url)
- FliSdkError [sendFlatHdrC2FromUrl](#) (std::string url)
- FliSdkError [sendBiasHdrC1File](#) (std::string filePath)
- FliSdkError [sendBiasHdrC2File](#) (std::string filePath)
- FliSdkError [sendFlatHdrC1File](#) (std::string filePath)
- FliSdkError [sendFlatHdrC2File](#) (std::string filePath)
- FliSdkError [xSendBiasFile](#) (std::string filePath, std::function< void(bool, int, int)> getBlockStatus=nullptr)
- FliSdkError [xSendBiasHdrC1File](#) (std::string filePath, std::function< void(bool, int, int)> getBlock↵
Status=nullptr)
- FliSdkError [xSendBiasHdrC2File](#) (std::string filePath, std::function< void(bool, int, int)> getBlock↵
Status=nullptr)
- FliSdkError [xSendFlatFile](#) (std::string filePath, std::function< void(bool, int, int)> getBlockStatus=nullptr)
- FliSdkError [xSendFlatHdrC1File](#) (std::string filePath, std::function< void(bool, int, int)> getBlock↵
Status=nullptr)
- FliSdkError [xSendFlatHdrC2File](#) (std::string filePath, std::function< void(bool, int, int)> getBlock↵
Status=nullptr)
- FliSdkError [xSendLicenseFile](#) (std::string filePath, std::string fileName, std::function< void(bool, int, int)>
getBlockStatus=nullptr)
- FliSdkError [sendLicenseFile](#) (std::string filePath, std::string fileName)
- FliSdkError [deleteLicense](#) (std::string licenseName)
- FliSdkError [disableLicense](#) (std::string licenseName)
- FliSdkError [enableLicense](#) (std::string licenseName)
- FliSdkError [softwareTrig](#) ()
- FliSdkError [sendBadPixelFile](#) (std::string filePath)
- FliSdkError [xSendBadPixelFile](#) (std::string filePath, std::function< void(bool, int, int)> getBlockStatus=nullptr)
- FliSdkError [sendBadPixelFromUrl](#) (std::string url)
- FliSdkError [isCroppingValid](#) (uint16_t col1, uint16_t col2, uint16_t row1, uint16_t row2)
- FliSdkError [buildBiasNuc](#) (uint16_t nblImages=256)
- FliSdkError [buildFlatNuc](#) (uint16_t nblImages=256)
- FliSdkError [buildFlatHdrC1Nuc](#) (uint16_t nblImages=256)
- FliSdkError [buildFlatHdrC2Nuc](#) (uint16_t nblImages=256)
- FliSdkError [abortBuildNuc](#) ()
- FliSdkError [getBuildNucProgress](#) (int &progress)
- FliSdkError [startHttpServer](#) ()
- FliSdkError [stopHttpServer](#) ()
- FliSdkError [startEthernetGrabber](#) ()
- FliSdkError [stopEthernetGrabber](#) ()

Additional Inherited Members

6.5.1 Detailed Description

This class manages the methods specific to the C-RED 3 camera.

6.5.2 Member Enumeration Documentation

6.5.2.1 AgcParam

```
enum FliCredThree::AgcParam
```

Enumerator

level_pix_high_hg	
level_pix_high_mg	
level_pix_low_lg	
level_pix_low_mg	
trigger_nb_frames_hg_to_mg	
trigger_nb_frames_lg_to_mg	
trigger_nb_frames_mg_to_hg	
trigger_nb_frames_mg_to_lg	
trigger_ratio_pixels_hg_to_mg	
trigger_ratio_pixels_lg_to_mg	
trigger_ratio_pixels_mg_to_hg	
trigger_ratio_pixels_mg_to_lg	

6.5.3 Constructor & Destructor Documentation

6.5.3.1 FliCredThree()

```
FliCredThree::FliCredThree (
    IFrameGrabberCL * grabber,
    bool isOem = false )
```

6.5.4 Member Function Documentation

6.5.4.1 abortBuildNuc()

```
FliSdkError FliCredThree::abortBuildNuc ( )
```

6.5.4.2 buildBiasNuc()

```
FliSdkError FliCredThree::buildBiasNuc (
    uint16_t nbImages = 256 )
```

6.5.4.3 buildFlatHdrC1()

```
FliSdkError FliCredThree::buildFlatHdrC1 ( )
```

6.5.4.4 buildFlatHdrC1Nuc()

```
FliSdkError FliCredThree::buildFlatHdrC1Nuc (
    uint16_t nbImages = 256 )
```

6.5.4.5 buildFlatHdrC2()

```
FliSdkError FliCredThree::buildFlatHdrC2 ( )
```

6.5.4.6 buildFlatHdrC2Nuc()

```
FliSdkError FliCredThree::buildFlatHdrC2Nuc (
    uint16_t nbImages = 256 )
```

6.5.4.7 buildFlatNuc()

```
FliSdkError FliCredThree::buildFlatNuc (
    uint16_t nbImages = 256 )
```

6.5.4.8 deleteLicense()

```
FliSdkError FliCredThree::deleteLicense (
    std::string licenseName )
```

6.5.4.9 disableLicense()

```
FliSdkError FliCredThree::disableLicense (
    std::string licenseName )
```

6.5.4.10 enableAdaptBias()

```
FliSdkError FliCredThree::enableAdaptBias (
    bool enable )
```

6.5.4.11 enableAgc()

```
FliSdkError FliCredThree::enableAgc (
    bool enable )
```

6.5.4.12 enableAntiBlooming()

```
FliSdkError FliCredThree::enableAntiBlooming (
    bool enable )
```

6.5.4.13 enableBadPixel()

```
FliSdkError FliCredThree::enableBadPixel (
    bool enable )
```

6.5.4.14 enableHdr()

```
FliSdkError FliCredThree::enableHdr (
    bool enable )
```

6.5.4.15 enableHdrExtended()

```
FliSdkError FliCredThree::enableHdrExtended (
    bool enable )
```

6.5.4.16 enableLicense()

```
FliSdkError FliCredThree::enableLicense (
    std::string licenseName )
```

6.5.4.17 enableRawImages()

```
FliSdkError FliCredThree::enableRawImages (
    bool enable )
```

6.5.4.18 enableRemoteMaintenance()

```
FliSdkError FliCredThree::enableRemoteMaintenance (
    bool enable )
```

6.5.4.19 enableSwSynchro()

```
FliSdkError FliCredThree::enableSwSynchro (
    bool enable )
```

6.5.4.20 enableTcdsAdjust()

```
FliSdkError FliCredThree::enableTcdsAdjust (
    bool enable )
```

6.5.4.21 enableTintGranularity()

```
FliSdkError FliCredThree::enableTintGranularity (
    bool enable )
```

6.5.4.22 enableUnsignedPixels()

```
FliSdkError FliCredThree::enableUnsignedPixels (
    bool enable )
```

6.5.4.23 enableVrefAdjust()

```
FliSdkError FliCredThree::enableVrefAdjust (
    bool enable )
```

6.5.4.24 getAccumulatedUptime()

```
FliSdkError FliCredThree::getAccumulatedUptime (
    std::string & uptime )
```

6.5.4.25 getAdaptBiasState()

```
FliSdkError FliCredThree::getAdaptBiasState (
    bool & enabled )
```

6.5.4.26 getAgcParam()

```
FliSdkError FliCredThree::getAgcParam (
    AgcParam param,
    double & value )
```

6.5.4.27 getAgcPriority()

```
FliSdkError FliCredThree::getAgcPriority (
    std::string & priority )
```

6.5.4.28 getAgcRoi()

```
FliSdkError FliCredThree::getAgcRoi (
    uint16_t & col1,
    uint16_t & row1,
    uint16_t & col2,
    uint16_t & row2 )
```

6.5.4.29 getAgcState()

```
FliSdkError FliCredThree::getAgcState (
    bool & enabled )
```

6.5.4.30 getAllTemp()

```
FliSdkError FliCredThree::getAllTemp (
    double & cpu,
    double & backend,
    double & interfaceTemp,
    double & ambient,
    double & sensor )
```

6.5.4.31 getAntiBloomingState()

```
FliSdkError FliCredThree::getAntiBloomingState (
    bool & enabled )
```

6.5.4.32 getBadPixelState()

```
FliSdkError FliCredThree::getBadPixelState (
    bool & enabled )
```

6.5.4.33 getBuildNucProgress()

```
FliSdkError FliCredThree::getBuildNucProgress (
    int & progress )
```

6.5.4.34 getConversionGain()

```
FliSdkError FliCredThree::getConversionGain (
    std::string & conversionGain )
```

6.5.4.35 getCropping()

```
FliSdkError FliCredThree::getCropping (
    bool & enabled,
    uint16_t & col1,
    uint16_t & col2,
    uint16_t & row1,
    uint16_t & row2 )
```

6.5.4.36 getDarkOptimLevel()

```
FliSdkError FliCredThree::getDarkOptimLevel (
    int & level )
```

6.5.4.37 getDate()

```
FliSdkError FliCredThree::getDate (
    std::string & date )
```

6.5.4.38 getExtMarkerSource()

```
FliSdkError FliCredThree::getExtMarkerSource (
    std::string & source )
```

6.5.4.39 getExtSynchroExposure()

```
FliSdkError FliCredThree::getExtSynchroExposure (
    std::string & exposure )
```

6.5.4.40 getExtSynchroPolarity()

```
FliSdkError FliCredThree::getExtSynchroPolarity (
    std::string & polarity )
```

6.5.4.41 getFactoryBadPixelMap()

```
FliSdkError FliCredThree::getFactoryBadPixelMap (
    std::vector< bool > & map,
    std::function< void(int)> getProgress = nullptr )
```

6.5.4.42 getHardwareFeatures()

```
FliSdkError FliCredThree::getHardwareFeatures (
    int & features )
```

6.5.4.43 getHdrCalibrationMode()

```
FliSdkError FliCredThree::getHdrCalibrationMode (
    std::string & mode )
```

6.5.4.44 getHdrExtendedState()

```
FliSdkError FliCredThree::getHdrExtendedState (
    bool & enabled )
```

6.5.4.45 getHdrState()

```
FliSdkError FliCredThree::getHdrState (
    bool & enabled )
```

6.5.4.46 getImagePattern()

```
FliSdkError FliCredThree::getImagePattern (
    std::string & pattern )
```

6.5.4.47 getIpAddress()

```
FliSdkError FliCredThree::getIpAddress (
    std::string & ip )
```


6.5.4.48 getIpAlternateDns()

```
FliSdkError FliCredThree::getIpAlternateDns (
    std::string & dns )
```

6.5.4.49 getIpDns()

```
FliSdkError FliCredThree::getIpDns (
    std::string & dns )
```

6.5.4.50 getIpGateway()

```
FliSdkError FliCredThree::getIpGateway (
    std::string & gateway )
```

6.5.4.51 getIpMode()

```
FliSdkError FliCredThree::getIpMode (
    std::string & mode )
```

6.5.4.52 getIpNetmask()

```
FliSdkError FliCredThree::getIpNetmask (
    std::string & netmask )
```

6.5.4.53 getLicenses()

```
FliSdkError FliCredThree::getLicenses (
    std::vector< std::string > & licenses )
```

6.5.4.54 getMaxFpsUsb()

```
FliSdkError FliCredThree::getMaxFpsUsb (
    double & maxFpsUsb )
```

6.5.4.55 getMaxSyncDelay()

```
FliSdkError FliCredThree::getMaxSyncDelay (
    double & maxSyncDelay )
```

6.5.4.56 getMaxTintItr()

```
FliSdkError FliCredThree::getMaxTintItr (
    double & maxTintItr )
```

6.5.4.57 getMinFps()

```
FliSdkError FliCredThree::getMinFps (
    double & minFps )
```

6.5.4.58 getMinSyncDelay()

```
FliSdkError FliCredThree::getMinSyncDelay (
    double & minSyncDelay )
```

6.5.4.59 getNbFramesPerSwTrig()

```
FliSdkError FliCredThree::getNbFramesPerSwTrig (
    int & nbFrames )
```

6.5.4.60 getPreset()

```
FliSdkError FliCredThree::getPreset (
    int & preset )
```

6.5.4.61 getRawImagesState()

```
FliSdkError FliCredThree::getRawImagesState (
    bool & enabled )
```

6.5.4.62 getRemoteMaintenanceState()

```
FliSdkError FliCredThree::getRemoteMaintenanceState (
    bool & enabled )
```

6.5.4.63 getSnakeParam()

```
FliSdkError FliCredThree::getSnakeParam (
    std::string parameter,
    uint16_t & value )
```

6.5.4.64 getSoftwareFeatures()

```
FliSdkError FliCredThree::getSoftwareFeatures (
    int & features )
```

6.5.4.65 getStepSyncDelay()

```
FliSdkError FliCredThree::getStepSyncDelay (
    double & delay )
```

6.5.4.66 getSwSynchroState()

```
FliSdkError FliCredThree::getSwSynchroState (
    bool & enabled )
```

6.5.4.67 getSyncDelay()

```
FliSdkError FliCredThree::getSyncDelay (
    double & delay )
```

6.5.4.68 getSyncSignalSource()

```
FliSdkError FliCredThree::getSyncSignalSource (
    std::string & source )
```

6.5.4.69 getTcdsAdjustState()

```
FliSdkError FliCredThree::getTcdsAdjustState (
    bool & enabled )
```

6.5.4.70 getTelnetState()

```
FliSdkError FliCredThree::getTelnetState (
    bool & enabled )
```

6.5.4.71 getTempAmbiant()

```
FliSdkError FliCredThree::getTempAmbiant (
    double & temp )
```

6.5.4.72 getTempBackEnd()

```
FliSdkError FliCredThree::getTempBackEnd (
    double & temp )
```

6.5.4.73 getTempCpu()

```
FliSdkError FliCredThree::getTempCpu (
    double & temp )
```

6.5.4.74 getTempInterface()

```
FliSdkError FliCredThree::getTempInterface (
    double & temp )
```

6.5.4.75 getTempSnake()

```
FliSdkError FliCredThree::getTempSnake (
    double & temp )
```

6.5.4.76 getTint()

```
FliSdkError FliCredThree::getTint (
    double & tint )
```

6.5.4.77 getTintGranularityState()

```
FliSdkError FliCredThree::getTintGranularityState (
    bool & enabled )
```

6.5.4.78 getTintRange()

```
FliSdkError FliCredThree::getTintRange (
    double & tintMin,
    double & tintMax )
```

6.5.4.79 getTintStep()

```
FliSdkError FliCredThree::getTintStep (
    double & step )
```

6.5.4.80 getTlsydel()

```
FliSdkError FliCredThree::getTlsydel (
    int & val )
```

6.5.4.81 getTotalUptime()

```
FliSdkError FliCredThree::getTotalUptime (
    std::string & uptime )
```

6.5.4.82 getTriggerSource()

```
FliSdkError FliCredThree::getTriggerSource (
    std::string & source )
```

6.5.4.83 getTuning()

```
FliSdkError FliCredThree::getTuning (
    std::string & tuning )
```

6.5.4.84 getUnsignedPixelsState()

```
FliSdkError FliCredThree::getUnsignedPixelsState (
    bool & enabled )
```

6.5.4.85 getUploadFirmwareConnectionInfo()

```
FliSdkError FliCredThree::getUploadFirmwareConnectionInfo (
    std::string & ip,
    uint16_t & port )
```

6.5.4.86 getUptime()

```
FliSdkError FliCredThree::getUptime (
    std::string & uptime )
```

6.5.4.87 getUserBadPixelMap()

```
FliSdkError FliCredThree::getUserBadPixelMap (
    std::vector< bool > & map,
    std::function< void(int)> getProgress = nullptr )
```

6.5.4.88 getVrefAdjustState()

```
FliSdkError FliCredThree::getVrefAdjustState (
    bool & enabled )
```

6.5.4.89 isCroppingValid()

```
FliSdkError FliCredThree::isCroppingValid (
    uint16_t col1,
    uint16_t col2,
    uint16_t row1,
    uint16_t row2 )
```

6.5.4.90 reboot()

```
FliSdkError FliCredThree::reboot ( )
```

6.5.4.91 sendBadPixelFile()

```
FliSdkError FliCredThree::sendBadPixelFile (
    std::string filePath )
```

6.5.4.92 sendBadPixelFromUrl()

```
FliSdkError FliCredThree::sendBadPixelFromUrl (
    std::string url )
```

6.5.4.93 sendBiasHdrC1File()

```
FliSdkError FliCredThree::sendBiasHdrC1File (
    std::string filePath )
```

6.5.4.94 sendBiasHdrC1FromUrl()

```
FliSdkError FliCredThree::sendBiasHdrC1FromUrl (
    std::string url )
```

6.5.4.95 sendBiasHdrC2File()

```
FliSdkError FliCredThree::sendBiasHdrC2File (
    std::string filePath )
```

6.5.4.96 sendBiasHdrC2FromUrl()

```
FliSdkError FliCredThree::sendBiasHdrC2FromUrl (
    std::string url )
```

6.5.4.97 sendFlatHdrC1File()

```
FliSdkError FliCredThree::sendFlatHdrC1File (
    std::string filePath )
```

6.5.4.98 sendFlatHdrC1FromUrl()

```
FliSdkError FliCredThree::sendFlatHdrC1FromUrl (
    std::string url )
```

6.5.4.99 sendFlatHdrC2File()

```
FliSdkError FliCredThree::sendFlatHdrC2File (
    std::string filePath )
```

6.5.4.100 sendFlatHdrC2FromUrl()

```
FliSdkError FliCredThree::sendFlatHdrC2FromUrl (
    std::string url )
```

6.5.4.101 sendLicenseFile()

```
FliSdkError FliCredThree::sendLicenseFile (
    std::string filePath,
    std::string fileName )
```

6.5.4.102 setAgcParam()

```
FliSdkError FliCredThree::setAgcParam (
    AgcParam param,
    double value )
```


6.5.4.103 setAgcPriorityNone()

```
FliSdkError FliCredThree::setAgcPriorityNone ( )
```

6.5.4.104 setAgcPriorityOverExposed()

```
FliSdkError FliCredThree::setAgcPriorityOverExposed ( )
```

6.5.4.105 setAgcPriorityUnderExposed()

```
FliSdkError FliCredThree::setAgcPriorityUnderExposed ( )
```

6.5.4.106 setAgcRoi()

```
FliSdkError FliCredThree::setAgcRoi (
    uint16_t col1,
    uint16_t row1,
    uint16_t col2,
    uint16_t row2 )
```

6.5.4.107 setConversionGainHigh()

```
FliSdkError FliCredThree::setConversionGainHigh ( )
```

6.5.4.108 setConversionGainLow()

```
FliSdkError FliCredThree::setConversionGainLow ( )
```

6.5.4.109 setConversionGainMedium()

```
FliSdkError FliCredThree::setConversionGainMedium ( )
```

6.5.4.110 setCropping()

```
FliSdkError FliCredThree::setCropping (
    bool enable,
    uint16_t col1,
    uint16_t col2,
    uint16_t row1,
    uint16_t row2 )
```

6.5.4.111 setCroppingColumns()

```
FliSdkError FliCredThree::setCroppingColumns (
    uint16_t col1,
    uint16_t col2 )
```

6.5.4.112 setCroppingRows()

```
FliSdkError FliCredThree::setCroppingRows (
    uint16_t row1,
    uint16_t row2 )
```

6.5.4.113 setDarkOptimLevel()

```
FliSdkError FliCredThree::setDarkOptimLevel (
    int level )
```

6.5.4.114 setExtSynchroExposureExternal()

```
FliSdkError FliCredThree::setExtSynchroExposureExternal ( )
```

6.5.4.115 setExtSynchroExposureInternal()

```
FliSdkError FliCredThree::setExtSynchroExposureInternal ( )
```

6.5.4.116 setExtSynchroPolarityInverted()

```
FliSdkError FliCredThree::setExtSynchroPolarityInverted ( )
```

6.5.4.117 setExtSynchroPolarityStandard()

```
FliSdkError FliCredThree::setExtSynchroPolarityStandard ( )
```

6.5.4.118 setFactoryBadPixelMap()

```
FliSdkError FliCredThree::setFactoryBadPixelMap (
    std::vector< bool > & map )
```

6.5.4.119 setFrameMarkerSourceCC1()

```
FliSdkError FliCredThree::setFrameMarkerSourceCC1 ( )
```

6.5.4.120 setFrameMarkerSourceCC2()

```
FliSdkError FliCredThree::setFrameMarkerSourceCC2 ( )
```

6.5.4.121 setFrameMarkerSourceCC3()

```
FliSdkError FliCredThree::setFrameMarkerSourceCC3 ( )
```

6.5.4.122 setFrameMarkerSourceCC4()

```
FliSdkError FliCredThree::setFrameMarkerSourceCC4 ( )
```

6.5.4.123 setFrameMarkerSourceExternal()

```
FliSdkError FliCredThree::setFrameMarkerSourceExternal ( )
```

6.5.4.124 setHdrCalibrationC1()

```
FliSdkError FliCredThree::setHdrCalibrationC1 ( )
```

6.5.4.125 setHdrCalibrationC2()

```
FliSdkError FliCredThree::setHdrCalibrationC2 ( )
```

6.5.4.126 setHdrCalibrationOff()

```
FliSdkError FliCredThree::setHdrCalibrationOff ( )
```

6.5.4.127 setImagePatternConstant()

```
FliSdkError FliCredThree::setImagePatternConstant (
    uint16_t val )
```

6.5.4.128 setImagePatternOff()

```
FliSdkError FliCredThree::setImagePatternOff ( )
```

6.5.4.129 setImagePatternRamp()

```
FliSdkError FliCredThree::setImagePatternRamp ( )
```

6.5.4.130 setNbFramesPerSwTrig()

```
FliSdkError FliCredThree::setNbFramesPerSwTrig (
    uint16_t nbFrames )
```

6.5.4.131 setPreset()

```
FliSdkError FliCredThree::setPreset ( )
```

6.5.4.132 setPresetNumber()

```
FliSdkError FliCredThree::setPresetNumber (
    uint8_t presetNumber )
```

6.5.4.133 setSnakeParam()

```
FliSdkError FliCredThree::setSnakeParam (
    std::string parameter,
    uint16_t value )
```

6.5.4.134 setSyncDelay()

```
FliSdkError FliCredThree::setSyncDelay (
    int delay )
```

6.5.4.135 setSyncSignalSourceCC1()

```
FliSdkError FliCredThree::setSyncSignalSourceCC1 ( )
```

6.5.4.136 setSyncSignalSourceCC2()

```
FliSdkError FliCredThree::setSyncSignalSourceCC2 ( )
```

6.5.4.137 setSyncSignalSourceCC3()

```
FliSdkError FliCredThree::setSyncSignalSourceCC3 ( )
```

6.5.4.138 setSyncSignalSourceCC4()

```
FliSdkError FliCredThree::setSyncSignalSourceCC4 ( )
```

6.5.4.139 setSyncSignalSourceExternal()

```
FliSdkError FliCredThree::setSyncSignalSourceExternal ( )
```

6.5.4.140 setTint()

```
FliSdkError FliCredThree::setTint (
    double tint )
```

6.5.4.141 setTlsyDel()

```
FliSdkError FliCredThree::setTlsyDel (
    int val )
```

6.5.4.142 setTriggerSourceExternal()

```
FliSdkError FliCredThree::setTriggerSourceExternal ( )
```

6.5.4.143 setTriggerSourceSoftware()

```
FliSdkError FliCredThree::setTriggerSourceSoftware ( )
```

6.5.4.144 setTuningGeneralUse()

```
FliSdkError FliCredThree::setTuningGeneralUse ( )
```

6.5.4.145 setTuningLongExposure()

```
FliSdkError FliCredThree::setTuningLongExposure ( )
```

6.5.4.146 setTuningShortExposure()

```
FliSdkError FliCredThree::setTuningShortExposure ( )
```

6.5.4.147 setUserBadPixelMap()

```
FliSdkError FliCredThree::setUserBadPixelMap (
    std::vector< bool > & map )
```

6.5.4.148 setVoltageVref()

```
FliSdkError FliCredThree::setVoltageVref (
    double vref )
```

6.5.4.149 softwareTrig()

```
FliSdkError FliCredThree::softwareTrig ( )
```

6.5.4.150 startEthernetGrabber()

```
FliSdkError FliCredThree::startEthernetGrabber ( )
```

6.5.4.151 startHttpServer()

```
FliSdkError FliCredThree::startHttpServer ( )
```

6.5.4.152 stopEthernetGrabber()

```
FliSdkError FliCredThree::stopEthernetGrabber ( )
```

6.5.4.153 stopHttpServer()

```
FliSdkError FliCredThree::stopHttpServer ( )
```

6.5.4.154 xSendBadPixelFile()

```
FliSdkError FliCredThree::xSendBadPixelFile (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.5.4.155 xSendBiasFile()

```
FliSdkError FliCredThree::xSendBiasFile (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.5.4.156 xSendBiasHdrC1File()

```
FliSdkError FliCredThree::xSendBiasHdrC1File (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.5.4.157 xSendBiasHdrC2File()

```
FliSdkError FliCredThree::xSendBiasHdrC2File (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```


6.5.4.158 xSendFlatFile()

```
FliSdkError FliCredThree::xSendFlatFile (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.5.4.159 xSendFlatHdrC1File()

```
FliSdkError FliCredThree::xSendFlatHdrC1File (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.5.4.160 xSendFlatHdrC2File()

```
FliSdkError FliCredThree::xSendFlatHdrC2File (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.5.4.161 xSendLicenseFile()

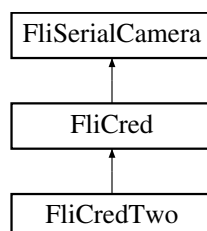
```
FliSdkError FliCredThree::xSendLicenseFile (
    std::string filePath,
    std::string fileName,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.6 FliCredTwo Class Reference

This class manages the methods specific to the C-RED 2 and C-RED 2 ER cameras.

```
#include <FliCredTwo.h>
```

Inheritance diagram for FliCredTwo:



Public Member Functions

- [FliCredTwo](#) (IFrameGrabberCL *grabber, bool isER=false)
- FliSdkError [getCropping](#) (bool &enabled, uint16_t &col1, uint16_t &col2, uint16_t &row1, uint16_t &row2)
- FliSdkError [getAllTemp](#) (double &mb, double &fe, double &pw, double &sensor, double &peltier, double &heatsink)
- FliSdkError [getNbReadWoReset](#) (int &nbread)
- FliSdkError [getRawImagesState](#) (bool &enabled)
- FliSdkError [getAntiBloomingState](#) (bool &enabled)
- FliSdkError [getTempFrontEnd](#) (double &temp)
- FliSdkError [getTempMotherBoard](#) (double &temp)
- FliSdkError [getTempPowerBoard](#) (double &temp)
- FliSdkError [getTempHeatSink](#) (double &temp)
- FliSdkError [getTempPeltier](#) (double &temp)
- FliSdkError [getTempSnake](#) (double &temp)
- FliSdkError [getTint](#) (double &tint)
- FliSdkError [getTintRange](#) (double &tintMin, double &tintMax)
- FliSdkError [getBadPixelState](#) (bool &enabled)
- FliSdkError [getConversionGain](#) (std::string &conversionGain)
- FliSdkError [getDarkOptimLevel](#) (int &level)
- FliSdkError [getTempSnakeSetpoint](#) (double &temp)
- FliSdkError [getFanMode](#) (std::string &mode)
- FliSdkError [getExtSynchroExposure](#) (std::string &exposure)
- FliSdkError [getExtSynchroPolarity](#) (std::string &polarity)
- FliSdkError [getTuning](#) (std::string &tuning)
- FliSdkError [getFanSpeed](#) (int &speed)
- FliSdkError [getHdrState](#) (bool &enabled)
- FliSdkError [getHdrCalibrationMode](#) (std::string &mode)
- FliSdkError [getHdrExtendedState](#) (bool &enabled)
- FliSdkError [getLicenses](#) (std::vector< std::string > &licenses)
- FliSdkError [getMaxFpsUsb](#) (double &maxFpsUsb)
- FliSdkError [getMaxSyncDelay](#) (double &maxSyncDelay)
- FliSdkError [getMinSyncDelay](#) (double &minSyncDelay)
- FliSdkError [getMaxTintItr](#) (double &maxTintItr)
- FliSdkError [getVoltageVref](#) (double &vref)
- FliSdkError [getMinFps](#) (double &minFps)
- FliSdkError [getNbFramesPerSwTrig](#) (int &nbFrames)
- FliSdkError [getTlsydel](#) (int &val)
- FliSdkError [getPreset](#) (int &preset)
- FliSdkError [getRemoteMaintenanceState](#) (bool &enabled)
- FliSdkError [getSwSynchroState](#) (bool &enabled)
- FliSdkError [getTcdsAdjustState](#) (bool &enabled)
- FliSdkError [getTelnetState](#) (bool &enabled)
- FliSdkError [getTintGranularityState](#) (bool &enabled)
- FliSdkError [getVrefAdjustState](#) (bool &enabled)
- FliSdkError [getStepSyncDelay](#) (double &delay)
- FliSdkError [getSyncDelay](#) (double &delay)
- FliSdkError [getSynchronization](#) (std::string &synchro)
- FliSdkError [getIpAlternateDns](#) (std::string &dns)
- FliSdkError [getIpDns](#) (std::string &dns)
- FliSdkError [getIpGateway](#) (std::string &gateway)
- FliSdkError [getIpMode](#) (std::string &mode)
- FliSdkError [getIpNetmask](#) (std::string &netmask)
- FliSdkError [getIpAddress](#) (std::string &ip)
- FliSdkError [getSnakeParam](#) (std::string parameter, uint16_t &value)

- FliSdkError [getPowers](#) (double &extPeltierCurrent, double &extPeltierVoltage, double &extPeltierPower, double &intPeltierCurrent, double &intPeltierVoltage, double &intPeltierPower)
- FliSdkError [getPowerExternalPeltier](#) (double ¤t, double &voltage, double &power)
- FliSdkError [getPowerSensor](#) (double ¤t, double &voltage, double &power)
- FliSdkError [getUploadFirmwareConnectionInfo](#) (std::string &ip, uint16_t &port)
- FliSdkError [getTriggerSource](#) (std::string &source)
- FliSdkError [getSyncSignalSource](#) (std::string &source)
- FliSdkError [getExtMarkerSource](#) (std::string &source)
- FliSdkError [getHardwareFeatures](#) (int &features)
- FliSdkError [getSoftwareFeatures](#) (int &features)
- FliSdkError [getFactoryBadPixelMap](#) (std::vector< bool > &map, std::function< void(int)> get←Progress=nullptr)
- FliSdkError [getUserBadPixelMap](#) (std::vector< bool > &map, std::function< void(int)> getProgress=nullptr)
- FliSdkError [getAgcState](#) (bool &enabled)
- FliSdkError [getAgcPriority](#) (std::string &priority)
- FliSdkError [getAgcRoi](#) (uint16_t &col1, uint16_t &row1, uint16_t &col2, uint16_t &row2)
- FliSdkError [getFactoryCorrectionState](#) (bool &enabled)
- FliSdkError [getTintStep](#) (double &step)
- FliSdkError [getImagePattern](#) (std::string &pattern)
- FliSdkError [getDate](#) (std::string &date)
- FliSdkError [getUptime](#) (std::string &uptime)
- FliSdkError [getAccumulatedUptime](#) (std::string &uptime)
- FliSdkError [getTotalUptime](#) (std::string &uptime)
- FliSdkError [getUnsignedPixelsState](#) (bool &enabled)
- FliSdkError [setCropping](#) (bool enable, uint16_t col1, uint16_t col2, uint16_t row1, uint16_t row2)
- FliSdkError [setCroppingColumns](#) (uint16_t col1, uint16_t col2)
- FliSdkError [setCroppingRows](#) (uint16_t row1, uint16_t row2)
- FliSdkError [setNbReadWoReset](#) (int nbRead)
- FliSdkError [setNbFramesPerSwTrig](#) (int nbFrames)
- FliSdkError [setDarkOptimLevel](#) (int level)
- FliSdkError [setSensorTemp](#) (double temp)
- FliSdkError [setTint](#) (double tint)
- FliSdkError [setConversionGainHigh](#) ()
- FliSdkError [setConversionGainMedium](#) ()
- FliSdkError [setConversionGainLow](#) ()
- FliSdkError [setExtSynchroExposureInternal](#) ()
- FliSdkError [setExtSynchroExposureExternal](#) ()
- FliSdkError [setExtSynchroPolarityInverted](#) ()
- FliSdkError [setExtSynchroPolarityStandard](#) ()
- FliSdkError [setTuningGeneralUse](#) ()
- FliSdkError [setTuningShortExposure](#) ()
- FliSdkError [setTuningLongExposure](#) ()
- FliSdkError [setFanSpeed](#) (int speed)
- FliSdkError [setSyncDelay](#) (int delay)
- FliSdkError [setTlSyDel](#) (int val)
- FliSdkError [setVoltageVref](#) (double vref)
- FliSdkError [setFanModeAutomatic](#) ()
- FliSdkError [setFanModeManual](#) ()
- FliSdkError [setHdrCalibrationC1](#) ()
- FliSdkError [setHdrCalibrationC2](#) ()
- FliSdkError [setHdrCalibrationOff](#) ()
- FliSdkError [setSynchronizationCmos](#) ()
- FliSdkError [setSynchronizationFullCmos](#) ()
- FliSdkError [setSynchronizationLvds](#) ()
- FliSdkError [setPreset](#) ()

- FliSdkError [setPresetNumber](#) (uint8_t presetNumber)
- FliSdkError [setSnakeParam](#) (std::string parameter, uint16_t value)
- FliSdkError [setTriggerSourceSoftware](#) ()
- FliSdkError [setTriggerSourceExternal](#) ()
- FliSdkError [setSyncSignalSourceExternal](#) ()
- FliSdkError [setSyncSignalSourceCC1](#) ()
- FliSdkError [setSyncSignalSourceCC2](#) ()
- FliSdkError [setSyncSignalSourceCC3](#) ()
- FliSdkError [setSyncSignalSourceCC4](#) ()
- FliSdkError [setFrameMarkerSourceExternal](#) ()
- FliSdkError [setFrameMarkerSourceCC1](#) ()
- FliSdkError [setFrameMarkerSourceCC2](#) ()
- FliSdkError [setFrameMarkerSourceCC3](#) ()
- FliSdkError [setFrameMarkerSourceCC4](#) ()
- FliSdkError [setFactoryBadPixelMap](#) (std::vector< bool > &map)
- FliSdkError [setUserBadPixelMap](#) (std::vector< bool > &map)
- FliSdkError [setAgcPriorityNone](#) ()
- FliSdkError [setAgcPriorityOverExposed](#) ()
- FliSdkError [setAgcPriorityUnderExposed](#) ()
- FliSdkError [setAgcRoi](#) (uint16_t col1, uint16_t row1, uint16_t col2, uint16_t row2)
- FliSdkError [setImagePatternRamp](#) ()
- FliSdkError [setImagePatternConstant](#) (uint16_t val)
- FliSdkError [setImagePatternOff](#) ()
- FliSdkError [enableRawImages](#) (bool enable)
- FliSdkError [enableBadPixel](#) (bool enable)
- FliSdkError [enableHdr](#) (bool enable)
- FliSdkError [enableAntiBlooming](#) (bool enable)
- FliSdkError [enableHdrExtended](#) (bool enable)
- FliSdkError [enableRemoteMaintenance](#) (bool enable)
- FliSdkError [enableSwSynchro](#) (bool enable)
- FliSdkError [enableTcdsAdjust](#) (bool enable)
- FliSdkError [enableTintGranularity](#) (bool enable)
- FliSdkError [enableVrefAdjust](#) (bool enable)
- FliSdkError [enableAgc](#) (bool enable)
- FliSdkError [enableFactoryCorrection](#) (bool enable)
- FliSdkError [enableUnsignedPixels](#) (bool enable)
- FliSdkError [reboot](#) ()
- FliSdkError [buildFlatHdrC1](#) ()
- FliSdkError [buildFlatHdrC2](#) ()
- FliSdkError [sendBiasHdrC1FromUrl](#) (std::string url)
- FliSdkError [sendBiasHdrC2FromUrl](#) (std::string url)
- FliSdkError [sendFlatHdrC1FromUrl](#) (std::string url)
- FliSdkError [sendFlatHdrC2FromUrl](#) (std::string url)
- FliSdkError [sendBiasHdrC1File](#) (std::string filePath)
- FliSdkError [sendBiasHdrC2File](#) (std::string filePath)
- FliSdkError [sendFlatHdrC1File](#) (std::string filePath)
- FliSdkError [sendFlatHdrC2File](#) (std::string filePath)
- FliSdkError [xSendBiasFile](#) (std::string filePath, std::function< void(bool, int, int)> getBlockStatus=nullptr)
- FliSdkError [xSendBiasHdrC1File](#) (std::string filePath, std::function< void(bool, int, int)> getBlock↔ Status=nullptr)
- FliSdkError [xSendBiasHdrC2File](#) (std::string filePath, std::function< void(bool, int, int)> getBlock↔ Status=nullptr)
- FliSdkError [xSendFlatFile](#) (std::string filePath, std::function< void(bool, int, int)> getBlockStatus=nullptr)
- FliSdkError [xSendFlatHdrC1File](#) (std::string filePath, std::function< void(bool, int, int)> getBlock↔ Status=nullptr)

- FliSdkError [xSendFlatHdrC2File](#) (std::string filePath, std::function< void(bool, int, int)> getBlockStatus=nullptr)
- FliSdkError [sendLicenseFile](#) (std::string filePath, std::string fileName)
- FliSdkError [xSendLicenseFile](#) (std::string filePath, std::string fileName, std::function< void(bool, int, int)> getBlockStatus=nullptr)
- FliSdkError [deleteLicense](#) (std::string licenseName)
- FliSdkError [disableLicense](#) (std::string licenseName)
- FliSdkError [enableLicense](#) (std::string licenseName)
- FliSdkError [softwareTrig](#) ()
- FliSdkError [sendBadPixelFile](#) (std::string filePath)
- FliSdkError [xSendBadPixelFile](#) (std::string filePath, std::function< void(bool, int, int)> getBlockStatus=nullptr)
- FliSdkError [sendBadPixelFromUrl](#) (std::string url)
- FliSdkError [isCroppingValid](#) (uint16_t col1, uint16_t col2, uint16_t row1, uint16_t row2)
- FliSdkError [buildBiasNuc](#) (uint16_t nblImages=256)
- FliSdkError [buildFlatNuc](#) (uint16_t nblImages=256)
- FliSdkError [buildFlatHdrC1Nuc](#) (uint16_t nblImages=256)
- FliSdkError [buildFlatHdrC2Nuc](#) (uint16_t nblImages=256)
- FliSdkError [abortBuildNuc](#) ()
- FliSdkError [getBuildNucProgress](#) (int &progress)
- FliSdkError [startHttpServer](#) ()
- FliSdkError [stopHttpServer](#) ()
- FliSdkError [startEthernetGrabber](#) ()
- FliSdkError [stopEthernetGrabber](#) ()

Additional Inherited Members

6.6.1 Detailed Description

This class manages the methods specific to the C-RED 2 and C-RED 2 ER cameras.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 FliCredTwo()

```
FliCredTwo::FliCredTwo (
    IFrameGrabberCL * grabber,
    bool isER = false )
```

6.6.3 Member Function Documentation

6.6.3.1 abortBuildNuc()

```
FliSdkError FliCredTwo::abortBuildNuc ( )
```

6.6.3.2 buildBiasNuc()

```
FliSdkError FliCredTwo::buildBiasNuc (
    uint16_t nbImages = 256 )
```

6.6.3.3 buildFlatHdrC1()

```
FliSdkError FliCredTwo::buildFlatHdrC1 ( )
```

6.6.3.4 buildFlatHdrC1Nuc()

```
FliSdkError FliCredTwo::buildFlatHdrC1Nuc (
    uint16_t nbImages = 256 )
```

6.6.3.5 buildFlatHdrC2()

```
FliSdkError FliCredTwo::buildFlatHdrC2 ( )
```

6.6.3.6 buildFlatHdrC2Nuc()

```
FliSdkError FliCredTwo::buildFlatHdrC2Nuc (
    uint16_t nbImages = 256 )
```

6.6.3.7 buildFlatNuc()

```
FliSdkError FliCredTwo::buildFlatNuc (
    uint16_t nbImages = 256 )
```

6.6.3.8 deleteLicense()

```
FliSdkError FliCredTwo::deleteLicense (
    std::string licenseName )
```

6.6.3.9 disableLicense()

```
FliSdkError FliCredTwo::disableLicense (
    std::string licenseName )
```

6.6.3.10 enableAgc()

```
FliSdkError FliCredTwo::enableAgc (
    bool enable )
```

6.6.3.11 enableAntiBlooming()

```
FliSdkError FliCredTwo::enableAntiBlooming (
    bool enable )
```

6.6.3.12 enableBadPixel()

```
FliSdkError FliCredTwo::enableBadPixel (
    bool enable )
```

6.6.3.13 enableFactoryCorrection()

```
FliSdkError FliCredTwo::enableFactoryCorrection (
    bool enable )
```

6.6.3.14 enableHdr()

```
FliSdkError FliCredTwo::enableHdr (
    bool enable )
```

6.6.3.15 enableHdrExtended()

```
FliSdkError FliCredTwo::enableHdrExtended (
    bool enable )
```

6.6.3.16 enableLicense()

```
FliSdkError FliCredTwo::enableLicense (
    std::string licenseName )
```

6.6.3.17 enableRawImages()

```
FliSdkError FliCredTwo::enableRawImages (
    bool enable )
```

6.6.3.18 enableRemoteMaintenance()

```
FliSdkError FliCredTwo::enableRemoteMaintenance (
    bool enable )
```

6.6.3.19 enableSwSynchro()

```
FliSdkError FliCredTwo::enableSwSynchro (
    bool enable )
```

6.6.3.20 enableTcdsAdjust()

```
FliSdkError FliCredTwo::enableTcdsAdjust (
    bool enable )
```

6.6.3.21 enableTintGranularity()

```
FliSdkError FliCredTwo::enableTintGranularity (
    bool enable )
```

6.6.3.22 enableUnsignedPixels()

```
FliSdkError FliCredTwo::enableUnsignedPixels (
    bool enable )
```


6.6.3.23 enableVrefAdjust()

```
FliSdkError FliCredTwo::enableVrefAdjust (
    bool enable )
```

6.6.3.24 getAccumulatedUptime()

```
FliSdkError FliCredTwo::getAccumulatedUptime (
    std::string & uptime )
```

6.6.3.25 getAgcPriority()

```
FliSdkError FliCredTwo::getAgcPriority (
    std::string & priority )
```

6.6.3.26 getAgcRoi()

```
FliSdkError FliCredTwo::getAgcRoi (
    uint16_t & col1,
    uint16_t & row1,
    uint16_t & col2,
    uint16_t & row2 )
```

6.6.3.27 getAgcState()

```
FliSdkError FliCredTwo::getAgcState (
    bool & enabled )
```

6.6.3.28 getAllTemp()

```
FliSdkError FliCredTwo::getAllTemp (
    double & mb,
    double & fe,
    double & pw,
    double & sensor,
    double & peltier,
    double & heatsink )
```

6.6.3.29 getAntiBloomingState()

```
FliSdkError FliCredTwo::getAntiBloomingState (
    bool & enabled )
```

6.6.3.30 getBadPixelState()

```
FliSdkError FliCredTwo::getBadPixelState (
    bool & enabled )
```

6.6.3.31 getBuildNucProgress()

```
FliSdkError FliCredTwo::getBuildNucProgress (
    int & progress )
```

6.6.3.32 getConversionGain()

```
FliSdkError FliCredTwo::getConversionGain (
    std::string & conversionGain )
```

6.6.3.33 getCropping()

```
FliSdkError FliCredTwo::getCropping (
    bool & enabled,
    uint16_t & col1,
    uint16_t & col2,
    uint16_t & row1,
    uint16_t & row2 )
```

6.6.3.34 getDarkOptimLevel()

```
FliSdkError FliCredTwo::getDarkOptimLevel (
    int & level )
```

6.6.3.35 getDate()

```
FliSdkError FliCredTwo::getDate (
    std::string & date )
```

6.6.3.36 getExtMarkerSource()

```
FliSdkError FliCredTwo::getExtMarkerSource (
    std::string & source )
```

6.6.3.37 getExtSynchroExposure()

```
FliSdkError FliCredTwo::getExtSynchroExposure (
    std::string & exposure )
```

6.6.3.38 getExtSynchroPolarity()

```
FliSdkError FliCredTwo::getExtSynchroPolarity (
    std::string & polarity )
```

6.6.3.39 getFactoryBadPixelMap()

```
FliSdkError FliCredTwo::getFactoryBadPixelMap (
    std::vector< bool > & map,
    std::function< void(int)> getProgress = nullptr )
```

6.6.3.40 getFactoryCorrectionState()

```
FliSdkError FliCredTwo::getFactoryCorrectionState (
    bool & enabled )
```

6.6.3.41 getFanMode()

```
FliSdkError FliCredTwo::getFanMode (
    std::string & mode )
```

6.6.3.42 getFanSpeed()

```
FliSdkError FliCredTwo::getFanSpeed (
    int & speed )
```

6.6.3.43 getHardwareFeatures()

```
FliSdkError FliCredTwo::getHardwareFeatures (
    int & features )
```

6.6.3.44 getHdrCalibrationMode()

```
FliSdkError FliCredTwo::getHdrCalibrationMode (
    std::string & mode )
```

6.6.3.45 getHdrExtendedState()

```
FliSdkError FliCredTwo::getHdrExtendedState (
    bool & enabled )
```

6.6.3.46 getHdrState()

```
FliSdkError FliCredTwo::getHdrState (
    bool & enabled )
```

6.6.3.47 getImagePattern()

```
FliSdkError FliCredTwo::getImagePattern (
    std::string & pattern )
```

6.6.3.48 getIpAddress()

```
FliSdkError FliCredTwo::getIpAddress (
    std::string & ip )
```

6.6.3.49 getIpAlternateDns()

```
FliSdkError FliCredTwo::getIpAlternateDns (
    std::string & dns )
```

6.6.3.50 getIpDns()

```
FliSdkError FliCredTwo::getIpDns (
    std::string & dns )
```

6.6.3.51 getIpGateway()

```
FliSdkError FliCredTwo::getIpGateway (
    std::string & gateway )
```

6.6.3.52 getIpMode()

```
FliSdkError FliCredTwo::getIpMode (
    std::string & mode )
```

6.6.3.53 getIpNetmask()

```
FliSdkError FliCredTwo::getIpNetmask (
    std::string & netmask )
```

6.6.3.54 getLicenses()

```
FliSdkError FliCredTwo::getLicenses (
    std::vector< std::string > & licenses )
```

6.6.3.55 getMaxFpsUsb()

```
FliSdkError FliCredTwo::getMaxFpsUsb (
    double & maxFpsUsb )
```

6.6.3.56 getMaxSyncDelay()

```
FliSdkError FliCredTwo::getMaxSyncDelay (
    double & maxSyncDelay )
```

6.6.3.57 getMaxTintItr()

```
FliSdkError FliCredTwo::getMaxTintItr (
    double & maxTintItr )
```

6.6.3.58 getMinFps()

```
FliSdkError FliCredTwo::getMinFps (
    double & minFps )
```

6.6.3.59 getMinSyncDelay()

```
FliSdkError FliCredTwo::getMinSyncDelay (
    double & minSyncDelay )
```

6.6.3.60 getNbFramesPerSwTrig()

```
FliSdkError FliCredTwo::getNbFramesPerSwTrig (
    int & nbFrames )
```

6.6.3.61 getNbReadWoReset()

```
FliSdkError FliCredTwo::getNbReadWoReset (
    int & nbread )
```

6.6.3.62 getPowerExternalPeltier()

```
FliSdkError FliCredTwo::getPowerExternalPeltier (
    double & current,
    double & voltage,
    double & power )
```

6.6.3.63 getPowers()

```
FliSdkError FliCredTwo::getPowers (
    double & extPeltierCurrent,
    double & extPeltierVoltage,
    double & extPeltierPower,
    double & intPeltierCurrent,
    double & intPeltierVoltage,
    double & intPeltierPower )
```

6.6.3.64 getPowerSensor()

```
FliSdkError FliCredTwo::getPowerSensor (
    double & current,
    double & voltage,
    double & power )
```

6.6.3.65 getPreset()

```
FliSdkError FliCredTwo::getPreset (
    int & preset )
```

6.6.3.66 getRawImagesState()

```
FliSdkError FliCredTwo::getRawImagesState (
    bool & enabled )
```

6.6.3.67 getRemoteMaintenanceState()

```
FliSdkError FliCredTwo::getRemoteMaintenanceState (
    bool & enabled )
```

6.6.3.68 getSnakeParam()

```
FliSdkError FliCredTwo::getSnakeParam (
    std::string parameter,
    uint16_t & value )
```

6.6.3.69 getSoftwareFeatures()

```
FliSdkError FliCredTwo::getSoftwareFeatures (
    int & features )
```

6.6.3.70 getStepSyncDelay()

```
FliSdkError FliCredTwo::getStepSyncDelay (
    double & delay )
```

6.6.3.71 getSwSynchroState()

```
FliSdkError FliCredTwo::getSwSynchroState (
    bool & enabled )
```

6.6.3.72 getSyncDelay()

```
FliSdkError FliCredTwo::getSyncDelay (
    double & delay )
```

6.6.3.73 getSynchronization()

```
FliSdkError FliCredTwo::getSynchronization (
    std::string & synchro )
```

6.6.3.74 getSyncSignalSource()

```
FliSdkError FliCredTwo::getSyncSignalSource (
    std::string & source )
```

6.6.3.75 getTcdsAdjustState()

```
FliSdkError FliCredTwo::getTcdsAdjustState (
    bool & enabled )
```


6.6.3.76 getTelnetState()

```
FliSdkError FliCredTwo::getTelnetState (
    bool & enabled )
```

6.6.3.77 getTempFrontEnd()

```
FliSdkError FliCredTwo::getTempFrontEnd (
    double & temp )
```

6.6.3.78 getTempHeatSink()

```
FliSdkError FliCredTwo::getTempHeatSink (
    double & temp )
```

6.6.3.79 getTempMotherBoard()

```
FliSdkError FliCredTwo::getTempMotherBoard (
    double & temp )
```

6.6.3.80 getTempPeltier()

```
FliSdkError FliCredTwo::getTempPeltier (
    double & temp )
```

6.6.3.81 getTempPowerBoard()

```
FliSdkError FliCredTwo::getTempPowerBoard (
    double & temp )
```

6.6.3.82 getTempSnake()

```
FliSdkError FliCredTwo::getTempSnake (
    double & temp )
```

6.6.3.83 getTempSnakeSetpoint()

```
FliSdkError FliCredTwo::getTempSnakeSetpoint (
    double & temp )
```

6.6.3.84 getTint()

```
FliSdkError FliCredTwo::getTint (
    double & tint )
```

6.6.3.85 getTintGranularityState()

```
FliSdkError FliCredTwo::getTintGranularityState (
    bool & enabled )
```

6.6.3.86 getTintRange()

```
FliSdkError FliCredTwo::getTintRange (
    double & tintMin,
    double & tintMax )
```

6.6.3.87 getTintStep()

```
FliSdkError FliCredTwo::getTintStep (
    double & step )
```

6.6.3.88 getTlsydel()

```
FliSdkError FliCredTwo::getTlsydel (
    int & val )
```

6.6.3.89 getTotalUptime()

```
FliSdkError FliCredTwo::getTotalUptime (
    std::string & uptime )
```

6.6.3.90 getTriggerSource()

```
FliSdkError FliCredTwo::getTriggerSource (
    std::string & source )
```

6.6.3.91 getTuning()

```
FliSdkError FliCredTwo::getTuning (
    std::string & tuning )
```

6.6.3.92 getUnsignedPixelsState()

```
FliSdkError FliCredTwo::getUnsignedPixelsState (
    bool & enabled )
```

6.6.3.93 getUploadFirmwareConnectionInfo()

```
FliSdkError FliCredTwo::getUploadFirmwareConnectionInfo (
    std::string & ip,
    uint16_t & port )
```

6.6.3.94 getUptime()

```
FliSdkError FliCredTwo::getUptime (
    std::string & uptime )
```

6.6.3.95 getUserBadPixelMap()

```
FliSdkError FliCredTwo::getUserBadPixelMap (
    std::vector< bool > & map,
    std::function< void(int)> getProgress = nullptr )
```

6.6.3.96 getVoltageVref()

```
FliSdkError FliCredTwo::getVoltageVref (
    double & vref )
```

6.6.3.97 getVrefAdjustState()

```
FliSdkError FliCredTwo::getVrefAdjustState (
    bool & enabled )
```

6.6.3.98 isCroppingValid()

```
FliSdkError FliCredTwo::isCroppingValid (
    uint16_t col1,
    uint16_t col2,
    uint16_t row1,
    uint16_t row2 )
```

6.6.3.99 reboot()

```
FliSdkError FliCredTwo::reboot ( )
```

6.6.3.100 sendBadPixelFile()

```
FliSdkError FliCredTwo::sendBadPixelFile (
    std::string filePath )
```

6.6.3.101 sendBadPixelFromUrl()

```
FliSdkError FliCredTwo::sendBadPixelFromUrl (
    std::string url )
```

6.6.3.102 sendBiasHdrC1File()

```
FliSdkError FliCredTwo::sendBiasHdrC1File (
    std::string filePath )
```

6.6.3.103 sendBiasHdrC1FromUrl()

```
FliSdkError FliCredTwo::sendBiasHdrC1FromUrl (
    std::string url )
```

6.6.3.104 sendBiasHdrC2File()

```
FliSdkError FliCredTwo::sendBiasHdrC2File (
    std::string filePath )
```

6.6.3.105 sendBiasHdrC2FromUrl()

```
FliSdkError FliCredTwo::sendBiasHdrC2FromUrl (
    std::string url )
```

6.6.3.106 sendFlatHdrC1File()

```
FliSdkError FliCredTwo::sendFlatHdrC1File (
    std::string filePath )
```

6.6.3.107 sendFlatHdrC1FromUrl()

```
FliSdkError FliCredTwo::sendFlatHdrC1FromUrl (
    std::string url )
```

6.6.3.108 sendFlatHdrC2File()

```
FliSdkError FliCredTwo::sendFlatHdrC2File (
    std::string filePath )
```

6.6.3.109 sendFlatHdrC2FromUrl()

```
FliSdkError FliCredTwo::sendFlatHdrC2FromUrl (
    std::string url )
```

6.6.3.110 sendLicenseFile()

```
FliSdkError FliCredTwo::sendLicenseFile (
    std::string filePath,
    std::string fileName )
```

6.6.3.111 setAgcPriorityNone()

```
FliSdkError FliCredTwo::setAgcPriorityNone ( )
```

6.6.3.112 setAgcPriorityOverExposed()

```
FliSdkError FliCredTwo::setAgcPriorityOverExposed ( )
```

6.6.3.113 setAgcPriorityUnderExposed()

```
FliSdkError FliCredTwo::setAgcPriorityUnderExposed ( )
```

6.6.3.114 setAgcRoi()

```
FliSdkError FliCredTwo::setAgcRoi (
    uint16_t col1,
    uint16_t row1,
    uint16_t col2,
    uint16_t row2 )
```

6.6.3.115 setConversionGainHigh()

```
FliSdkError FliCredTwo::setConversionGainHigh ( )
```

6.6.3.116 setConversionGainLow()

```
FliSdkError FliCredTwo::setConversionGainLow ( )
```

6.6.3.117 setConversionGainMedium()

```
FliSdkError FliCredTwo::setConversionGainMedium ( )
```

6.6.3.118 setCropping()

```
FliSdkError FliCredTwo::setCropping (
    bool enable,
    uint16_t col1,
    uint16_t col2,
    uint16_t row1,
    uint16_t row2 )
```

6.6.3.119 setCroppingColumns()

```
FliSdkError FliCredTwo::setCroppingColumns (
    uint16_t col1,
    uint16_t col2 )
```

6.6.3.120 setCroppingRows()

```
FliSdkError FliCredTwo::setCroppingRows (
    uint16_t row1,
    uint16_t row2 )
```

6.6.3.121 setDarkOptimLevel()

```
FliSdkError FliCredTwo::setDarkOptimLevel (
    int level )
```

6.6.3.122 setExtSynchroExposureExternal()

```
FliSdkError FliCredTwo::setExtSynchroExposureExternal ( )
```

6.6.3.123 setExtSynchroExposureInternal()

```
FliSdkError FliCredTwo::setExtSynchroExposureInternal ( )
```

6.6.3.124 setExtSynchroPolarityInverted()

```
FliSdkError FliCredTwo::setExtSynchroPolarityInverted ( )
```

6.6.3.125 setExtSynchroPolarityStandard()

```
FliSdkError FliCredTwo::setExtSynchroPolarityStandard ( )
```

6.6.3.126 setFactoryBadPixelMap()

```
FliSdkError FliCredTwo::setFactoryBadPixelMap (
    std::vector< bool > & map )
```

6.6.3.127 setFanModeAutomatic()

```
FliSdkError FliCredTwo::setFanModeAutomatic ( )
```

6.6.3.128 setFanModeManual()

```
FliSdkError FliCredTwo::setFanModeManual ( )
```

6.6.3.129 setFanSpeed()

```
FliSdkError FliCredTwo::setFanSpeed (
    int speed )
```

6.6.3.130 setFrameMarkerSourceCC1()

```
FliSdkError FliCredTwo::setFrameMarkerSourceCC1 ( )
```


6.6.3.131 setFrameMarkerSourceCC2()

```
FliSdkError FliCredTwo::setFrameMarkerSourceCC2 ( )
```

6.6.3.132 setFrameMarkerSourceCC3()

```
FliSdkError FliCredTwo::setFrameMarkerSourceCC3 ( )
```

6.6.3.133 setFrameMarkerSourceCC4()

```
FliSdkError FliCredTwo::setFrameMarkerSourceCC4 ( )
```

6.6.3.134 setFrameMarkerSourceExternal()

```
FliSdkError FliCredTwo::setFrameMarkerSourceExternal ( )
```

6.6.3.135 setHdrCalibrationC1()

```
FliSdkError FliCredTwo::setHdrCalibrationC1 ( )
```

6.6.3.136 setHdrCalibrationC2()

```
FliSdkError FliCredTwo::setHdrCalibrationC2 ( )
```

6.6.3.137 setHdrCalibrationOff()

```
FliSdkError FliCredTwo::setHdrCalibrationOff ( )
```

6.6.3.138 setImagePatternConstant()

```
FliSdkError FliCredTwo::setImagePatternConstant (
    uint16_t val )
```

6.6.3.139 setImagePatternOff()

```
FliSdkError FliCredTwo::setImagePatternOff ( )
```

6.6.3.140 setImagePatternRamp()

```
FliSdkError FliCredTwo::setImagePatternRamp ( )
```

6.6.3.141 setNbFramesPerSwTrig()

```
FliSdkError FliCredTwo::setNbFramesPerSwTrig (
    int nbFrames )
```

6.6.3.142 setNbReadWoReset()

```
FliSdkError FliCredTwo::setNbReadWoReset (
    int nbRead )
```

6.6.3.143 setPreset()

```
FliSdkError FliCredTwo::setPreset ( )
```

6.6.3.144 setPresetNumber()

```
FliSdkError FliCredTwo::setPresetNumber (
    uint8_t presetNumber )
```

6.6.3.145 setSensorTemp()

```
FliSdkError FliCredTwo::setSensorTemp (
    double temp )
```

6.6.3.146 setSnakeParam()

```
FliSdkError FliCredTwo::setSnakeParam (
    std::string parameter,
    uint16_t value )
```

6.6.3.147 setSyncDelay()

```
FliSdkError FliCredTwo::setSyncDelay (
    int delay )
```

6.6.3.148 setSynchronizationCmos()

```
FliSdkError FliCredTwo::setSynchronizationCmos ( )
```

6.6.3.149 setSynchronizationFullCmos()

```
FliSdkError FliCredTwo::setSynchronizationFullCmos ( )
```

6.6.3.150 setSynchronizationLvds()

```
FliSdkError FliCredTwo::setSynchronizationLvds ( )
```

6.6.3.151 setSyncSignalSourceCC1()

```
FliSdkError FliCredTwo::setSyncSignalSourceCC1 ( )
```

6.6.3.152 setSyncSignalSourceCC2()

```
FliSdkError FliCredTwo::setSyncSignalSourceCC2 ( )
```

6.6.3.153 setSyncSignalSourceCC3()

```
FliSdkError FliCredTwo::setSyncSignalSourceCC3 ( )
```

6.6.3.154 setSyncSignalSourceCC4()

```
FliSdkError FliCredTwo::setSyncSignalSourceCC4 ( )
```

6.6.3.155 setSyncSignalSourceExternal()

```
FliSdkError FliCredTwo::setSyncSignalSourceExternal ( )
```

6.6.3.156 setTint()

```
FliSdkError FliCredTwo::setTint (
    double tint )
```

6.6.3.157 setTlsyDel()

```
FliSdkError FliCredTwo::setTlsyDel (
    int val )
```

6.6.3.158 setTriggerSourceExternal()

```
FliSdkError FliCredTwo::setTriggerSourceExternal ( )
```

6.6.3.159 setTriggerSourceSoftware()

```
FliSdkError FliCredTwo::setTriggerSourceSoftware ( )
```

6.6.3.160 setTuningGeneralUse()

```
FliSdkError FliCredTwo::setTuningGeneralUse ( )
```

6.6.3.161 setTuningLongExposure()

```
FliSdkError FliCredTwo::setTuningLongExposure ( )
```

6.6.3.162 setTuningShortExposure()

```
FliSdkError FliCredTwo::setTuningShortExposure ( )
```

6.6.3.163 setUserBadPixelMap()

```
FliSdkError FliCredTwo::setUserBadPixelMap (
    std::vector< bool > & map )
```

6.6.3.164 setVoltageVref()

```
FliSdkError FliCredTwo::setVoltageVref (
    double vref )
```

6.6.3.165 softwareTrig()

```
FliSdkError FliCredTwo::softwareTrig ( )
```

6.6.3.166 startEthernetGrabber()

```
FliSdkError FliCredTwo::startEthernetGrabber ( )
```

6.6.3.167 startHttpServer()

```
FliSdkError FliCredTwo::startHttpServer ( )
```

6.6.3.168 stopEthernetGrabber()

```
FliSdkError FliCredTwo::stopEthernetGrabber ( )
```

6.6.3.169 stopHttpServer()

```
FliSdkError FliCredTwo::stopHttpServer ( )
```

6.6.3.170 xSendBadPixelFile()

```
FliSdkError FliCredTwo::xSendBadPixelFile (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.6.3.171 xSendBiasFile()

```
FliSdkError FliCredTwo::xSendBiasFile (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.6.3.172 xSendBiasHdrC1File()

```
FliSdkError FliCredTwo::xSendBiasHdrC1File (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.6.3.173 xSendBiasHdrC2File()

```
FliSdkError FliCredTwo::xSendBiasHdrC2File (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.6.3.174 xSendFlatFile()

```
FliSdkError FliCredTwo::xSendFlatFile (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.6.3.175 xSendFlatHdrC1File()

```
FliSdkError FliCredTwo::xSendFlatHdrC1File (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.6.3.176 xSendFlatHdrC2File()

```
FliSdkError FliCredTwo::xSendFlatHdrC2File (
    std::string filePath,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.6.3.177 xSendLicenseFile()

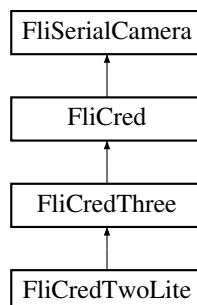
```
FliSdkError FliCredTwo::xSendLicenseFile (
    std::string filePath,
    std::string fileName,
    std::function< void(bool, int, int)> getBlockStatus = nullptr )
```

6.7 FliCredTwoLite Class Reference

This class manages the methods specific to the C-RED 2 Lite camera.

```
#include <FliCredTwoLite.h>
```

Inheritance diagram for FliCredTwoLite:



Public Types

- enum `CoolingMode` { `MANUAL`, `AUTOMATIC`, `CUSTOM_STEPS` }

Public Member Functions

- `FliCredTwoLite` (`IFrameGrabberCL *grabber`)
- `FliSdkError` `setCoolingState` (`bool enable`)
- `FliSdkError` `setCoolingMode` (`CoolingMode mode`)
- `FliSdkError` `setCoolingFirstPoint` (`int16_t firstPoint`)
- `FliSdkError` `setCoolingStepWidth` (`uint8_t stepWidth`)
- `FliSdkError` `setSensorSetpoint` (`int16_t temp`)
- `FliSdkError` `getCoolingMode` (`CoolingMode &mode`)
- `FliSdkError` `getCoolingFirstPoint` (`int16_t &firstPoint`)
- `FliSdkError` `getCoolingStepWidth` (`uint8_t &stepWidth`)
- `FliSdkError` `getSensorSetpoint` (`int16_t &temp`)
- `FliSdkError` `getTecPower` (`double ¤t`, `double &voltage`, `double &power`)
- `FliSdkError` `getCoolingState` (`bool &enabled`)
- `FliSdkError` `getCurrentStep` (`std::string &step`)

Additional Inherited Members

6.7.1 Detailed Description

This class manages the methods specific to the C-RED 2 Lite camera.

6.7.2 Member Enumeration Documentation

6.7.2.1 CoolingMode

```
enum FliCredTwoLite::CoolingMode
```

Enumerator

<code>MANUAL</code>	
<code>AUTOMATIC</code>	
<code>CUSTOM_STEPS</code>	

6.7.3 Constructor & Destructor Documentation

6.7.3.1 FliCredTwoLite()

```
FliCredTwoLite::FliCredTwoLite (
    IFrameGrabberCL * grabber )
```

6.7.4 Member Function Documentation

6.7.4.1 getCoolingFirstPoint()

```
FliSdkError FliCredTwoLite::getCoolingFirstPoint (
    int16_t & firstPoint )
```

6.7.4.2 getCoolingMode()

```
FliSdkError FliCredTwoLite::getCoolingMode (
    CoolingMode & mode )
```

6.7.4.3 getCoolingState()

```
FliSdkError FliCredTwoLite::getCoolingState (
    bool & enabled )
```

6.7.4.4 getCoolingStepWidth()

```
FliSdkError FliCredTwoLite::getCoolingStepWidth (
    uint8_t & stepWidth )
```

6.7.4.5 getCurrentStep()

```
FliSdkError FliCredTwoLite::getCurrentStep (
    std::string & step )
```

6.7.4.6 getSensorSetpoint()

```
FliSdkError FliCredTwoLite::getSensorSetpoint (
    int16_t & temp )
```

6.7.4.7 getTecPower()

```
FliSdkError FliCredTwoLite::getTecPower (
    double & current,
    double & voltage,
    double & power )
```

6.7.4.8 setCoolingFirstPoint()

```
FliSdkError FliCredTwoLite::setCoolingFirstPoint (
    int16_t firstPoint )
```

6.7.4.9 setCoolingMode()

```
FliSdkError FliCredTwoLite::setCoolingMode (
    CoolingMode mode )
```

6.7.4.10 setCoolingState()

```
FliSdkError FliCredTwoLite::setCoolingState (
    bool enable )
```

6.7.4.11 setCoolingStepWidth()

```
FliSdkError FliCredTwoLite::setCoolingStepWidth (
    uint8_t stepWidth )
```

6.7.4.12 setSensorSetpoint()

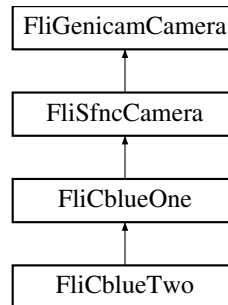
```
FliSdkError FliCredTwoLite::setSensorSetpoint (
    int16_t temp )
```

6.8 FliGenicamCamera Class Reference

This is the base class of all genicam camera (C-BLUE)

```
#include <FliGenicamCamera.h>
```

Inheritance diagram for FliGenicamCamera:



Public Member Functions

- [FliGenicamCamera](#) (IFrameGrabberGenicam *grabber)
- virtual [~FliGenicamCamera](#) ()
- Fli::CameraModel [getCameraModel](#) ()
Get the current camera model.
- IFrameGrabberGenicam * [getAssociatedGrabber](#) ()
Get the associated grabber.
- const std::map< std::string, BaseFeature * > & [getFeaturesList](#) ()
Get the list of all the camera features.
- bool [getStringFeature](#) (const std::string &featureName, std::string &val)
Get the string value of a feature.
- bool [setStringFeature](#) (const std::string &featureName, std::string val)
Set the string value of a feature.
- bool [getDoubleFeature](#) (const std::string &featureName, double &val)
Get the double value of a feature.
- bool [setDoubleFeature](#) (const std::string &featureName, double val)
Set the double value of a feature.
- bool [getIntegerFeature](#) (const std::string &featureName, int64_t &val)
Get the integer value of a feature.
- bool [setIntegerFeature](#) (const std::string &featureName, int64_t val)
Set the integer value of a feature.
- bool [getBooleanFeature](#) (const std::string &featureName, bool &val)
Get the boolean value of a feature.
- bool [setBooleanFeature](#) (const std::string &featureName, bool val)
Set the boolean value of a feature.
- bool [executeFeature](#) (const std::string &featureName)
Execute the desired feature.
- bool [getFeatureLength](#) (const std::string &featureName, int64_t &length)
Get the length in bytes of the feature.
- bool [getDoubleMinFeature](#) (const std::string &featureName, double &val)
Get the double minimum value of feature.

- bool [getDoubleMaxFeature](#) (const std::string &featureName, double &val)
Get the double maximum value of feature.
- bool [getIntegerMinFeature](#) (const std::string &featureName, int64_t &val)
Get the integer minimum value of feature.
- bool [getIntegerMaxFeature](#) (const std::string &featureName, int64_t &val)
Get the double maximum value of feature.
- bool [getIntegerIncrementFeature](#) (const std::string &featureName, int64_t &val)
Get the integer increment value of feature.
- bool [getDoubleIncrementFeature](#) (const std::string &featureName, double &val)
Get the double increment value of feature.
- bool [getPollingInterval](#) (const std::string &featureName, int64_t &interval)
Get the polling interval of feature.
- bool [getRepresentation](#) (const std::string &featureName, IFrameGrabberGenicam::Representation &repres)
Get the representation of feature.
- bool [getAccessMode](#) (const std::string &featureName, IFrameGrabberGenicam::AccessMode &mode)
Get the access mode of feature.
- bool [getVisibility](#) (const std::string &featureName, IFrameGrabberGenicam::Visibility &visibility)
Get the visibility of feature.
- bool [setRawData](#) (const std::string &featureName, const std::vector< unsigned char > &val)
Send raw data to feature.
- bool [getRawData](#) (const std::string &featureName, std::vector< unsigned char > &val, int64_t size=-1)
Get raw data from feature.
- void [addCallbackAllRegisters](#) (std::function< void(std::string, void *val)> callback, void *ptr)
Register a callback to all features.
- void [addCallbackDimensionsRegisters](#) (std::function< void(std::string, void *val)> callback, void *ptr)
Register a callback to all the dimensionnal features.
- void [removeCallbackAllRegisters](#) (void *ptr)
Remove all registers callback.

Protected Attributes

- IFrameGrabberGenicam * [_grabber](#)
- Fli::CameraModel [_cameraModel](#)
- std::map< std::string, BaseFeature * > [_stringToFeature](#)

6.8.1 Detailed Description

This is the base class of all genicam camera (C-BLUE)

6.8.2 Constructor & Destructor Documentation

6.8.2.1 FliGenicamCamera()

```
FliGenicamCamera::FliGenicamCamera (
    IFrameGrabberGenicam * grabber ) [explicit]
```

6.8.2.2 ~FliGenicamCamera()

```
virtual FliGenicamCamera::~~FliGenicamCamera ( ) [virtual]
```

6.8.3 Member Function Documentation

6.8.3.1 addCallbackAllRegisters()

```
void FliGenicamCamera::addCallbackAllRegisters (
    std::function< void(std::string, void *val)> callback,
    void * ptr )
```

Register a callback to all features.

Parameters

<i>callback</i>	: the function callback which is called with the feature name and a pointer to the value
<i>ptr</i>	: a pointer used for id

6.8.3.2 addCallbackDimensionsRegisters()

```
void FliGenicamCamera::addCallbackDimensionsRegisters (
    std::function< void(std::string, void *val)> callback,
    void * ptr )
```

Register a callback to all the dimensionnal features.

Parameters

<i>callback</i>	: the function callback which is called with the feature name and a pointer to the value
<i>ptr</i>	: a pointer used for id

6.8.3.3 executeFeature()

```
bool FliGenicamCamera::executeFeature (
    const std::string & featureName ) [inline]
```

Execute the desired feature.

Parameters

<i>featureName</i>	the desired feature name
--------------------	--------------------------

Returns

true if function is well done else false

6.8.3.4 getAccessMode()

```
bool FliGenicamCamera::getAccessMode (
    const std::string & featureName,
    IFrameGrabberGenicam::AccessMode & mode ) [inline]
```

Get the access mode of feature.

Parameters

<i>featureName</i>	the desired feature name
<i>mode</i>	the value container

Returns

true if function is well done else false

6.8.3.5 getAssociatedGrabber()

```
IFrameGrabberGenicam* FliGenicamCamera::getAssociatedGrabber ( )
```

Get the associated grabber.

Returns

Pointer on the grabber object

6.8.3.6 getBooleanFeature()

```
bool FliGenicamCamera::getBooleanFeature (
    const std::string & featureName,
    bool & val ) [inline]
```

Get the boolean value of a feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value container

Returns

true if function is well done else false

6.8.3.7 getCameraModel()

```
Fli::CameraModel FliGenicamCamera::getCameraModel ( )
```

Get the current camera model.

Returns

Camera cmodel

6.8.3.8 getDoubleFeature()

```
bool FliGenicamCamera::getDoubleFeature (
    const std::string & featureName,
    double & val ) [inline]
```

Get the double value of a feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value container

Returns

true if function is well done else false

6.8.3.9 getDoubleIncrementFeature()

```
bool FliGenicamCamera::getDoubleIncrementFeature (
    const std::string & featureName,
    double & val ) [inline]
```

Get the double increment value of feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value container

Returns

true if function is well done else false

6.8.3.10 getDoubleMaxFeature()

```
bool FliGenicamCamera::getDoubleMaxFeature (
    const std::string & featureName,
    double & val ) [inline]
```

Get the double maximum value of feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value container

Returns

true if function is well done else false

6.8.3.11 getDoubleMinFeature()

```
bool FliGenicamCamera::getDoubleMinFeature (
    const std::string & featureName,
    double & val ) [inline]
```

Get the double minimum value of feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value container

Returns

true if function is well done else false

6.8.3.12 getFeatureLength()

```
bool FliGenicamCamera::getFeatureLength (
    const std::string & featureName,
    int64_t & length ) [inline]
```

Get the length in bytes of the feature.

Parameters

<i>featureName</i>	the desired feature name
<i>length</i>	the value container

Returns

true if function is well done else false

6.8.3.13 getFeaturesList()

```
const std::map<std::string, BaseFeature*>& FliGenicamCamera::getFeaturesList ( )
```

Get the list of all the camera features.

Returns

A map which link the feature name to the BaseFeature object

6.8.3.14 getIntegerFeature()

```
bool FliGenicamCamera::getIntegerFeature (
    const std::string & featureName,
    int64_t & val ) [inline]
```

Get the integer value of a feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value container

Returns

true if function is well done else false

6.8.3.15 getIntegerIncrementFeature()

```
bool FliGenicamCamera::getIntegerIncrementFeature (
    const std::string & featureName,
    int64_t & val ) [inline]
```

Get the integer increment value of feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value container

Returns

true if function is well done else false

6.8.3.16 getIntegerMaxFeature()

```
bool FliGenicamCamera::getIntegerMaxFeature (
    const std::string & featureName,
    int64_t & val ) [inline]
```

Get the double maximum value of feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value container

Returns

true if function is well done else false

6.8.3.17 getIntegerMinFeature()

```
bool FliGenicamCamera::getIntegerMinFeature (
    const std::string & featureName,
    int64_t & val ) [inline]
```

Get the integer minimum value of feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value container

Returns

true if function is well done else false

6.8.3.18 getPollingInterval()

```
bool FliGenicamCamera::getPollingInterval (
    const std::string & featureName,
    int64_t & interval ) [inline]
```

Get the polling interval of feature.

Parameters

<i>featureName</i>	the desired feature name
<i>interval</i>	the value container

Returns

true if function is well done else false

6.8.3.19 getRawData()

```
bool FliGenicamCamera::getRawData (
    const std::string & featureName,
    std::vector< unsigned char > & val,
    int64_t size = -1 ) [inline]
```

Get raw data from feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the data container
<i>size</i>	: desired size

Returns

true if function is well done else false

6.8.3.20 getRepresentation()

```
bool FliGenicamCamera::getRepresentation (
    const std::string & featureName,
    IFrameGrabberGenicam::Representation & repres ) [inline]
```

Get the representation of feature.

Parameters

<i>featureName</i>	the desired feature name
<i>repres</i>	the value container

Returns

true if function is well done else false

6.8.3.21 getStringFeature()

```
bool FliGenicamCamera::getStringFeature (
    const std::string & featureName,
    std::string & val ) [inline]
```

Get the string value of a feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value container

Returns

true if function is well done else false

6.8.3.22 getVisibility()

```
bool FliGenicamCamera::getVisibility (
    const std::string & featureName,
    IFrameGrabberGenicam::Visibility & visibility ) [inline]
```

Get the visibility of feature.

Parameters

<i>featureName</i>	the desired feature name
<i>visibility</i>	the value container

Returns

true if function is well done else false

6.8.3.23 removeCallbackAllRegisters()

```
void FliGenicamCamera::removeCallbackAllRegisters (
    void * ptr )
```

Remove all registers callback.

Parameters

<i>ptr</i>	: a pointer used for id
------------	-------------------------

6.8.3.24 setBooleanFeature()

```
bool FliGenicamCamera::setBooleanFeature (
    const std::string & featureName,
    bool val ) [inline]
```

Set the boolean value of a feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value to send

Returns

true if function is well done else false

6.8.3.25 setDoubleFeature()

```
bool FliGenicamCamera::setDoubleFeature (
    const std::string & featureName,
    double val ) [inline]
```

Set the double value of a feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value to send

Returns

true if function is well done else false

6.8.3.26 setIntegerFeature()

```
bool FliGenicamCamera::setIntegerFeature (
    const std::string & featureName,
    int64_t val ) [inline]
```

Set the integer value of a feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value to send

Returns

true if function is well done else false

6.8.3.27 setRawData()

```
bool FliGenicamCamera::setRawData (
    const std::string & featureName,
    const std::vector< unsigned char > & val ) [inline]
```

Send raw data to feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the data container

Returns

true if function is well done else false

6.8.3.28 setStringFeature()

```
bool FliGenicamCamera::setStringFeature (
    const std::string & featureName,
    std::string val ) [inline]
```

Set the string value of a feature.

Parameters

<i>featureName</i>	the desired feature name
<i>val</i>	the value to send

Returns

true if function is well done else false

6.8.4 Member Data Documentation

6.8.4.1 `_cameraModel`

```
Fli::CameraModel FliGenicamCamera::_cameraModel [protected]
```

6.8.4.2 `_grabber`

```
IFrameGrabberGenicam* FliGenicamCamera::_grabber [protected]
```

6.8.4.3 `_stringToFeature`

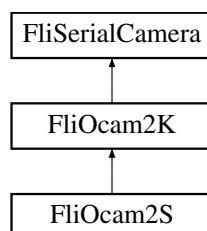
```
std::map<std::string, BaseFeature*> FliGenicamCamera::_stringToFeature [protected]
```

6.9 FliOcam2K Class Reference

This class manages the methods specific to the OCAM2K camera.

```
#include <FliOcam2K.h>
```

Inheritance diagram for FliOcam2K:



Public Member Functions

- [FliOcam2K](#) (IFrameGrabberCL *grabber)
- FliSdkError [setWorkMode](#) ([Ocam2Mode](#) mode)
- [Ocam2Conf](#) [getConfig](#) ()
- FliSdkError [getAllTemp](#) (double &ccdTemp, double &cpuTemp, double &powerTemp, double &biasTemp, double &waterTemp, double &leftTemp, double &rightTemp, double &setTemp)
- FliSdkError [enableBias](#) (bool enable)
 - enableBias enable or disable the bias into the camera*
- FliSdkError [enableFlat](#) (bool enable)
 - enableFlat enable or disable the flat into the camera*
- FliSdkError [protectionReset](#) ()
 - protectionReset send the command "protection reset" to the camera*
- FliSdkError [setBiasOffset](#) (unsigned int offset)
- FliSdkError [setFps](#) (double fps)
- FliSdkError [getFps](#) (double &fps)
- FliSdkError [getFpsMax](#) (double &fps)
- FliSdkError [setFpsMax](#) ()
- FliSdkError [setGain](#) (unsigned int gain)
- FliSdkError [sendBiasFile](#) (std::string filePath)
 - sendBiasFile send the bias file to the camera*
- FliSdkError [sendFlatFile](#) (std::string filePath)
 - sendFlatFile send the flat file to the camera*
- FliSdkError [getCoolingState](#) ([Ocam2CoolingState](#) &state)
- FliSdkError [getCoolingValue](#) (int64_t &val)
- FliSdkError [resetCoolingAlarm](#) ()
 - resetCoolingAlarm send the command "temp reset" to the camera*
- FliSdkError [disableCooling](#) ()
 - disableCooling send the command "cooling off" to the camera*
- FliSdkError [setCoolingValue](#) (int64_t val)

Protected Member Functions

- bool [isOldFirmware](#) ()
- bool [isInterface0](#) ()
- bool [sendInterface0Command](#) (const std::string &command, std::string &response, bool echo=false, int timeout=0)
- bool [sendInterface1Command](#) (const std::string &command, std::string &response, bool echo=false, int timeout=0)

Protected Attributes

- [Ocam2Conf](#) [_conf](#)

6.9.1 Detailed Description

This class manages the methods specific to the OCAM2K camera.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 FliOcam2K()

```
FliOcam2K::FliOcam2K (
    IFrameGrabberCL * grabber )
```

6.9.3 Member Function Documentation

6.9.3.1 disableCooling()

```
FliSdkError FliOcam2K::disableCooling ( )
```

disableCooling send the command "cooling off" to the camera

Returns

FLISDK_NO_ERROR if no error else an FliSdkError

6.9.3.2 enableBias()

```
FliSdkError FliOcam2K::enableBias (
    bool enable )
```

enableBias enable or disable the bias into the camera

Parameters

<i>enable</i>	: enable the bias if true, disable it if false
---------------	--

Returns

FLISDK_NO_ERROR if no error else an FliSdkError

6.9.3.3 enableFlat()

```
FliSdkError FliOcam2K::enableFlat (
    bool enable )
```

enableFlat enable or disable the flat into the camera

Parameters

<i>enable</i>	: enable the flat if true, disable it if false
---------------	--

Returns

FLISDK_NO_ERROR if no error else an `FliSdkError`

6.9.3.4 `getAllTemp()`

```
FliSdkError FliOcam2K::getAllTemp (
    double & ccdTemp,
    double & cpuTemp,
    double & powerTemp,
    double & biasTemp,
    double & waterTemp,
    double & leftTemp,
    double & rightTemp,
    double & setTemp )
```

6.9.3.5 `getConf()`

```
Ocam2Conf FliOcam2K::getConf ( )
```

6.9.3.6 `getCoolingState()`

```
FliSdkError FliOcam2K::getCoolingState (
    Ocam2CoolingState & state )
```

6.9.3.7 `getCoolingValue()`

```
FliSdkError FliOcam2K::getCoolingValue (
    int64_t & val )
```

6.9.3.8 `getFps()`

```
FliSdkError FliOcam2K::getFps (
    double & fps )
```

6.9.3.9 getFpsMax()

```
FliSdkError FliOcam2K::getFpsMax (
    double & fps )
```

6.9.3.10 isInterface0()

```
bool FliOcam2K::isInterface0 ( ) [protected]
```

6.9.3.11 isOldFirmware()

```
bool FliOcam2K::isOldFirmware ( ) [protected]
```

6.9.3.12 protectionReset()

```
FliSdkError FliOcam2K::protectionReset ( )
```

protectionReset send the command "protection reset" to the camera

Returns

FLISDK_NO_ERROR if no error else an FliSdkError

6.9.3.13 resetCoolingAlarm()

```
FliSdkError FliOcam2K::resetCoolingAlarm ( )
```

resetCoolingAlarm send the command "temp reset" to the camera

Returns

FLISDK_NO_ERROR if no error else an FliSdkError

6.9.3.14 sendBiasFile()

```
FliSdkError FliOcam2K::sendBiasFile (
    std::string filePath )
```

sendBiasFile send the bias file to the camera

Parameters

<i>filePath</i>	the file name and path that describe the bias
-----------------	---

Returns

FLISDK_NO_ERROR if no error else an FliSdkError

6.9.3.15 sendFlatFile()

```
FliSdkError FliOcam2K::sendFlatFile (
    std::string filePath )
```

sendFlatFile send the flat file to the camera

Parameters

<i>filePath</i>	the file name and path that describe the flat
-----------------	---

Returns

FLISDK_NO_ERROR if no error or an FliSdkError else

6.9.3.16 sendInterface0Command()

```
bool FliOcam2K::sendInterface0Command (
    const std::string & command,
    std::string & response,
    bool echo = false,
    int timeout = 0 ) [protected]
```

6.9.3.17 sendInterface1Command()

```
bool FliOcam2K::sendInterface1Command (
    const std::string & command,
    std::string & response,
    bool echo = false,
    int timeout = 0 ) [protected]
```

6.9.3.18 setBiasOffset()

```
FliSdkError FliOcam2K::setBiasOffset (
    unsigned int offset )
```

6.9.3.19 setCoolingValue()

```
FliSdkError FliOcam2K::setCoolingValue (
    int64_t val )
```

6.9.3.20 setFps()

```
FliSdkError FliOcam2K::setFps (
    double fps )
```

6.9.3.21 setFpsMax()

```
FliSdkError FliOcam2K::setFpsMax ( )
```

6.9.3.22 setGain()

```
FliSdkError FliOcam2K::setGain (
    unsigned int gain )
```

6.9.3.23 setWorkMode()

```
FliSdkError FliOcam2K::setWorkMode (
    Ocam2Mode mode )
```

6.9.4 Member Data Documentation

6.9.4.1 _conf

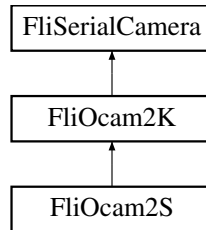
```
Ocam2Conf FliOcam2K::_conf [protected]
```

6.10 FliOcam2S Class Reference

This class manages the methods specific to the OCAM2S camera.

```
#include <FliOcam2S.h>
```

Inheritance diagram for FliOcam2S:



Public Member Functions

- [FliOcam2S](#) (IFrameGrabberCL *grabber)
- FliSdkError [enableShutter](#) (bool enable)
 - enableShutter set the shutter on or off*
- FliSdkError [setShutterInternal](#) ()
- FliSdkError [setShutterExternal](#) ()
- FliSdkError [setShutterSingle](#) ()
- FliSdkError [setShutterBurst](#) ()
- FliSdkError [setShutterSweepMode](#) (unsigned int mode)
- FliSdkError [setShutterPulseWidth](#) (unsigned int ns)
- FliSdkError [setShutterBlanking](#) (unsigned int ns)
- FliSdkError [setShutterPulsePosition](#) (unsigned int ns)
- FliSdkError [setShutterStep](#) (unsigned int ns)
- FliSdkError [setShutterEnd](#) (unsigned int ns)
- FliSdkError [setShutterPulseCount](#) (unsigned int count)
- FliSdkError [enableShutterBlockOnRead](#) (bool enable)
 - enableShutterBlockOnRead set the shutter blockonread to 1 or 0*
- FliSdkError [enableShutterCorrectGlitch](#) (bool enable)
 - enableShutterCorrectGlitch set the shutter correctglitch to 1 or 0*
- FliSdkError [getShutterState](#) (Shutter &shutter)
- FliSdkError [sendShutterBias](#) (std::string buf)
 - sendShutterBias load the shutter bias to the camera*

Additional Inherited Members

6.10.1 Detailed Description

This class manages the methods specific to the OCAM2S camera.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 FliOcam2S()

```
FliOcam2S::FliOcam2S (
    IFrameGrabberCL * grabber )
```

6.10.3 Member Function Documentation

6.10.3.1 enableShutter()

```
FliSdkError FliOcam2S::enableShutter (
    bool enable )
```

enableShutter set the shutter on or off

Parameters

<i>enable</i>	: set the shutter to "on" if true or "off" if false
---------------	---

Returns

FLISDK_NO_ERROR if no error else an FliSdkError

6.10.3.2 enableShutterBlockOnRead()

```
FliSdkError FliOcam2S::enableShutterBlockOnRead (
    bool enable )
```

enableShutterBlockOnRead set the shutter blockonread to 1 or 0

Parameters

<i>enable</i>	: set the shutter blockonread to "1" if true or "0" if false
---------------	--

Returns

FLISDK_NO_ERROR if no error else an FliSdkError

6.10.3.3 enableShutterCorrectGlitch()

```
FliSdkError FliOcam2S::enableShutterCorrectGlitch (
    bool enable )
```

enableShutterCorrectGlitch set the shutter correctglitch to 1 or 0

Parameters

<i>enable</i>	: set the shutter correctglitch to "1" if true or "0" if false
---------------	--

Returns

FLISDK_NO_ERROR if no error else an FliSdkError

6.10.3.4 getShutterState()

```
FliSdkError FliOcam2S::getShutterState (
    Shutter & shutter )
```

6.10.3.5 sendShutterBias()

```
FliSdkError FliOcam2S::sendShutterBias (
    std::string buf )
```

sendShutterBias load the shutter bias to the camera

Parameters

<i>buf</i>	the buffer that contain the bias to be loaded
------------	---

Returns

FLISDK_NO_ERROR if no error else an FliSdkError

6.10.3.6 setShutterBlanking()

```
FliSdkError FliOcam2S::setShutterBlanking (
    unsigned int ns )
```

6.10.3.7 setShutterBurst()

```
FliSdkError FliOcam2S::setShutterBurst ( )
```


6.10.3.8 setShutterEnd()

```
FliSdkError FliOcam2S::setShutterEnd (
    unsigned int ns )
```

6.10.3.9 setShutterExternal()

```
FliSdkError FliOcam2S::setShutterExternal ( )
```

6.10.3.10 setShutterInternal()

```
FliSdkError FliOcam2S::setShutterInternal ( )
```

6.10.3.11 setShutterPulseCount()

```
FliSdkError FliOcam2S::setShutterPulseCount (
    unsigned int count )
```

6.10.3.12 setShutterPulsePosition()

```
FliSdkError FliOcam2S::setShutterPulsePosition (
    unsigned int ns )
```

6.10.3.13 setShutterPulseWidth()

```
FliSdkError FliOcam2S::setShutterPulseWidth (
    unsigned int ns )
```

6.10.3.14 setShutterSingle()

```
FliSdkError FliOcam2S::setShutterSingle ( )
```

6.10.3.15 setShutterStep()

```
FliSdkError FliOcam2S::setShutterStep (
    unsigned int ns )
```

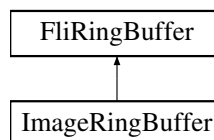
6.10.3.16 setShutterSweepMode()

```
FliSdkError FliOcam2S::setShutterSweepMode (
    unsigned int mode )
```

6.11 FliRingBuffer Class Reference

```
#include <FliRingBuffer.h>
```

Inheritance diagram for FliRingBuffer:



Public Member Functions

- virtual uint32_t [getFilling](#) () const =0
Get buffer filling.
- virtual int64_t [getLastImageIndex](#) () const =0
Get the last image acquired index.
- virtual void [setSizeInMo](#) (uint16_t sizeMo)=0
Change the buffer capacity in Mo.
- virtual void [setSizeInFrames](#) (uint32_t nbFrames)=0
Change the buffer capacity in number of images.
- virtual void [setSizeInFramesThermo](#) (uint32_t nbFrames)=0
Change the buffer capacity in number of images for a thermographic analysis (.thr.raw files)*
- virtual uint16_t [getSizeInMo](#) ()=0
Get current buffer size.
- virtual uint32_t [getSizeInFrames](#) ()=0
Give the images capacity of the buffer.
- virtual void [enable](#) (bool enable)=0
Enable or disable internal ring buffer of the SDK.
- virtual bool [isEnabled](#) ()=0
Return true if the buffer is enabled else false.
- virtual void [reset](#) ()=0
Reset the buffer.
- virtual void [enableGrabN](#) (uint32_t nbFrames)=0
Enable grab N mode, the buffer will stop copy images when nbFrames images is hit.

- virtual void `disableGrabN ()`=0
Disable grab N mode.
- virtual bool `isGrabNFinished ()` const =0
State of the grab N.
- virtual bool `isGrabNEnabled ()` const =0
State of the grab N mode.
- virtual uint32_t `getNumberOfWrap ()`=0
Get the number of times that the buffer had been full since reset.
- virtual void `enableSubstractMode (bool enable)`=0
Enable/disable the mode substract that will substract the image N by the image N-1 and save it in the buffer.
- virtual void `enableAccumulationMode (bool enable)`=0
- virtual void `resetAccumulation ()`=0
- virtual uint16_t `nbFramesInAccumulation ()`=0
- virtual uint64_t `getNbCountError ()`=0
Get the number of frame count error.
- virtual void `setFowlerOffset (uint16_t offset)`=0

6.11.1 Member Function Documentation

6.11.1.1 `disableGrabN()`

```
virtual void FliRingBuffer::disableGrabN ( ) [pure virtual]
```

Disable grab N mode.

Implemented in [ImageRingBuffer](#).

6.11.1.2 `enable()`

```
virtual void FliRingBuffer::enable (
    bool enable ) [pure virtual]
```

Enable or disable internal ring buffer of the SDK.

Attention

If ring buffer is disabled `loadBuffer`, `saveBuffer`, `getRawImage`, `getImage` and `getImage16b` will be disabled. `grabN` and error count will be disable too.

Implemented in [ImageRingBuffer](#).

6.11.1.3 enableAccumulationMode()

```
virtual void FliRingBuffer::enableAccumulationMode (
    bool enable ) [pure virtual]
```

Implemented in [ImageRingBuffer](#).

6.11.1.4 enableGrabN()

```
virtual void FliRingBuffer::enableGrabN (
    uint32_t nbFrames ) [pure virtual]
```

Enable grab N mode, the buffer will stop copy images when nbFrames images is hit.

Parameters

<i>nbFrames</i>	: number of frames to grab.
-----------------	-----------------------------

Implemented in [ImageRingBuffer](#).

6.11.1.5 enableSubstractMode()

```
virtual void FliRingBuffer::enableSubstractMode (
    bool enable ) [pure virtual]
```

Enable/disable the mode substract that will substract the image N by the image N-1 and save it in the buffer.

Parameters

<i>enable</i>	enable/disable the mode
---------------	-------------------------

Implemented in [ImageRingBuffer](#).

6.11.1.6 getFilling()

```
virtual uint32_t FliRingBuffer::getFilling ( ) const [pure virtual]
```

Get buffer filling.

Returns

a number representing the filling

Implemented in [ImageRingBuffer](#).

6.11.1.7 getLastImageIndex()

```
virtual int64_t FliRingBuffer::getLastImageIndex ( ) const [pure virtual]
```

Get the last image acquired index.

Returns

the index or -1 if no image in buffer

Implemented in [ImageRingBuffer](#).

6.11.1.8 getNbCountError()

```
virtual uint64_t FliRingBuffer::getNbCountError ( ) [pure virtual]
```

Get the number of frame count error.

Returns

the number of count error

Implemented in [ImageRingBuffer](#).

6.11.1.9 getNumberOfWrap()

```
virtual uint32_t FliRingBuffer::getNumberOfWrap ( ) [pure virtual]
```

Get the number of times that the buffer had been full since reset.

Returns

the number of wrap

Implemented in [ImageRingBuffer](#).

6.11.1.10 getSizeInFrames()

```
virtual uint32_t FliRingBuffer::getSizeInFrames ( ) [pure virtual]
```

Give the images capacity of the buffer.

Returns

FliSdkError

Implemented in [ImageRingBuffer](#).

6.11.1.11 getSizeInMo()

```
virtual uint16_t FliRingBuffer::getSizeInMo ( ) [pure virtual]
```

Get current buffer size.

Returns

the buffer size in Mo

Implemented in [ImageRingBuffer](#).

6.11.1.12 isEnabled()

```
virtual bool FliRingBuffer::isEnabled ( ) [pure virtual]
```

Return true if the buffer is enabled else false.

Implemented in [ImageRingBuffer](#).

6.11.1.13 isGrabNEnabled()

```
virtual bool FliRingBuffer::isGrabNEnabled ( ) const [pure virtual]
```

State of the grab N mode.

Returns

true if grab N mode activated else false

Implemented in [ImageRingBuffer](#).

6.11.1.14 isGrabNFinished()

```
virtual bool FliRingBuffer::isGrabNFinished ( ) const [pure virtual]
```

State of the grab N.

Returns

true if the grab is over else false

Implemented in [ImageRingBuffer](#).

6.11.1.15 nbFramesInAccumulation()

```
virtual uint16_t FliRingBuffer::nbFramesInAccumulation ( ) [pure virtual]
```

Implemented in [ImageRingBuffer](#).

6.11.1.16 reset()

```
virtual void FliRingBuffer::reset ( ) [pure virtual]
```

Reset the buffer.

Implemented in [ImageRingBuffer](#).

6.11.1.17 resetAccumulation()

```
virtual void FliRingBuffer::resetAccumulation ( ) [pure virtual]
```

Implemented in [ImageRingBuffer](#).

6.11.1.18 setFowlerOffset()

```
virtual void FliRingBuffer::setFowlerOffset (
    uint16_t offset ) [pure virtual]
```

Implemented in [ImageRingBuffer](#).

6.11.1.19 setSizeInFrames()

```
virtual void FliRingBuffer::setSizeInFrames (
    uint32_t nbFrames ) [pure virtual]
```

Change the buffer capacity in number of images.

Parameters

<i>nbFrames</i>	: capacity of the ring buffer in nb images
-----------------	--

Implemented in [ImageRingBuffer](#).

6.11.1.20 setSizeInFramesThermo()

```
virtual void FliRingBuffer::setSizeInFramesThermo (
    uint32_t nbFrames ) [pure virtual]
```

Change the buffer capacity in number of images for a thermographic analysis (*.thr.raw files)

Parameters

<code>nbFrames</code>	: capacity of the ring buffer in nb images
-----------------------	--

Implemented in [ImageRingBuffer](#).

6.11.1.21 setSizeInMo()

```
virtual void FliRingBuffer::setSizeInMo (
    uint16_t sizeMo ) [pure virtual]
```

Change the buffer capacity in Mo.

Parameters

<code>sizeMo</code>	: capacity of the ring buffer in Mo
---------------------	-------------------------------------

Implemented in [ImageRingBuffer](#).

6.12 FliSdk Class Reference

This class manages the interface with the camera and the grabber.

```
#include <FliSdk.h>
```

Public Types

- enum [Mode](#) { [Full](#), [GrabOnly](#), [ConfigOnly](#) }

Public Member Functions

- [FliSdk](#) ()
Constructor.
- virtual [~FliSdk](#) ()
Destructor.
- [FliSdk](#) (const [FliSdk](#) &)=delete
- [FliSdk](#) & [operator=](#) (const [FliSdk](#) &)=delete
- std::vector< std::string > [detectGrabbers](#) ()

- Start the grabbers detection.*

 - `std::vector< std::string > detectCameras` (bool *invertedCl=nullptr)

Start the cameras detection.

 - `std::vector< std::string > detectCameras` (const std::vector< std::string > &grabbers, bool *invertedCl=nullptr)

Start the cameras detection only using the grabbers in "grabbers" list.

 - `std::vector< std::string > detectEthernetCameras` (const std::vector< std::string > &ips, const std::string &sshLogin, const std::string &sshPassword)

Start the cameras detection only using the grabbers in "grabbers" list.

 - `bool setCamera` (std::string cameraName)

Set the camera to be used.

 - `std::string getCurrentCameraName` () const

returns the current camera name

 - `bool isCurrentCameraLink` () const

returns true if the current camera has a camera link, false otherwise

 - `bool setGrabber` (std::string grabberName)

Set the grabber to be used.

 - `void setMode` (Mode mode)

Set the mode of use of the sdk.

 - `Mode getMode` ()

Get the current mode of use of the sdk.

 - `void setImageDimension` (uint16_t width, uint16_t height)

Force the image dimension apply to the grabber.

 - `void setImageDimensionImageRingBuffer` (uint16_t width, uint16_t height)

Force the image dimension apply to the image ring buffer needed when there is no grabber but images loaded from files.

 - `void setImageDimensionImageRingBufferThermo` (uint16_t width, uint16_t height)

Force the image dimension apply to the thermographic image ring buffer needed when there is no grabber but thermographic images loaded from files (.thr.raw files)*

 - `FliSdkError update` ()

Update the changes, must be call after setCamera, setGrabber or setMode to take effects.

 - `std::vector< std::string > getDetectedCameras` () const

Get the detected cameras names.

 - `std::vector< std::string > getDetectedGrabbers` () const

Get the detected grabbers names.

 - `void forceCurrentCameraModel` (Fli::CameraModel model)

use this function when a camera is undefined

 - `Fli::CameraModel getCurrentCameraModel` ()

returns the current camera model

 - `FliSdkError start` ()

Start the grabber (must be initialized before)

 - `FliSdkError stop` ()

Stop the grabber.

 - `bool isStarted` () const

Get the state of the grabber (started or stopped)

 - `FliCredOne * credOne` ()

Get C-RED One camera interface.

 - `FliCredTwo * credTwo` ()

Get C-RED 2 camera interface.

 - `FliCredTwoLite * credTwoLite` ()

Get C-RED 2 LITE camera interface.

- [FliCredThree](#) * [credThree](#) ()
Get C-RED 3 camera interface.
- [FliOcam2K](#) * [ocam2k](#) ()
Get Ocam2K camera interface.
- [FliOcam2S](#) * [ocam2s](#) ()
Get Ocam2S camera interface.
- [FliCamera](#) * [camera](#) ()
Get general camera interface for C-RED cameras (deprecated, use [cred\(\)](#) instead)
- [FliSerialCamera](#) * [serialCamera](#) ()
Get common camera interface for C-RED and OCAM2 cameras.
- [FliCred](#) * [cred](#) ()
Get general camera interface for C-RED cameras.
- [FliGenicamCamera](#) * [genicamCamera](#) ()
Get commond interface for genicam camera.
- [FliSfncCamera](#) * [sfncCamera](#) ()
Get commond interface for genicam camera.
- [FliSfncCamera](#) * [cblueSfnc](#) ()
Get Cblue SFNC camera interface (DEPRECATED please use [sfncCamera](#) instead)
- [FliCblueOne](#) * [cblueOne](#) ()
Get C-BLUE One camera interface.
- [FliCblueTwo](#) * [cblueTwo](#) ()
Get C-BLUE 2 camera interface.
- void [setNblImagesPerBuffer](#) (uint8_t nblImages)
Set set number of images the grabber should acquire before trigger, use this function for high FPS.
- [FliRingBuffer](#) & [ringBuffer](#) ()
Return the interface used to interact with the SDK ring buffer.
- FliSdkError [enableGrabN](#) (uint32_t nbFrames)
Enable grab N mode.
- FliSdkError [disableGrabN](#) ()
Disable grab N mode.
- bool [isGrabNFinished](#) ()
State of the grab N.
- bool [isGrabNEnabled](#) ()
State of the grab N mode.
- const unsigned char * [getRawImage](#) (int64_t index=-1)
Get the image at index or the last image if index is -1, without processing.
- double [getRealFps](#) () const
Get the buffer acquisition rate.
- uint32_t [getBufferFilling](#) ()
Get buffer filling.
- void [setBufferSize](#) (uint16_t sizeMo)
Change the buffer capacity in Mo.
- void [setBufferSizeInImages](#) (uint64_t nblImages)
Change the buffer capacity in number of images.
- uint16_t [getBufferSize](#) ()
Get current buffer size.
- uint32_t [getBufferNbTimesFull](#) ()
Get the number of times that the buffer had been full since reset.
- FliSdkError [loadBuffer](#) (const std::string &path, CroppingData &bufferCrop)
Load a buffer from a file, in the ringBuffer of the SDK.
- FliSdkError [loadBuffer](#) (const std::string &path, Fli::LoadBufferInfo &bufferInfo, bool inRingBuffer=false)

- Load a buffer from a file, allocate memory, and return that memory to the user.*

 - FliSdkError **loadBuffer** (const uint8_t *buffer, uint32_t nblImages, uint64_t imageSize=0)
- Load a buffer in the ringBuffer of the SDK.*

 - bool **isUnsignedPixel** ()
- Return the pixel sign (int16 or uint16)*

 - bool **isMono8Pixel** ()
- Return the pixel size (1 byte if true, 2 bytes if false)*

 - void **enableMono8Pixel** (bool enable)
 - void **enableMono8PixelThermo** ()
 - void **enableUnsignedPixel** (bool enable)
- Change the pixel sign (int16 or uint16)*

 - void **resetBuffer** ()
- Reset the buffer.*

 - void **enableRingBuffer** (bool enable)
- Enable or disable internal ring buffer of the SDK.*

 - const std::map< std::string, std::string > **getAvailableSaveFormats** () const
- Return a map with the full name of the save format in the key and the extension in the value Example <"TIFF", ".tif">*

 - FliSdkError **saveBuffer** (std::string path, uint32_t start, uint32_t end, std::function< bool(int)> progression← Callback=nullptr, bool withMetadata=false, uint16_t offset=0, bool forceUnsigned=false, uint16_t decimation=1)
- Save the buffer at path.*

 - FliSdkError **saveBuffer** (const std::string &path, const Fli::LoadBufferInfo &info, uint32_t start, uint32_t end, std::function< bool(int)> progression← Callback=nullptr, bool withMetadata=false, uint16_t offset=0, bool forceUnsigned=false, uint16_t decimation=1, ProcessingId id=-1)
- Save as above but with a LoadBufferInfo struct.*

 - uint32_t **getImagesCapacity** ()
- Give the images capacity of the buffer.*

 - uint8_t * **getImage** (int64_t index=-1, ProcessingId id=-1)
- Get the RGB processed image at the given index, if no index then the last image is processed The buffer is overwritten only when the function is recalled.*

 - unsigned int **getSize** (ProcessingId id=-1)
- getSize return the total size of the buffer returned by getImage*

 - uint8_t * **getImage16b** (int64_t index=-1, ProcessingId id=-1)
- Get the 16bits grayscale processed image at the given index, if no index then the last image is processed The buffer is overwritten only when the function is recalled.*

 - **ImageProcessing** * **imageProcessing** (ProcessingId id=-1)
- A pointer to the image processing interface.*

 - FliSdkError **isCroppingDataValid** (CroppingData croppingData)
- Check if the cropping data is valid for Cred2 & Cred3.*

 - FliSdkError **isCroppingDataValid** (std::string columns, std::string rows)
- Check if the cropping data is valid for Cred1.*

 - FliSdkError **getCroppingState** (bool &enabled, CroppingData &croppingData)
- Get the cropping data from the camera.*

 - FliSdkError **setCroppingState** (bool enable, CroppingData croppingData)
- Set the cropping data.*

 - void **getCurrentImageDimension** (uint16_t &width, uint16_t &height)
- Get the current image dimension considering cropping.*

 - uint64_t **getNbCountError** ()
- Get the number of frame count error.*

 - unsigned int **getOcamFrameNumber** (int64_t index=-1)
- Get the frame number of ocam image at index.*

 - void **setOcamFrameNumberOffset** (uint8_t offset)

- Set the offset for the frame number.*

 - void [addRawImageReceivedObserver](#) ([IRawImageReceivedObserver](#) *obs, bool beforeCopy=true)

Add an observer on the raw image received.
- void [removeRawImageReceivedObserver](#) ([IRawImageReceivedObserver](#) *obs)
- Remove an observer.*
- std::string [version](#) ()
- Get the version of the sdk.*
- void [display8bImage](#) (uint8_t *image, std::string windowName="")
- Open an Opencv window to display image.*
- void [display16bImage](#) (uint8_t *image, std::string windowName="", bool unsignedPixel=true)
- Open an Opencv window to display image.*
- void [enableObserversNotif](#) (bool enable)
- enable observer to be notified*
- bool [observersNotifEnabled](#) ()
- void [addObserver](#) ([IFliSdkObserver](#) *obs)
- add a FliSdk state observer*
- void [removeObserver](#) ([IFliSdkObserver](#) *obs)
- remove a FliSdk state observer*
- ProcessingId [addImageProcessing](#) ()
- add an image processing and return an id*
- void [removeImageProcessing](#) (ProcessingId id)
- remove an image processing*
- void [setBurstFilter](#) (int16_t id)
- Set the burst filter for id.*
- int16_t [getBurstFilter](#) ()
- Get the current burst filter id applied.*
- void [initLog](#) (std::string appName)
- init SDK logging*
- void [log](#) (const std::string &text)
- add text to log file*
- void [logOutside](#) (const std::string &text)
- add text to log file from outside the SDK*
- void [enableSubstractMode](#) (bool enable)
- Enable/disable the mode substract that will substract the image N by the image N-1 and save it in the buffer.*
- void [enableFowlerProcessing](#) (bool enable)
- enable the fowler processing for Cred1*
- FliSdkError [addEthernetCamera](#) (std::string ip, std::string userName, std::string sshPassword, std::string &cameraName)
- Try to detect an ethernet camera and add it in the list.*
- FliSdkError [addFakeEthernetCamera](#) (std::string ip, uint16_t comPort, uint16_t streamPort, std::string &cameraName, bool genicam=false)
- Add fake ethernet camera (use with FliFakeCamera software)*
- FliSdkError [enableImageTagStateChanged](#) (bool enabled)
- Change Image Tag State to be used when mode Grab Only with web software link to camera that will set the Tags inside the camera in parallel.*
- void [defineGrabOnlySlowMode](#) (bool slowmode)
- For grabb only mode the user need to define if the camera is in slowmode or not.*
- void [enablePowerOverCXP](#) (bool enable)
- Enable the power on the CXP cables, only available for Matrox CXP Grabber.*
- void [enableOsForCCsFrameGrabber](#) (bool enable)
- Enable or disable the M_IO_SOURCE of CC, available only for Matrox CL Grabber.*

- void [setupFixedCCsFrameGrabber](#) (int set)
Set the CC IO_SOURCE for one set (and the other 3 will be unset), available only for Matrox CL Grabber.
- bool [openMatroxGenicamBrowser](#) ()
Open CXP Matrox browser.
- void [setFowlerOffset](#) (uint16_t offset)
Set the value of the Fowler offset to apply on sum of the images.
- void [enableFollowUpTheRamp](#) (bool enable)
Enable the initialisation of the pixel sum for the follow up.
- bool [getGrabberIsUSB](#) ()
Find out if the grabber is an USB one.
- IFrameGrabber * [getCurrentGrabber](#) ()
getCurrentGrabber getter of the pointer to the current grabber
- bool [exitAllGrabbers](#) ()
Exit (calling method exit()) for all the grabber to be done before a delete and outside any destructor.
- std::list< IFrameGrabber * > [listAllGrabbers](#) ()
listAllGrabbers return the list of pointers of all the current grabbers
- int [detectOneCamera](#) (IFrameGrabber *aGrabber)
detectOneCamera detect only one camera linked to a given grabber

6.12.1 Detailed Description

This class manages the interface with the camera and the grabber.

6.12.2 Member Enumeration Documentation

6.12.2.1 Mode

```
enum FliSdk::Mode
```

Enumerator

Full	Mode Full when the grabber and the serial port are both opened for the application.
GrabOnly	Mode GrabOnly when the grabber is opened but the serial port is closed.
ConfigOnly	Mode ConfigOnly when the serial port is opened but the grabber is closed.

6.12.3 Constructor & Destructor Documentation

6.12.3.1 FliSdk() [1/2]

```
FliSdk::FliSdk ( )
```

Constructor.

6.12.3.2 ~FliSdk()

```
virtual FliSdk::~~FliSdk ( ) [virtual]
```

Destructor.

6.12.3.3 FliSdk() [2/2]

```
FliSdk::FliSdk (
    const FliSdk & ) [delete]
```

6.12.4 Member Function Documentation

6.12.4.1 addEthernetCamera()

```
FliSdkError FliSdk::addEthernetCamera (
    std::string ip,
    std::string userName,
    std::string sshPassword,
    std::string & cameraName )
```

Try to detect an ethernet camera and add it in the list.

Parameters

<i>ip</i>	: ip of the camera or a range of ip to auto detect (ex: 192.168.0.1-60)
<i>userName</i>	: the ssh user name of the camera
<i>sshPassword</i>	: the ssh password of the camera
<i>cameraName</i>	: return the detected camera name

6.12.4.2 addFakeEthernetCamera()

```
FliSdkError FliSdk::addFakeEthernetCamera (
    std::string ip,
    uint16_t comPort,
    uint16_t streamPort,
    std::string & cameraName,
    bool genicam = false )
```

Add fake ethernet camera (use with FliFakeCamera software)

Parameters

<i>ip</i>	: ip of the fake camera
<i>comPort</i>	: port for com
<i>streamPort</i>	: port for video stream
<i>cameraName</i>	: return the detected camera name
<i>genicam</i>	: set true if is a genicam camera else false

6.12.4.3 addImageProcessing()

```
ProcessingId FliSdk::addImageProcessing ( )
```

add an image processing and return an id

6.12.4.4 addObserver()

```
void FliSdk::addObserver (
    IFliSdkObserver * obs )
```

add a [FliSdk](#) state observer

6.12.4.5 addRawImageReceivedObserver()

```
void FliSdk::addRawImageReceivedObserver (
    IRawImageReceivedObserver * obs,
    bool beforeCopy = true )
```

Add an observer on the raw image received.

Parameters

<i>obs</i>	pointer on the observer
<i>beforeCopy</i>	if true then the observer will be notified before the copy in the ringbuffer (image from grabber), else after the copy in the ringBuffer (image from ringBuffer). if beforeCopy is set to true, user will have only the time of the buffer overflow of the grabber but less time between the grabber and the notification. if beforeCopy is set to false, user will have more time because the ringBuffer can be bigger than the grabber buffer but it will have a copy between the grabber and the notification. If you want to switch from before to after or after to before then call <code>removeRawImageReceivedObserver</code> before <code>addRawImageReceivedObserver</code>

6.12.4.6 camera()

```
FliCamera* FliSdk::camera ( )
```

Get general camera interface for C-RED cameras (deprecated, use [cred\(\)](#) instead)

Returns

a pointer to the camera object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.7 cblueOne()

```
FliCblueOne* FliSdk::cblueOne ( )
```

Get C-BLUE One camera interface.

Returns

a pointer to the camera object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.8 cblueSfnc()

```
FliSfncCamera* FliSdk::cblueSfnc ( )
```

Get Cblue SFNC camera interface (DEPRECATED please use [sfncCamera](#) instead)

Returns

a pointer to the camera object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.9 cblueTwo()

```
FliCblueTwo* FliSdk::cblueTwo ( )
```

Get C-BLUE 2 camera interface.

Returns

a pointer to the camera object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.10 cred()

```
FliCred* FliSdk::cred ( )
```

Get general camera interface for C-RED cameras.

Returns

a pointer to the camera object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.11 credOne()

```
FliCredOne* FliSdk::credOne ( )
```

Get C-RED One camera interface.

Returns

a pointer to the camera object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.12 credThree()

```
FliCredThree* FliSdk::credThree ( )
```

Get C-RED 3 camera interface.

Returns

a pointer to the camera object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.13 credTwo()

```
FliCredTwo* FliSdk::credTwo ( )
```

Get C-RED 2 camera interface.

Returns

a pointer to the camera object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.14 credTwoLite()

```
FliCredTwoLite* FliSdk::credTwoLite ( )
```

Get C-RED 2 LITE camera interface.

Returns

a pointer to the camera object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.15 defineGrabOnlySlowMode()

```
void FliSdk::defineGrabOnlySlowMode (
    bool slowmode )
```

For grab only mode the user need to define if the camera is in slowmode or not.

Parameters

<code>slowmode</code>	: a boolean to define the slowmode (true) or the normal mode (false)
-----------------------	--

6.12.4.16 detectCameras() [1/2]

```
std::vector<std::string> FliSdk::detectCameras (
    bool * invertedCl = nullptr )
```

Start the cameras detection.

Returns

a list with the names of detected cameras

Attention

This function must be called after [detectGrabbers\(\)](#)

6.12.4.17 detectCameras() [2/2]

```
std::vector<std::string> FliSdk::detectCameras (
    const std::vector< std::string > & grabbers,
    bool * invertedCl = nullptr )
```

Start the cameras detection only using the grabbers in "grabbers" list.

Returns

a list with the names of detected cameras

Attention

This function must be called after [detectGrabbers\(\)](#)

6.12.4.18 detectEthernetCameras()

```
std::vector<std::string> FliSdk::detectEthernetCameras (
    const std::vector< std::string > & ips,
    const std::string & sshLogin,
    const std::string & sshPassword )
```

Start the cameras detection only using the grabbers in "grabbers" list.

Returns

a list with the names of detected cameras

Attention

This function must be called after [detectGrabbers\(\)](#)

6.12.4.19 detectGrabbers()

```
std::vector<std::string> FliSdk::detectGrabbers ( )
```

Start the grabbers detection.

Returns

a list with the names of detected grabbers

Attention

This function must be called before [detectCameras\(\)](#)

6.12.4.20 detectOneCamera()

```
int FliSdk::detectOneCamera (
    IFrameGrabber * aGrabber )
```

detectOneCamera detect only one camera linked to a given grabber

Parameters

<i>aGrabber</i>	the grabber to use for the detection
-----------------	--------------------------------------

Returns

0 no camera 1 or more a camera is connected

6.12.4.21 disableGrabN()

```
FliSdkError FliSdk::disableGrabN ( )
```

Disable grab N mode.

Returns

FliSdkError

6.12.4.22 display16bImage()

```
void FliSdk::display16bImage (
    uint8_t * image,
    std::string windowName = "",
    bool unsignedPixel = true )
```

Open an Opencv window to display image.

Parameters

<i>image</i>	pointer to the image buffer
<i>windowName</i>	name of the window
<i>unsignedPixel</i>	indicate if pixel are signed/unsigned

6.12.4.23 display8bImage()

```
void FliSdk::display8bImage (
    uint8_t * image,
    std::string windowName = "" )
```

Open an Opencv window to display image.

Parameters

<i>image</i>	pointer to the image buffer
<i>windowName</i>	name of the window

6.12.4.24 enableFollowUpTheRamp()

```
void FliSdk::enableFollowUpTheRamp (
    bool enable )
```

Enable the initialisation of the pixel sum for the follow up.

Parameters

<i>enable</i>	true to enable, else false
---------------	----------------------------

6.12.4.25 enableFowlerProcessing()

```
void FliSdk::enableFowlerProcessing (
    bool enable )
```

enable the fowler processing for Cred1

6.12.4.26 enableGrabN()

```
FliSdkError FliSdk::enableGrabN (  
    uint32_t nbFrames )
```

Enable grab N mode.

Parameters

<i>nbFrames</i>	: number of frames to grab.
-----------------	-----------------------------

Returns

FliSdkError

6.12.4.27 enableImageTagStateChanged()

```
FliSdkError FliSdk::enableImageTagStateChanged (
    bool enabled )
```

Change Image Tag State to be used when mode Grab Only with web software link to camera that will set the Tags inside the camera in parallel.

Parameters

<i>enabled</i>	: a boolean to enable (true) / disable (false) the state
----------------	--

6.12.4.28 enableIOsForCCsFrameGrabber()

```
void FliSdk::enableIOsForCCsFrameGrabber (
    bool enable )
```

Enable or disable the M_IO_SOURCE of CC, available only for Matrox CL Grabber.

Parameters

<i>enable</i>	if true or disable if false the M_IO_SOURCE of CC
---------------	---

6.12.4.29 enableMono8Pixel()

```
void FliSdk::enableMono8Pixel (
    bool enable )
```

6.12.4.30 enableMono8PixelThermo()

```
void FliSdk::enableMono8PixelThermo ( )
```

6.12.4.31 enableObserversNotif()

```
void FliSdk::enableObserversNotif (
    bool enable )
```

enable observer to be notified

6.12.4.32 enablePowerOverCXP()

```
void FliSdk::enablePowerOverCXP (
    bool enable )
```

Enable the power on the CXP cables, only available for Matrox CXP Grabber.

Parameters

<i>enable</i>	the power if true, else disable
---------------	---------------------------------

6.12.4.33 enableRingBuffer()

```
void FliSdk::enableRingBuffer (
    bool enable )
```

Enable or disable internal ring buffer of the SDK.

Attention

If ring buffer is disabled loadBuffer, saveBuffer, getRawImage, getImage and getImage16b will be disabled. grabN and error count will be disable too.

6.12.4.34 enableSubstractMode()

```
void FliSdk::enableSubstractMode (
    bool enable )
```

Enable/disable the mode substract that will substract the image N by the image N-1 and save it in the buffer.

Parameters

<i>enable</i>	enable/disable the mode
---------------	-------------------------

6.12.4.35 enableUnsignedPixel()

```
void FliSdk::enableUnsignedPixel (
    bool enable )
```

Change the pixel sign (int16 or uint16)

6.12.4.36 exitAllGrabbers()

```
bool FliSdk::exitAllGrabbers ( )
```

Exit (calling method exit()) for all the grabber to be done before a delete and outside any destructor.

Returns

true if the exits succeeded, false otherwise

6.12.4.37 forceCurrentCameraModel()

```
void FliSdk::forceCurrentCameraModel (
    Fli::CameraModel model )
```

use this function when a camera is undefined

Parameters

<i>model</i>	: model to set to the camera
--------------	------------------------------

6.12.4.38 genicamCamera()

```
FliGenicamCamera* FliSdk::genicamCamera ( )
```

Get command interface for genicam camera.

Returns

a pointer to the camera object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.39 `getAvailableSaveFormats()`

```
const std::map<std::string, std::string> FliSdk::getAvailableSaveFormats ( ) const
```

Return a map with the full name of the save format in the key and the extension in the value Example <"TIFF", ".tif">

6.12.4.40 `getBufferFilling()`

```
uint32_t FliSdk::getBufferFilling ( )
```

Get buffer filling.

Returns

a number representing the filling

6.12.4.41 `getBufferNbTimesFull()`

```
uint32_t FliSdk::getBufferNbTimesFull ( )
```

Get the number of times that the buffer had been full since reset.

Returns

the number

6.12.4.42 `getBufferSize()`

```
uint16_t FliSdk::getBufferSize ( )
```

Get current buffer size.

Returns

the buffer size in Mo

6.12.4.43 `getBurstFilter()`

```
int16_t FliSdk::getBurstFilter ( )
```

Get the current burst filter id applied.

6.12.4.44 `getCroppingState()`

```
FliSdkError FliSdk::getCroppingState (
    bool & enabled,
    CroppingData & croppingData )
```

Get the cropping data from the camera.

Parameters

<i>enabled</i>	a reference to a bool from the user
<i>croppingData</i>	a reference of the cropping data of user

Returns

FliSdkError

6.12.4.45 getCurrentCameraModel()

```
Fli::CameraModel FliSdk::getCurrentCameraModel ( )
```

returns the current camera model

6.12.4.46 getCurrentCameraName()

```
std::string FliSdk::getCurrentCameraName ( ) const
```

returns the current camera name

6.12.4.47 getCurrentGrabber()

```
IFrameGrabber* FliSdk::getCurrentGrabber ( )
```

getCurrentGrabber getter of the pointer to the current grabber

Returns

the pointer to the current grabber or nullptr if none

6.12.4.48 getCurrentImageDimension()

```
void FliSdk::getCurrentImageDimension (
    uint16_t & width,
    uint16_t & height )
```

Get the current image dimension considering cropping.

Parameters

<i>width</i>	reference to user variable width
<i>height</i>	reference to user variable height

6.12.4.49 getDetectedCameras()

```
std::vector<std::string> FliSdk::getDetectedCameras ( ) const
```

Get the detected cameras names.

Returns

a list with the names of detected cameras

6.12.4.50 getDetectedGrabbers()

```
std::vector<std::string> FliSdk::getDetectedGrabbers ( ) const
```

Get the detected grabbers names.

Returns

a list with the names of detected grabbers

6.12.4.51 getGrabberIsUSB()

```
bool FliSdk::getGrabberIsUSB ( )
```

Find out if the grabber is an USB one.

Returns

true if the grabber is an USB one, false otherwise

6.12.4.52 getImage()

```
uint8_t* FliSdk::getImage (
    int64_t index = -1,
    ProcessingId id = -1 )
```

Get the RGB processed image at the given index, if no index then the last image is processed The buffer is overwritten only when the function is recalled.

Parameters

<i>index</i>	: index of the image in the buffer
<i>id</i>	: id of the imageProcessing to get the image

Returns

a pointer to the processed image

6.12.4.53 getImage16b()

```
uint8_t* FliSdk::getImage16b (
    int64_t index = -1,
    ProcessingId id = -1 )
```

Get the 16bits grayscale processed image at the given index, if no index then the last image is processed The buffer is overwritten only when the function is recalled.

Parameters

<i>index</i>	: index of the image in the buffer
<i>id</i>	: id of the imageProcessing to get the image

Returns

a pointer to the processed image

6.12.4.54 getImagesCapacity()

```
uint32_t FliSdk::getImagesCapacity ( )
```

Give the images capacity of the buffer.

Returns

FliSdkError

6.12.4.55 getMode()

```
Mode FliSdk::getMode ( )
```

Get the current mode of use of the sdk.

Returns

mode used (full, grabOnly, configOnly)

6.12.4.56 getNbCountError()

```
uint64_t FliSdk::getNbCountError ( )
```

Get the number of frame count error.

Returns

the number of count error

6.12.4.57 getOcamFrameNumber()

```
unsigned int FliSdk::getOcamFrameNumber (
    int64_t index = -1 )
```

Get the frame number of ocam image at index.

Parameters

<i>index</i>	index of the image in the buffer, -1 is last image
--------------	--

Returns

the frame number of the image at index

6.12.4.58 getRawImage()

```
const unsigned char* FliSdk::getRawImage (
    int64_t index = -1 )
```

Get the image at index or the last image if index is -1, without processing.

Parameters

<i>index</i>	: index of the image in the buffer.
--------------	-------------------------------------

Returns

pointer to the image array if index is valid else nullptr

6.12.4.59 getRealFps()

```
double FliSdk::getRealFps ( ) const
```

Get the buffer acquisition rate.

Returns

a number representing acquisition FPS

6.12.4.60 getSize()

```
unsigned int FliSdk::getSize (
    ProcessingId id = -1 )
```

getSize return the total size of the buffer returned by getImage

Parameters

<i>id</i>	: id of the imageProcessing to get the image
-----------	--

Returns

the total size of the buffer

6.12.4.61 imageProcessing()

```
IImageProcessing* FliSdk::imageProcessing (
    ProcessingId id = -1 )
```

A pointer to the image processing interface.

Parameters

<i>id</i>	: id of the imageProcessing
-----------	-----------------------------

Returns

pointer of type [IImageProcessing](#)

6.12.4.62 initLog()

```
void FliSdk::initLog (
    std::string appName )
```

init SDK logging

Parameters

<i>appName</i>	: appName will be used for the name of the file
----------------	---

6.12.4.63 isCroppingDataValid() [1/2]

```
FliSdkError FliSdk::isCroppingDataValid (
    CroppingData croppingData )
```

Check if the cropping data is valid for Cred2 & Cred3.

Parameters

<i>croppingData</i>	the cropping data of user
---------------------	---------------------------

Returns

FliSdkError

6.12.4.64 isCroppingDataValid() [2/2]

```
FliSdkError FliSdk::isCroppingDataValid (
    std::string columns,
    std::string rows )
```

Check if the cropping data is valid for Cred1.

Parameters

<i>columns</i>	columns cropping
<i>rows</i>	rows cropping

Returns

FliSdkError

6.12.4.65 isCurrentCameraLink()

```
bool FliSdk::isCurrentCameraLink ( ) const
```

returns true if the current camera has a camera link, false otherwise

6.12.4.66 isGrabNEnabled()

```
bool FliSdk::isGrabNEnabled ( )
```

State of the grab N mode.

Returns

true if grab N mode activated else false

6.12.4.67 isGrabNFinished()

```
bool FliSdk::isGrabNFinished ( )
```

State of the grab N.

Returns

true if the grab is over else false

6.12.4.68 isMono8Pixel()

```
bool FliSdk::isMono8Pixel ( )
```

Return the pixel size (1 byte if true, 2 bytes if false)

6.12.4.69 isStarted()

```
bool FliSdk::isStarted ( ) const
```

Get the state of the grabber (started or stopped)

Returns

true if grabber is started else false

6.12.4.70 isUnsignedPixel()

```
bool FliSdk::isUnsignedPixel ( )
```

Return the pixel sign (int16 or uint16)

6.12.4.71 listAllGrabbers()

```
std::list<IFrameGrabber *> FliSdk::listAllGrabbers ( )
```

listAllGrabbers return the list of pointers of all the current grabbers

Returns

a list of grabbers pointers

6.12.4.72 loadBuffer() [1/3]

```
FliSdkError FliSdk::loadBuffer (
    const std::string & path,
    CroppingData & bufferCrop )
```

Load a buffer from a file, in the ringBuffer of the SDK.

Parameters

<i>path</i>	path to the file
<i>bufferCrop</i>	a ref to CroppingData to get current cropping

Returns

FliSdkError

6.12.4.73 loadBuffer() [2/3]

```
FliSdkError FliSdk::loadBuffer (
    const std::string & path,
    Fli::LoadBufferInfo & bufferInfo,
    bool inRingBuffer = false )
```

Load a buffer from a file, allocate memory, and return that memory to the user.

User must delete this memory if not used. For a raw file, user have to set width, height and isMono8 info in the LoadBufferInfo.

Parameters

<i>path</i>	path to the file
<i>bufferInfo</i>	struct with images and info
<i>inRingBuffer</i>	bool load buffer in ring buffer (in case of thermo) false by default

Returns

FliSdkError

6.12.4.74 loadBuffer() [3/3]

```
FliSdkError FliSdk::loadBuffer (
    const uint8_t * buffer,
    uint32_t nbImages,
    uint64_t imageSize = 0 )
```

Load a buffer in the ringBuffer of the SDK.

Parameters

<i>buffer</i>	data buffer
<i>nbImages</i>	nbImages in the buffer
<i>imageSize</i>	: the size of one image

Returns

FliSdkError

6.12.4.75 log()

```
void FliSdk::log (
    const std::string & text )
```

add text to log file

Parameters

<i>text</i>	: text to add
-------------	---------------

6.12.4.76 logOutside()

```
void FliSdk::logOutside (
    const std::string & text )
```

add text to log file from outside the SDK

Parameters

<i>text</i>	: text to add
-------------	---------------

6.12.4.77 observersNotifEnabled()

```
bool FliSdk::observersNotifEnabled ( )
```

6.12.4.78 ocam2k()

```
FliOcam2K* FliSdk::ocam2k ( )
```

Get Ocam2K camera interface.

Returns

a pointer to the camera object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.79 ocam2s()

```
FliOcam2S* FliSdk::ocam2s ( )
```

Get Ocam2S camera interface.

Returns

a pointer to the camera object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.80 openMatroxGenicamBrowser()

```
bool FliSdk::openMatroxGenicamBrowser ( )
```

Open CXP Matrox browser.

Returns

true if the browser is opened, false otherwise

6.12.4.81 operator=()

```
FliSdk& FliSdk::operator= (
    const FliSdk & ) [delete]
```

6.12.4.82 removeImageProcessing()

```
void FliSdk::removeImageProcessing (
    ProcessingId id )
```

remove an image processing

6.12.4.83 removeObserver()

```
void FliSdk::removeObserver (
    IFliSdkObserver * obs )
```

remove a [FliSdk](#) state observer

6.12.4.84 removeRawImageReceivedObserver()

```
void FliSdk::removeRawImageReceivedObserver (
    IRawImageReceivedObserver * obs )
```

Remove an observer.

Parameters

<i>obs</i>	pointer on the observer
------------	-------------------------

6.12.4.85 resetBuffer()

```
void FliSdk::resetBuffer ( )
```

Reset the buffer.

6.12.4.86 ringBuffer()

```
FliRingBuffer& FliSdk::ringBuffer ( )
```

Return the interface used to interact with the SDK ring buffer.

6.12.4.87 saveBuffer() [1/2]

```
FliSdkError FliSdk::saveBuffer (
    const std::string & path,
    const Fli::LoadBufferInfo & info,
    uint32_t start,
    uint32_t end,
    std::function< bool(int)> progressionCallback = nullptr,
    bool withMetadata = false,
    uint16_t offset = 0,
    bool forceUnsigned = false,
    uint16_t decimation = 1,
    ProcessingId id = -1 )
```

Save as above but with a LoadBufferInfo struct.

6.12.4.88 saveBuffer() [2/2]

```
FliSdkError FliSdk::saveBuffer (
    std::string path,
    uint32_t start,
    uint32_t end,
    std::function< bool(int)> progressionCallback = nullptr,
    bool withMetadata = false,
    uint16_t offset = 0,
    bool forceUnsigned = false,
    uint16_t decimation = 1 )
```

Save the buffer at path.

Parameters

<i>path</i>	: path of the file
<i>start</i>	: start index of the buffer
<i>end</i>	: end index of the buffer
<i>progressionCallback</i>	: a callback to notify the progression of the save, return false to stop the save
<i>withMetadata</i>	: true to include camera conf in metadata
<i>offset</i>	: apply an offset on all the pixels of all images
<i>forceUnsigned</i>	: force the save with unsigned pixels type
<i>decimation</i>	: apply a decimation on the index of saved images

Returns

FliSdkError

6.12.4.89 serialCamera()

```
FliSerialCamera* FliSdk::serialCamera ( )
```

Get common camera interface for C-RED and OCAM2 cameras.

Returns

a pointer to the [FliSerialCamera](#) object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.90 setBufferSize()

```
void FliSdk::setBufferSize (
    uint16_t sizeMo )
```

Change the buffer capacity in Mo.

Parameters

<i>sizeMo</i>	: capacity of the ring buffer in Mo
---------------	-------------------------------------

6.12.4.91 setBufferSizeInImages()

```
void FliSdk::setBufferSizeInImages (
    uint64_t nbImages )
```

Change the buffer capacity in number of images.

Parameters

<i>nblImages</i>	: capacity of the ring buffer in nb images
------------------	--

6.12.4.92 setBurstFilter()

```
void FliSdk::setBurstFilter (
    int16_t id )
```

Set the burst filter for id.

Parameters

<i>id</i>	: id to display
-----------	-----------------

6.12.4.93 setCamera()

```
bool FliSdk::setCamera (
    std::string cameraName )
```

Set the camera to be used.

Parameters

<i>cameraName</i>	name of the camera
-------------------	--------------------

Returns

true if camera exists else false

Attention

Call [update\(\)](#) to apply.

6.12.4.94 setCroppingState()

```
FliSdkError FliSdk::setCroppingState (
    bool enable,
    CroppingData croppingData )
```

Set the cropping data.

Parameters

<i>enable</i>	enable or disable cropping
<i>croppingData</i>	cropping data

Returns

FliSdkError

6.12.4.95 setFowlerOffset()

```
void FliSdk::setFowlerOffset (
    uint16_t offset )
```

Set the value of the Fowler offset to apply on sum of the images.

Parameters

<i>offset</i>	the value of the offset (between 0 and 65535), default is 0
---------------	---

6.12.4.96 setGrabber()

```
bool FliSdk::setGrabber (
    std::string grabberName )
```

Set the grabber to be used.

Parameters

<i>grabberName</i>	name of the grabber
--------------------	---------------------

Returns

true if grabber exists else false

Attention

Call [update\(\)](#) to apply.

6.12.4.97 setImageDimension()

```
void FliSdk::setImageDimension (
    uint16_t width,
    uint16_t height )
```

Force the image dimension apply to the grabber.

Parameters

<i>width</i>	width of image
<i>height</i>	height of image

6.12.4.98 setImageDimensionImageRingBuffer()

```
void FliSdk::setImageDimensionImageRingBuffer (
    uint16_t width,
    uint16_t height )
```

Force the image dimension apply to the image ring buffer needed when there is no grabber but images loaded from files.

Parameters

<i>width</i>	width of image
<i>height</i>	height of image

6.12.4.99 setImageDimensionImageRingBufferThermo()

```
void FliSdk::setImageDimensionImageRingBufferThermo (
    uint16_t width,
    uint16_t height )
```

Force the image dimension apply to the thermographic image ring buffer needed when there is no grabber but thermographic images loaded from files (*.thr.raw files)

Parameters

<i>width</i>	width of image
<i>height</i>	height of image

6.12.4.100 setMode()

```
void FliSdk::setMode (
    Mode mode )
```

Set the mode of use of the sdk.

Parameters

<i>mode</i>	mode used (full, grabOnly, configOnly)
-------------	--

Attention

Call [update\(\)](#) to apply.

6.12.4.101 setNbImagesPerBuffer()

```
void FliSdk::setNbImagesPerBuffer (
    uint8_t nbImages )
```

Set set number of images the grabber should acquire before trigger, use this function for high FPS.

Parameters

<i>nbImages</i>	: number of images.
-----------------	---------------------

6.12.4.102 setOcamFrameNumberOffset()

```
void FliSdk::setOcamFrameNumberOffset (
    uint8_t offset )
```

Set the offset for the frame number.

Parameters

<i>offset</i>	0 for simulator, 8 for camera
---------------	-------------------------------

6.12.4.103 setupFixedCCsFrameGrabber()

```
void FliSdk::setupFixedCCsFrameGrabber (
    int set )
```

Set the CC IO_SOURCE for one set (and the other 3 will be unset), available only for Matrox CL Grabber.

Parameters

<i>set</i>	a value between 1 to 4
------------	------------------------

6.12.4.104 sfncCamera()

```
FliSfncCamera* FliSdk::sfncCamera ( )
```

Get commond interface for genicam camera.

Returns

a pointer to the camera object (nullptr if name doesn't exist or if sdk not initialized)

6.12.4.105 start()

```
FliSdkError FliSdk::start ( )
```

Start the grabber (must be initialized before)

Returns

FliSdkError

6.12.4.106 stop()

```
FliSdkError FliSdk::stop ( )
```

Stop the grabber.

Returns

FliSdkError

6.12.4.107 update()

```
FliSdkError FliSdk::update ( )
```

Update the changes, must be call after setCamera, setGrabber or setMode to take effects.

Returns

FliSdkError

6.12.4.108 version()

```
std::string FliSdk::version ( )
```

Get the version of the sdk.

Returns

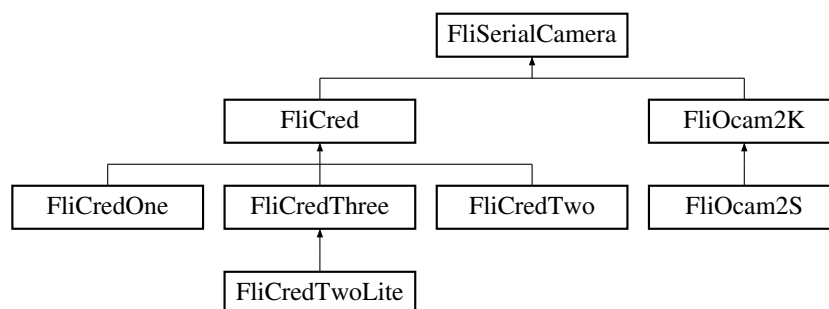
a string containing verison of sdk

6.13 FliSerialCamera Class Reference

This class is the base class of all serial camera (CameraLink). It implements common C-RED and Ocam functions.

```
#include <FliSerialCamera.h>
```

Inheritance diagram for FliSerialCamera:



Public Member Functions

- [FliSerialCamera](#) (IFrameGrabberCL *grabber)
- virtual [~FliSerialCamera](#) ()
- FliSdkError [getModel](#) (Fli::CameraModel &model)
- void [getCurrentImageDimension](#) (uint16_t &width, uint16_t &height)
- FliSdkError [getFps](#) (double &fps)
- FliSdkError [getFpsMax](#) (double &fps)
- FliSdkError [setFps](#) (double fps)
- FliSdkError [enableBias](#) (bool enable)
- FliSdkError [enableFlat](#) (bool enable)
- bool [sendCommand](#) (const std::string &command, std::string &response, int timeout=0, bool filtered=true, bool echo=false)
 - Send a command to the camera and get the response in one time.*
- FliSdkError [sendCommand](#) (const std::string &command, int timeout=0, std::function< void(std::string)> getStringStream=nullptr, bool echo=false)
 - Send a command to the camera and get the response with a callback called at each readSerial.*
- void [resynchronizeSerial](#) ()
 - Resynchronize the serial, can be done sometimes.*
- void [purgeSerial](#) (int timeout)
 - Resynchronize the serial, can be done sometimes.*
- void [addObserver](#) (IFliCameraObserver *obs)
- void [removeObserver](#) (IFliCameraObserver *obs)
- void [writeSerial](#) (const std::string &str)
- std::string [readSerial](#) ()
- void [setCustomSerial](#) (ICustomSerial *customSerial)
- void [sleep](#) (int ms)
- bool [isCameraConnected](#) ()

Protected Member Functions

- bool [isNumber](#) (const std::string &s)
- void [getRawData](#) (std::string &s)
- void [notifyObservers](#) (const std::string &command)
- void [grabberWriteSerial](#) (const std::string command, bool echo)
- void [grabberReadSerial](#) (std::string &buf, bool echo)

Protected Attributes

- IFrameGrabberCL * [_grabber](#)
- Fli::CameraModel [_cameraModel](#)
- std::list< IFliCameraObserver * > [_observers](#)
- bool [_croppingFromFunction](#)
- ICustomSerial * [_customSerial](#)
- bool [_needEcho](#)

Friends

- class [FliSdkImpl](#)
- class [FliSdkImplCL](#)
- class [FliCred](#)
- class [FliCredOne](#)
- class [FliCredTwo](#)
- class [FliCredTwoLite](#)
- class [FliCredThree](#)
- class [FliOcam2K](#)
- class [FliOcam2S](#)

6.13.1 Detailed Description

This class is the base class of all serial camera (CameraLink). It implements common C-RED and Ocam functions.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 FliSerialCamera()

```
FliSerialCamera::FliSerialCamera (
    IFrameGrabberCL * grabber )
```

6.13.2.2 ~FliSerialCamera()

```
virtual FliSerialCamera::~~FliSerialCamera ( ) [virtual]
```

6.13.3 Member Function Documentation

6.13.3.1 addObserver()

```
void FliSerialCamera::addObserver (
    IFliCameraObserver * obs ) [inline]
```

6.13.3.2 enableBias()

```
FliSdkError FliSerialCamera::enableBias (
    bool enable )
```

6.13.3.3 enableFlat()

```
FliSdkError FliSerialCamera::enableFlat (
    bool enable )
```

6.13.3.4 getCurrentImageDimension()

```
void FliSerialCamera::getCurrentImageDimension (
    uint16_t & width,
    uint16_t & height )
```

6.13.3.5 getFps()

```
FliSdkError FliSerialCamera::getFps (
    double & fps )
```

6.13.3.6 getFpsMax()

```
FliSdkError FliSerialCamera::getFpsMax (
    double & fps )
```

6.13.3.7 getModel()

```
FliSdkError FliSerialCamera::getModel (
    Fli::CameraModel & model )
```

6.13.3.8 getRawData()

```
void FliSerialCamera::getRawData (
    std::string & s ) [protected]
```

6.13.3.9 grabberReadSerial()

```
void FliSerialCamera::grabberReadSerial (
    std::string & buf,
    bool echo ) [protected]
```

6.13.3.10 grabberWriteSerial()

```
void FliSerialCamera::grabberWriteSerial (
    const std::string command,
    bool echo ) [protected]
```

6.13.3.11 isCameraConnected()

```
bool FliSerialCamera::isCameraConnected ( )
```

6.13.3.12 isNumber()

```
bool FliSerialCamera::isNumber (
    const std::string & s ) [protected]
```

6.13.3.13 notifyObservers()

```
void FliSerialCamera::notifyObservers (
    const std::string & command ) [protected]
```


6.13.3.14 purgeSerial()

```
void FliSerialCamera::purgeSerial (
    int timeout )
```

Resynchronize the serial, can be done sometimes.

6.13.3.15 readSerial()

```
std::string FliSerialCamera::readSerial ( )
```

6.13.3.16 removeObserver()

```
void FliSerialCamera::removeObserver (
    IFliCameraObserver * obs ) [inline]
```

6.13.3.17 resynchronizeSerial()

```
void FliSerialCamera::resynchronizeSerial ( )
```

Resynchronize the serial, can be done sometimes.

6.13.3.18 sendCommand() [1/2]

```
FliSdkError FliSerialCamera::sendCommand (
    const std::string & command,
    int timeout = 0,
    std::function< void(std::string)> getStringStream = nullptr,
    bool echo = false )
```

Send a command to the camera and get the response with a callback called at each readSerial.

Parameters

<i>command</i>	command to send
<i>timeout</i>	command timeout
<i>getStringStream</i>	callback called each time read serial is called

Returns

a `FliSdkError`

6.13.3.19 sendCommand() [2/2]

```
bool FliSerialCamera::sendCommand (
    const std::string & command,
    std::string & response,
    int timeout = 0,
    bool filtered = true,
    bool echo = false )
```

Send a command to the camera and get the response in one time.

Parameters

<i>command</i>	command to send
<i>response</i>	response of the camera
<i>timeout</i>	command timeout
<i>filtered</i>	if true all useless string in response will be deleted

Returns

true if command is ok else false

6.13.3.20 setCustomSerial()

```
void FliSerialCamera::setCustomSerial (
    ICustomSerial * customSerial )
```

6.13.3.21 setFps()

```
FliSdkError FliSerialCamera::setFps (
    double fps )
```

6.13.3.22 sleep()

```
void FliSerialCamera::sleep (
    int ms ) [inline]
```

6.13.3.23 writeSerial()

```
void FliSerialCamera::writeSerial (
    const std::string & str )
```

6.13.4 Friends And Related Function Documentation

6.13.4.1 FliCred

```
friend class FliCred [friend]
```

6.13.4.2 FliCredOne

```
friend class FliCredOne [friend]
```

6.13.4.3 FliCredThree

```
friend class FliCredThree [friend]
```

6.13.4.4 FliCredTwo

```
friend class FliCredTwo [friend]
```

6.13.4.5 FliCredTwoLite

```
friend class FliCredTwoLite [friend]
```

6.13.4.6 FliOcam2K

```
friend class FliOcam2K [friend]
```

6.13.4.7 FliOcam2S

```
friend class FliOcam2S [friend]
```

6.13.4.8 FliSdkImpl

```
friend class FliSdkImpl [friend]
```

6.13.4.9 FliSdkImplCL

```
friend class FliSdkImplCL [friend]
```

6.13.5 Member Data Documentation

6.13.5.1 _cameraModel

```
Fli::CameraModel FliSerialCamera::_cameraModel [protected]
```

6.13.5.2 _croppingFromFunction

```
bool FliSerialCamera::_croppingFromFunction [protected]
```

6.13.5.3 _customSerial

```
ICustomSerial* FliSerialCamera::_customSerial [protected]
```

6.13.5.4 _grabber

```
IFrameGrabberCL* FliSerialCamera::_grabber [protected]
```

6.13.5.5 `_needEcho`

```
bool FliSerialCamera::_needEcho [protected]
```

6.13.5.6 `_observers`

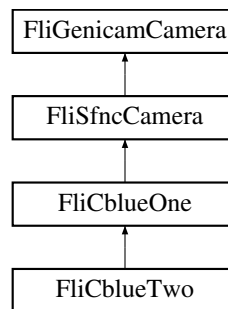
```
std::list<IFliCameraObserver*> FliSerialCamera::_observers [protected]
```

6.14 FliSfncCamera Class Reference

This class defined all the register of an SFNC compliant camera.

```
#include <FliSfncCamera.h>
```

Inheritance diagram for FliSfncCamera:



Public Member Functions

- [FliSfncCamera](#) (IFrameGrabberGenicam *grabber)
- virtual `~FliSfncCamera` ()

Public Attributes

- GenicamFeature< bool > * [CameraPresence](#)
- GenicamFeature< FliSfncCameraEnum::DeviceTypeEnum > * [DeviceType](#)
Returns the device type.
- GenicamFeature< FliSfncCameraEnum::DeviceScanTypeEnum > * [DeviceScanType](#)
Scan type of the sensor of the device.
- GenicamFeature< std::string > * [DeviceVendorName](#)
Name of the manufacturer of the device.
- GenicamFeature< std::string > * [DeviceModelName](#)
Model of the device.
- GenicamFeature< std::string > * [DeviceFamilyName](#)
Identifier of the product family of the device.
- GenicamFeature< std::string > * [DeviceManufacturerInfo](#)

- Manufacturer information about the device.*

 - GenicamFeature< std::string > * [DeviceVersion](#)
Version of the device.
 - GenicamFeature< std::string > * [DeviceFirmwareVersion](#)
Version of the firmware in the device.
 - GenicamFeature< std::string > * [DeviceSerialNumber](#)
Device's serial number. This string is a unique identifier of the device.
 - GenicamFeature< std::string > * [DeviceUserID](#)
User-programmable device identifier.
 - GenicamFeature< int64_t > * [DeviceSFNCVersionMajor](#)
Major version of the Standard Features Naming Convention that was used to create the device's GenICam XML.
 - GenicamFeature< int64_t > * [DeviceSFNCVersionMinor](#)
Minor version of the Standard Features Naming Convention that was used to create the device's GenICam XML.
 - GenicamFeature< int64_t > * [DeviceSFNCVersionSubMinor](#)
Sub minor version of Standard Features Naming Convention that was used to create the device's GenICam XML.
 - GenicamFeature< int64_t > * [DeviceManifestEntrySelector](#)
Selects the manifest entry to reference.
 - GenicamFeature< int64_t > * [DeviceManifestXMLMajorVersion](#)
Indicates the major version number of the GenICam XML file of the selected manifest entry.
 - GenicamFeature< int64_t > * [DeviceManifestXMLMinorVersion](#)
Indicates the minor version number of the GenICam XML file of the selected manifest entry.
 - GenicamFeature< int64_t > * [DeviceManifestXMLSubMinorVersion](#)
Indicates the subminor version number of the GenICam XML file of the selected manifest entry.
 - GenicamFeature< int64_t > * [DeviceManifestSchemaMajorVersion](#)
Indicates the major version number of the schema file of the selected manifest entry.
 - GenicamFeature< int64_t > * [DeviceManifestSchemaMinorVersion](#)
Indicates the minor version number of the schema file of the selected manifest entry.
 - GenicamFeature< std::string > * [DeviceManifestPrimaryURL](#)
Indicates the first URL to the GenICam XML device description file of the selected manifest entry.
 - GenicamFeature< std::string > * [DeviceManifestSecondaryURL](#)
Indicates the second URL to the GenICam XML device description file of the selected manifest entry.
 - GenicamFeature< FliSfncCameraEnum::DeviceTLTypeEnum > * [DeviceTLType](#)
Transport Layer type of the device.
 - GenicamFeature< int64_t > * [DeviceTLVersionMajor](#)
Major version of the Transport Layer of the device.
 - GenicamFeature< int64_t > * [DeviceTLVersionMinor](#)
Minor version of the Transport Layer of the device.
 - GenicamFeature< int64_t > * [DeviceTLVersionSubMinor](#)
Sub minor version of the Transport Layer of the device.
 - GenicamFeature< int64_t > * [DeviceGenCPVersionMajor](#)
Major version of the GenCP protocol supported by the device.
 - GenicamFeature< int64_t > * [DeviceGenCPVersionMinor](#)
Minor version of the GenCP protocol supported by the device.
 - GenicamFeature< int64_t > * [DeviceMaxThroughput](#)
Maximum bandwidth of the data that can be streamed out of the device. This can be used to estimate if the physical connection(s) can sustain transfer of free-running images from the camera at its maximum speed.
 - GenicamFeature< int64_t > * [DeviceConnectionSelector](#)
Selects which Connection of the device to control.
 - GenicamFeature< int64_t > * [DeviceConnectionSpeed](#)
Indicates the speed of transmission of the specified Connection.
 - GenicamFeature< FliSfncCameraEnum::DeviceConnectionStatusEnum > * [DeviceConnectionStatus](#)

- Indicates the status of the specified Connection.*

 - GenicamFeature< int64_t > * [DeviceLinkSelector](#)
Selects which Link of the device to control.
 - GenicamFeature< int64_t > * [DeviceLinkSpeed](#)
Indicates the speed of transmission negotiated on the specified Link.
 - GenicamFeature< FliSfncCameraEnum::DeviceLinkThroughputLimitModeEnum > * [DeviceLinkThroughputLimitMode](#)
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.
 - GenicamFeature< int64_t > * [DeviceLinkThroughputLimit](#)
Limits the maximum bandwidth of the data that will be streamed out by the device on the selected Link. If necessary, delays will be uniformly inserted between transport layer packets in order to control the peak bandwidth.
 - GenicamFeature< int64_t > * [DeviceLinkConnectionCount](#)
Returns the number of physical connection of the device used by a particular Link.
 - GenicamFeature< FliSfncCameraEnum::DeviceLinkHeartbeatModeEnum > * [DeviceLinkHeartbeatMode](#)
Activate or deactivate the Link's heartbeat.
 - GenicamFeature< double > * [DeviceLinkHeartbeatTimeout](#)
Controls the current heartbeat timeout of the specific Link.
 - GenicamFeature< double > * [DeviceLinkCommandTimeout](#)
Indicates the command timeout of the specified Link. This corresponds to the maximum response time of the device for a command sent on that link.
 - GenicamFeature< int64_t > * [DeviceStreamChannelCount](#)
Indicates the number of streaming channels supported by the device.
 - GenicamFeature< int64_t > * [DeviceStreamChannelSelector](#)
Selects the stream channel to control.
 - GenicamFeature< FliSfncCameraEnum::DeviceStreamChannelTypeEnum > * [DeviceStreamChannelType](#)
Reports the type of the stream channel.
 - GenicamFeature< int64_t > * [DeviceStreamChannelLink](#)
Index of device's Link to use for streaming the specified stream channel.
 - GenicamFeature< FliSfncCameraEnum::DeviceStreamChannelEndiannessEnum > * [DeviceStreamChannelEndianness](#)
Endianness of multi-byte pixel data for this stream.
 - GenicamFeature< int64_t > * [DeviceStreamChannelPacketSize](#)
Specifies the stream packet size, in bytes, to send on the selected channel for a Transmitter or specifies the maximum packet size supported by a receiver.
 - GenicamFeature< int64_t > * [DeviceEventChannelCount](#)
Indicates the number of event channels supported by the device.
 - GenicamFeature< FliSfncCameraEnum::DeviceCharacterSetEnum > * [DeviceCharacterSet](#)
Character set used by the strings of the device.
 - GenicamFeature * [DeviceReset](#)
Resets the device to its power up state. After reset, the device must be rediscovered.
 - GenicamFeature< FliSfncCameraEnum::DeviceIndicatorModeEnum > * [DeviceIndicatorMode](#)
Controls the behavior of the indicators (such as LEDs) showing the status of the Device.
 - GenicamFeature * [DeviceFeaturePersistenceStart](#)
Indicate to the device and GenICam XML to get ready for persisting of all streamable features.
 - GenicamFeature * [DeviceFeaturePersistenceEnd](#)
Indicate to the device the end of feature persistence.
 - GenicamFeature * [DeviceRegistersStreamingStart](#)
Prepare the device for registers streaming without checking for consistency.
 - GenicamFeature * [DeviceRegistersStreamingEnd](#)
Announce the end of registers streaming. This will do a register set validation for consistency and activate it. This will also update the DeviceRegistersValid flag.
 - GenicamFeature * [DeviceRegistersCheck](#)
Perform the validation of the current register set for consistency. This will update the DeviceRegistersValid flag.

- GenicamFeature< bool > * [DeviceRegistersValid](#)
Returns if the current register set is valid and consistent.
- GenicamFeature< FliSfncCameraEnum::DeviceRegistersEndiannessEnum > * [DeviceRegistersEndianness](#)
Endianness of the registers of the device.
- GenicamFeature< FliSfncCameraEnum::DeviceTemperatureSelectorEnum > * [DeviceTemperatureSelector](#)
Selects the location within the device, where the temperature will be measured.
- GenicamFeature< double > * [DeviceTemperature](#)
Device temperature in degrees Celsius (C). It is measured at the location selected by DeviceTemperatureSelector.
- GenicamFeature< FliSfncCameraEnum::DeviceClockSelectorEnum > * [DeviceClockSelector](#)
Selects the clock frequency to access from the device.
- GenicamFeature< double > * [DeviceClockFrequency](#)
Returns the frequency of the selected Clock.
- GenicamFeature< FliSfncCameraEnum::DeviceSerialPortSelectorEnum > * [DeviceSerialPortSelector](#)
Selects which serial port of the device to control.
- GenicamFeature< FliSfncCameraEnum::DeviceSerialPortBaudRateEnum > * [DeviceSerialPortBaudRate](#)
This feature controls the baud rate used by the selected serial port.
- GenicamFeature< int64_t > * [Timestamp](#)
Reports the current value of the device timestamp counter.
- GenicamFeature * [TimestampReset](#)
Resets the current value of the device timestamp counter.
- GenicamFeature * [TimestampLatch](#)
Latches the current timestamp counter into TimestampLatchValue.
- GenicamFeature< int64_t > * [TimestampLatchValue](#)
Returns the latched value of the timestamp counter.
- GenicamFeature< std::string > * [UserSetDescription](#)
Description of the selected User Set content.
- GenicamFeature< int64_t > * [SensorWidth](#)
Effective width of the sensor in pixels.
- GenicamFeature< int64_t > * [SensorHeight](#)
Effective height of the sensor in pixels.
- GenicamFeature< double > * [SensorPixelWidth](#)
Physical size (pitch) in the x direction of a photo sensitive pixel unit.
- GenicamFeature< double > * [SensorPixelHeight](#)
Physical size (pitch) in the y direction of a photo sensitive pixel unit.
- GenicamFeature< std::string > * [SensorName](#)
Product name of the imaging Sensor.
- GenicamFeature< FliSfncCameraEnum::SensorShutterModeEnum > * [SensorShutterMode](#)
Specifies the shutter mode of the device.
- GenicamFeature< FliSfncCameraEnum::SensorTapsEnum > * [SensorTaps](#)
Number of taps of the camera sensor.
- GenicamFeature< FliSfncCameraEnum::SensorDigitizationTapsEnum > * [SensorDigitizationTaps](#)
Number of digitized samples outputted simultaneously by the camera A/D conversion stage.
- GenicamFeature< int64_t > * [WidthMax](#)
Maximum width of the image (in pixels). The dimension is calculated after horizontal binning, decimation or any other function changing the horizontal dimension of the image.
- GenicamFeature< int64_t > * [HeightMax](#)
Maximum height of the image (in pixels). This dimension is calculated after vertical binning, decimation or any other function changing the vertical dimension of the image.
- GenicamFeature< FliSfncCameraEnum::RegionSelectorEnum > * [RegionSelector](#)
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.

- GenicamFeature< FliSfncCameraEnum::RegionModeEnum > * [RegionMode](#)
Controls if the selected Region of interest is active and streaming.
- GenicamFeature< FliSfncCameraEnum::RegionDestinationEnum > * [RegionDestination](#)
Control the destination of the selected region.
- GenicamFeature< int64_t > * [RegionIDValue](#)
Returns a unique Identifier value that corresponds to the selected Region.
- GenicamFeature< FliSfncCameraEnum::ComponentSelectorEnum > * [ComponentSelector](#)
Selects a component to activate/deactivate its data streaming.
- GenicamFeature< bool > * [ComponentEnable](#)
Controls if the selected component streaming is active.
- GenicamFeature< int64_t > * [ComponentIDValue](#)
Returns a unique Identifier value that corresponds to type of the component selected by ComponentSelector.
- GenicamFeature< FliSfncCameraEnum::GroupSelectorEnum > * [GroupSelector](#)
Selects a Group of component to control or inquire. The GroupSelector determines which components Group will be used for the selected features.
- GenicamFeature< int64_t > * [GroupIDValue](#)
Returns a unique Identifier value corresponding to the selected Group of Components. If no grouping is required, this value should be set to 0.
- GenicamFeature< int64_t > * [Width](#)
Width of the image provided by the device (in pixels).
- GenicamFeature< int64_t > * [Height](#)
Height of the image provided by the device (in pixels).
- GenicamFeature< int64_t > * [OffsetX](#)
Horizontal offset from the origin to the region of interest (in pixels).
- GenicamFeature< int64_t > * [OffsetY](#)
Vertical offset from the origin to the region of interest (in pixels).
- GenicamFeature< bool > * [LinePitchEnable](#)
This feature controls whether the LinePitch feature is writable. Otherwise LinePitch is implicitly controlled by the combination of features like Width, PixelFormat, etc...
- GenicamFeature< int64_t > * [LinePitch](#)
Total number of bytes between the starts of 2 consecutive lines. This feature is used to facilitate alignment of image data.
- GenicamFeature< FliSfncCameraEnum::BinningSelectorEnum > * [BinningSelector](#)
Selects which binning engine is controlled by the BinningHorizontal and BinningVertical features.
- GenicamFeature< FliSfncCameraEnum::BinningHorizontalModeEnum > * [BinningHorizontalMode](#)
Sets the mode to use to combine horizontal photo-sensitive cells together when BinningHorizontal is used.
- GenicamFeature< int64_t > * [BinningHorizontal](#)
Number of horizontal photo-sensitive cells to combine together. This reduces the horizontal resolution (width) of the image.
- GenicamFeature< FliSfncCameraEnum::BinningVerticalModeEnum > * [BinningVerticalMode](#)
Sets the mode to use to combine vertical photo-sensitive cells together when BinningVertical is used.
- GenicamFeature< int64_t > * [BinningVertical](#)
Number of vertical photo-sensitive cells to combine together. This reduces the vertical resolution (height) of the image.
- GenicamFeature< FliSfncCameraEnum::DecimationHorizontalModeEnum > * [DecimationHorizontalMode](#)
Sets the mode used to reduce the horizontal resolution when DecimationHorizontal is used.
- GenicamFeature< int64_t > * [DecimationHorizontal](#)
Horizontal sub-sampling of the image. This reduces the horizontal resolution (width) of the image by the specified horizontal decimation factor.
- GenicamFeature< FliSfncCameraEnum::DecimationVerticalModeEnum > * [DecimationVerticalMode](#)
Sets the mode used to reduce the Vertical resolution when DecimationVertical is used.
- GenicamFeature< int64_t > * [DecimationVertical](#)

Vertical sub-sampling of the image. This reduces the vertical resolution (height) of the image by the specified vertical decimation factor.

- GenicamFeature< bool > * [ReverseX](#)

Flip horizontally the image sent by the device. The Region of interest is applied after the flipping.

- GenicamFeature< bool > * [ReverseY](#)

Flip vertically the image sent by the device. The Region of interest is applied after the flipping.

- GenicamFeature< FliSfncCameraEnum::PixelFormatEnum > * [PixelFormat](#)

Format of the pixels provided by the device. It represents all the information provided by PixelSize, PixelColorFilter combined in a single feature.

- GenicamFeature< FliSfncCameraEnum::PixelFormatInfoSelectorEnum > * [PixelFormatInfoSelector](#)

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

- GenicamFeature< int64_t > * [PixelFormatInfoID](#)

Returns the value used by the streaming channels to identify the selected pixel format.

- GenicamFeature< FliSfncCameraEnum::PixelSizeEnum > * [PixelSize](#)

Total size in bits of a pixel of the image.

- GenicamFeature< FliSfncCameraEnum::PixelColorFilterEnum > * [PixelColorFilter](#)

Type of color filter that is applied to the image.

- GenicamFeature< int64_t > * [PixelDynamicRangeMin](#)

Minimum value that can be returned during the digitization process. This corresponds to the darkest value of the camera. For color camera, this returns the smallest value that each color component can take.

- GenicamFeature< int64_t > * [PixelDynamicRangeMax](#)

Maximum value that will be returned during the digitization process. This corresponds to the brightest value of the camera. For color camera, this returns the biggest value that each color component can take.

- GenicamFeature< FliSfncCameraEnum::TestPatternGeneratorSelectorEnum > * [TestPatternGeneratorSelector](#)

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

- GenicamFeature< FliSfncCameraEnum::TestPatternEnum > * [TestPattern](#)

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

- GenicamFeature< FliSfncCameraEnum::DeinterlacingEnum > * [Deinterlacing](#)

Controls how the device performs de-interlacing.

- GenicamFeature< FliSfncCameraEnum::ImageCompressionModeEnum > * [ImageCompressionMode](#)

Enable a specific image compression mode as the base mode for image transfer. Optionally, chunk data can be appended to the compressed image (See the [REF_Ref397502619](#) \h chapter).

- GenicamFeature< FliSfncCameraEnum::ImageCompressionRateOptionEnum > * [ImageCompressionRateOption](#)

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

- GenicamFeature< int64_t > * [ImageCompressionQuality](#)

Control the quality of the produced compressed stream.

- GenicamFeature< double > * [ImageCompressionBitrate](#)

Control the rate of the produced compressed stream.

- GenicamFeature< FliSfncCameraEnum::ImageCompressionJPEGFormatOptionEnum > * [ImageCompressionJPEGFormatOption](#)

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

- GenicamFeature< FliSfncCameraEnum::AcquisitionModeEnum > * [AcquisitionMode](#)

Sets the acquisition mode of the device. It defines mainly the number of frames to capture during an acquisition and the way the acquisition stops.

- GenicamFeature * [AcquisitionStart](#)

Starts the Acquisition of the device. The number of frames captured is specified by AcquisitionMode.

- GenicamFeature * [AcquisitionStop](#)

Stops the Acquisition of the device at the end of the current Frame. It is mainly used when AcquisitionMode is Continuous but can be used in any acquisition mode.

- GenicamFeature< FliSfncCameraEnum::AcquisitionStopModeEnum > * [AcquisitionStopMode](#)

Controls how the `AcquisitionStop` command and the acquisition stopped using a trigger (e.g. `AcquisitionActive`, `FrameBurstActive`, `FrameActive` or `FrameEnd` trigger), ends an ongoing frame. This feature is mainly used in Linescan devices where each line in a frame is acquired sequentially.

- GenicamFeature * [AcquisitionAbort](#)

Aborts the Acquisition immediately. This will end the capture without completing the current Frame or waiting on a trigger. If no Acquisition is in progress, the command is ignored.
- GenicamFeature * [AcquisitionArm](#)

Arms the device before an `AcquisitionStart` command. This optional command validates all the current features for consistency and prepares the device for a fast start of the Acquisition.
- GenicamFeature< int64_t > * [AcquisitionFrameCount](#)

Number of frames to acquire in MultiFrame Acquisition mode.
- GenicamFeature< int64_t > * [AcquisitionBurstFrameCount](#)

Number of frames to acquire for each `FrameBurstStart` trigger.
- GenicamFeature< double > * [AcquisitionFrameRate](#)

Controls the acquisition rate (in Hertz) at which the frames are captured.
- GenicamFeature< bool > * [AcquisitionFrameRateEnable](#)

Controls if the `AcquisitionFrameRate` feature is writable and used to control the acquisition rate. Otherwise, the acquisition rate is implicitly controlled by the combination of other features like `ExposureTime`, etc...
- GenicamFeature< double > * [AcquisitionLineRate](#)

Controls the rate (in Hertz) at which the Lines in a Frame are captured.
- GenicamFeature< bool > * [AcquisitionLineRateEnable](#)

Controls if the `AcquisitionLineRate` feature is writable and used to control the acquisition rate. Otherwise, the acquisition rate is implicitly controlled by the combination of other features like `ExposureTime`, etc...
- GenicamFeature< FliSfncCameraEnum::AcquisitionStatusSelectorEnum > * [AcquisitionStatusSelector](#)

Selects the internal acquisition signal to read using `AcquisitionStatus`.
- GenicamFeature< bool > * [AcquisitionStatus](#)

Reads the state of the internal acquisition signal selected using `AcquisitionStatusSelector`.
- GenicamFeature< FliSfncCameraEnum::TriggerSelectorEnum > * [TriggerSelector](#)

Selects the type of trigger to configure.
- GenicamFeature< FliSfncCameraEnum::TriggerModeEnum > * [TriggerMode](#)

Controls if the selected trigger is active.
- GenicamFeature * [TriggerSoftware](#)

Generates an internal trigger. `TriggerSource` must be set to `Software`.
- GenicamFeature< FliSfncCameraEnum::TriggerSourceEnum > * [TriggerSource](#)

Specifies the internal signal or physical input Line to use as the trigger source. The selected trigger must have its `TriggerMode` set to `On`.
- GenicamFeature< FliSfncCameraEnum::TriggerActivationEnum > * [TriggerActivation](#)

Specifies the activation mode of the trigger.
- GenicamFeature< FliSfncCameraEnum::TriggerOverlapEnum > * [TriggerOverlap](#)

Specifies the type trigger overlap permitted with the previous frame or line. This defines when a valid trigger will be accepted (or latched) for a new frame or a new line.
- GenicamFeature< double > * [TriggerDelay](#)

Specifies the delay in microseconds (us) to apply after the trigger reception before activating it.
- GenicamFeature< int64_t > * [TriggerDivider](#)

Specifies a division factor for the incoming trigger pulses.
- GenicamFeature< int64_t > * [TriggerMultiplier](#)

Specifies a multiplication factor for the incoming trigger pulses. It is generally used in conjunction with `TriggerDivider` to control the ratio of triggers that are accepted.
- GenicamFeature< FliSfncCameraEnum::ExposureModeEnum > * [ExposureMode](#)

Sets the operation mode of the Exposure.
- GenicamFeature< FliSfncCameraEnum::ExposureTimeModeEnum > * [ExposureTimeMode](#)

Sets the configuration mode of the `ExposureTime` feature.
- GenicamFeature< FliSfncCameraEnum::ExposureTimeSelectorEnum > * [ExposureTimeSelector](#)

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

- GenicamFeature< double > * [ExposureTime](#)
Sets the Exposure time when ExposureMode is Timed and ExposureAuto is Off. This controls the duration where the photosensitive cells are exposed to light.
- GenicamFeature< FliSfncCameraEnum::ExposureAutoEnum > * [ExposureAuto](#)
Sets the automatic exposure mode when ExposureMode is Timed. The exact algorithm used to implement this control is device-specific.
- GenicamFeature< FliSfncCameraEnum::MultiSlopeModeEnum > * [MultiSlopeMode](#)
Controls multi-slope exposure state.
- GenicamFeature< int64_t > * [MultiSlopeKneePointCount](#)
The number of knee-points as well as the number of additional exposure slopes used for multi-slope exposure.
- GenicamFeature< int64_t > * [MultiSlopeKneePointSelector](#)
Selects the parameters for controlling an additional slope in multi-slope exposure.
- GenicamFeature< double > * [MultiSlopeExposureLimit](#)
Percent of the ExposureTime at a certain knee-point of multi-slope exposure.
- GenicamFeature< double > * [MultiSlopeSaturationThreshold](#)
The percentage of the full saturation that is applied at a certain knee-point of a multi-slope exposure.
- GenicamFeature< double > * [MultiSlopeIntensityLimit](#)
The relative intensity which divides intensities influenced by different exposure slopes.
- GenicamFeature< double > * [MultiSlopeExposureGradient](#)
The gradient of the additional slope that is defined by this knee-point.
- GenicamFeature< bool > * [CxpFirstLineTriggerWithFrameStart](#)
Specifies if a FrameStart trigger also triggers the first LineStart at the same time.
- GenicamFeature< FliSfncCameraEnum::GainSelectorEnum > * [GainSelector](#)
Selects which Gain is controlled by the various Gain features.
- GenicamFeature< double > * [Gain](#)
Controls the selected gain as an absolute physical value. This is an amplification factor applied to the video signal.
- GenicamFeature< FliSfncCameraEnum::GainAutoEnum > * [GainAuto](#)
Sets the automatic gain control (AGC) mode. The exact algorithm used to implement AGC is device-specific.
- GenicamFeature< FliSfncCameraEnum::GainAutoBalanceEnum > * [GainAutoBalance](#)
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.
- GenicamFeature< FliSfncCameraEnum::BlackLevelSelectorEnum > * [BlackLevelSelector](#)
Selects which Black Level is controlled by the various Black Level features.
- GenicamFeature< double > * [BlackLevel](#)
Controls the analog black level as an absolute physical value. This represents a DC offset applied to the video signal.
- GenicamFeature< FliSfncCameraEnum::BlackLevelAutoEnum > * [BlackLevelAuto](#)
Controls the mode for automatic black level adjustment. The exact algorithm used to implement this adjustment is device-specific.
- GenicamFeature< FliSfncCameraEnum::BlackLevelAutoBalanceEnum > * [BlackLevelAutoBalance](#)
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.
- GenicamFeature< FliSfncCameraEnum::WhiteClipSelectorEnum > * [WhiteClipSelector](#)
Selects which White Clip to control.
- GenicamFeature< double > * [WhiteClip](#)
Controls the maximal intensity taken by the video signal before being clipped as an absolute physical value. The video signal will never exceed the white clipping point: it will saturate at that level.
- GenicamFeature< FliSfncCameraEnum::BalanceRatioSelectorEnum > * [BalanceRatioSelector](#)
Selects which Balance ratio to control.
- GenicamFeature< double > * [BalanceRatio](#)
Controls ratio of the selected color component to a reference color component. It is used for white balancing.

- GenicamFeature< FliSfncCameraEnum::BalanceWhiteAutoEnum > * [BalanceWhiteAuto](#)
Controls the mode for automatic white balancing between the color channels. The white balancing ratios are automatically adjusted.
- GenicamFeature< double > * [Gamma](#)
Controls the gamma correction of pixel intensity. This is typically used to compensate for non-linearity of the display system (such as CRT).
- GenicamFeature< FliSfncCameraEnum::LUTSelectorEnum > * [LUTSelector](#)
Selects which LUT to control.
- GenicamFeature< bool > * [LUTEnable](#)
Activates the selected LUT.
- GenicamFeature< int64_t > * [LUTIndex](#)
Control the index (offset) of the coefficient to access in the selected LUT.
- GenicamFeature< int64_t > * [LUTValue](#)
Returns the Value at entry LUTIndex of the LUT selected by LUTSelector.
- GenicamFeature< FliSfncCameraEnum::ColorTransformationSelectorEnum > * [ColorTransformationSelector](#)
Selects which Color Transformation module is controlled by the various Color Transformation features.
- GenicamFeature< bool > * [ColorTransformationEnable](#)
Activates the selected Color Transformation module.
- GenicamFeature< FliSfncCameraEnum::ColorTransformationValueSelectorEnum > * [ColorTransformationValueSelector](#)
Selects the Gain factor or Offset of the Transformation matrix to access in the selected Color Transformation module.
- GenicamFeature< double > * [ColorTransformationValue](#)
Represents the value of the selected Gain factor or Offset inside the Transformation matrix.
- GenicamFeature< FliSfncCameraEnum::LineSelectorEnum > * [LineSelector](#)
Selects the physical line (or pin) of the external device connector or the virtual line of the Transport Layer to configure.
- GenicamFeature< FliSfncCameraEnum::LineModeEnum > * [LineMode](#)
Controls if the physical Line is used to Input or Output a signal.
- GenicamFeature< bool > * [LineInverter](#)
Controls the inversion of the signal of the selected input or output Line.
- GenicamFeature< bool > * [LineStatus](#)
Returns the current status of the selected input or output Line.
- GenicamFeature< int64_t > * [LineStatusAll](#)
Returns the current status of all available Line signals at time of polling in a single bitfield.
- GenicamFeature< FliSfncCameraEnum::LineSourceEnum > * [LineSource](#)
Selects which internal acquisition or I/O source signal to output on the selected Line. LineMode must be Output.
- GenicamFeature< FliSfncCameraEnum::LineFormatEnum > * [LineFormat](#)
Controls the current electrical format of the selected physical input or output Line.
- GenicamFeature< FliSfncCameraEnum::UserOutputSelectorEnum > * [UserOutputSelector](#)
Selects which bit of the User Output register will be set by UserOutputValue.
- GenicamFeature< bool > * [UserOutputValue](#)
Sets the value of the bit selected by UserOutputSelector.
- GenicamFeature< int64_t > * [UserOutputValueAll](#)
Sets the value of all the bits of the User Output register. It is subject to the UserOutputValueAllMask.
- GenicamFeature< int64_t > * [UserOutputValueAllMask](#)
Sets the write mask to apply to the value specified by UserOutputValueAll before writing it in the User Output register. If the UserOutputValueAllMask feature is present, setting the user Output register using UserOutputValueAll will only change the bits that have a corresponding bit in the mask set to one.
- GenicamFeature< FliSfncCameraEnum::CounterSelectorEnum > * [CounterSelector](#)
Selects which Counter to configure.
- GenicamFeature< FliSfncCameraEnum::CounterEventSourceEnum > * [CounterEventSource](#)
Select the events that will be the source to increment the Counter.
- GenicamFeature< FliSfncCameraEnum::CounterEventActivationEnum > * [CounterEventActivation](#)

- Selects the Activation mode Event Source signal.*

 - GenicamFeature< FliSfncCameraEnum::CounterResetSourceEnum > * [CounterResetSource](#)

Selects the signals that will be the source to reset the Counter.
- GenicamFeature< FliSfncCameraEnum::CounterResetActivationEnum > * [CounterResetActivation](#)

Selects the Activation mode of the Counter Reset Source signal.
- GenicamFeature * [CounterReset](#)

Does a software reset of the selected Counter and starts it. The counter starts counting events immediately after the reset unless a Counter trigger is active. CounterReset can be used to reset the Counter independently from the CounterResetSource. To disable the counter temporarily, set CounterEventSource to Off.
- GenicamFeature< int64_t > * [CounterValue](#)

Reads or writes the current value of the selected Counter.
- GenicamFeature< int64_t > * [CounterValueAtReset](#)

Reads the value of the selected Counter when it was reset by a trigger or by an explicit CounterReset command.
- GenicamFeature< int64_t > * [CounterDuration](#)

Sets the duration (or number of events) before the CounterEnd event is generated.
- GenicamFeature< FliSfncCameraEnum::CounterStatusEnum > * [CounterStatus](#)

Returns the current status of the Counter.
- GenicamFeature< FliSfncCameraEnum::CounterTriggerSourceEnum > * [CounterTriggerSource](#)

Selects the source to start the Counter.
- GenicamFeature< FliSfncCameraEnum::CounterTriggerActivationEnum > * [CounterTriggerActivation](#)

Selects the activation mode of the trigger to start the Counter.
- GenicamFeature< FliSfncCameraEnum::TimerSelectorEnum > * [TimerSelector](#)

Selects which Timer to configure.
- GenicamFeature< double > * [TimerDuration](#)

Sets the duration (in microseconds) of the Timer pulse.
- GenicamFeature< double > * [TimerDelay](#)

Sets the duration (in microseconds) of the delay to apply at the reception of a trigger before starting the Timer.
- GenicamFeature * [TimerReset](#)

Does a software reset of the selected timer and starts it. The timer starts immediately after the reset unless a timer trigger is active.
- GenicamFeature< double > * [TimerValue](#)

Reads or writes the current value (in microseconds) of the selected Timer.
- GenicamFeature< FliSfncCameraEnum::TimerStatusEnum > * [TimerStatus](#)

Returns the current status of the Timer.
- GenicamFeature< FliSfncCameraEnum::TimerTriggerSourceEnum > * [TimerTriggerSource](#)

Selects the source of the trigger to start the Timer.
- GenicamFeature< FliSfncCameraEnum::TimerTriggerActivationEnum > * [TimerTriggerActivation](#)

Selects the activation mode of the trigger to start the Timer.
- GenicamFeature< double > * [TimerTriggerArmDelay](#)

Sets the minimum period between two valid timer triggers.
- GenicamFeature< FliSfncCameraEnum::EncoderSelectorEnum > * [EncoderSelector](#)

Selects which Encoder to configure.
- GenicamFeature< FliSfncCameraEnum::EncoderSourceAEnum > * [EncoderSourceA](#)

Selects the signal which will be the source of the A input of the Encoder.
- GenicamFeature< FliSfncCameraEnum::EncoderSourceBEnum > * [EncoderSourceB](#)

Selects the signal which will be the source of the B input of the Encoder.
- GenicamFeature< FliSfncCameraEnum::EncoderModeEnum > * [EncoderMode](#)

Selects if the count of encoder uses FourPhase mode with jitter filtering or the HighResolution mode without jitter filtering.
- GenicamFeature< int64_t > * [EncoderDivider](#)

Sets how many Encoder increments/decrements are needed to generate an Encoder output pulse signal.

- GenicamFeature< FliSfncCameraEnum::EncoderOutputModeEnum > * [EncoderOutputMode](#)
Selects the conditions for the Encoder interface to generate a valid Encoder output signal.
- GenicamFeature< FliSfncCameraEnum::EncoderStatusEnum > * [EncoderStatus](#)
Returns the motion status of the encoder.
- GenicamFeature< double > * [EncoderTimeout](#)
Sets the maximum time interval between encoder counter increments before the status turns to static.
- GenicamFeature< FliSfncCameraEnum::EncoderResetSourceEnum > * [EncoderResetSource](#)
Selects the signals that will be the source to reset the Encoder.
- GenicamFeature< FliSfncCameraEnum::EncoderResetActivationEnum > * [EncoderResetActivation](#)
Selects the Activation mode of the Encoder Reset Source signal.
- GenicamFeature * [EncoderReset](#)
Does a software reset of the selected Encoder and starts it. The Encoder starts counting events immediately after the reset. EncoderReset can be used to reset the Encoder independently from the EncoderResetSource.
- GenicamFeature< int64_t > * [EncoderValue](#)
Reads or writes the current value of the position counter of the selected Encoder.
- GenicamFeature< int64_t > * [EncoderValueAtReset](#)
Reads the value of the of the position counter of the selected Encoder when it was reset by a signal or by an explicit EncoderReset command.
- GenicamFeature< double > * [EncoderResolution](#)
Defines the resolution of one encoder step.
- GenicamFeature< FliSfncCameraEnum::LogicBlockSelectorEnum > * [LogicBlockSelector](#)
Specifies the Logic Block to configure.
- GenicamFeature< FliSfncCameraEnum::LogicBlockFunctionEnum > * [LogicBlockFunction](#)
Selects the combinational logic Function of the Logic Block to configure.
- GenicamFeature< int64_t > * [LogicBlockInputNumber](#)
Specifies the number of active signal inputs of the Logic Block.
- GenicamFeature< int64_t > * [LogicBlockInputSelector](#)
Selects the Logic Block's input to configure.
- GenicamFeature< FliSfncCameraEnum::LogicBlockInputSourceEnum > * [LogicBlockInputSource](#)
Selects the source signal for the input into the Logic Block. True or False indicates the input is forced constant.
- GenicamFeature< bool > * [LogicBlockInputInverter](#)
Selects if the selected Logic Block Input source signal is inverted. This feature is not available when the LogicBlock↔InputSource is set to True or False.
- GenicamFeature< int64_t > * [LogicBlockLUTIndex](#)
Controls the index of the truth table to access in the selected LUT.
- GenicamFeature< bool > * [LogicBlockLUTValue](#)
Read or Write the Value associated with the entry at index LogicBlockLUTIndex of the selected LUT.
- GenicamFeature< int64_t > * [LogicBlockLUTValueAll](#)
Sets the values of all the output bits of the selected LUT in one access ignoring LogicBlockLUTIndex. LogicBlockL↔UTValueAll value can be any binary number and each bit correspond to the output value for the corresponding index (i.e. Bit 0 = LUT Index 0 output binary value).
- GenicamFeature< FliSfncCameraEnum::LogicBlockLUTSelectorEnum > * [LogicBlockLUTSelector](#)
Selects which of the two LUTs to configure when the selected Logic Block is a Latched dual LUTs (i.e.: Logical↔BlockFunction = LatchedLUT).
- GenicamFeature< FliSfncCameraEnum::SoftwareSignalSelectorEnum > * [SoftwareSignalSelector](#)
Selects which Software Signal features to control.
- GenicamFeature * [SoftwareSignalPulse](#)
Generates a pulse signal that can be used as a software trigger. This command can be used to trigger other modules that accept a SoftwareSignal as trigger source.
- GenicamFeature< FliSfncCameraEnum::ActionUnconditionalModeEnum > * [ActionUnconditionalMode](#)
Enables the unconditional action command mode where action commands are processed even when the primary control channel is closed.

- GenicamFeature< int64_t > * [ActionDeviceKey](#)
Provides the device key that allows the device to check the validity of action commands. The device internal assertion of an action signal is only authorized if the ActionDeviceKey and the action device key value in the protocol message are equal.
- GenicamFeature< int64_t > * [ActionQueueSize](#)
Indicates the size of the scheduled action commands queue. This number represents the maximum number of scheduled action commands that can be pending at a given point in time.
- GenicamFeature< int64_t > * [ActionSelector](#)
Selects to which Action Signal further Action settings apply.
- GenicamFeature< int64_t > * [ActionGroupMask](#)
Provides the mask that the device will use to validate the action on reception of the action protocol message.
- GenicamFeature< int64_t > * [ActionGroupKey](#)
Provides the key that the device will use to validate the action on reception of the action protocol message.
- GenicamFeature< FliSfncCameraEnum::EventSelectorEnum > * [EventSelector](#)
Selects which Event to signal to the host application.
- GenicamFeature< FliSfncCameraEnum::EventNotificationEnum > * [EventNotification](#)
Activate or deactivate the notification to the host application of the occurrence of the selected Event.
- GenicamFeature< int64_t > * [EventAcquisitionTrigger](#)
Returns the unique Identifier of the Acquisition Trigger type of Event.
- GenicamFeature< int64_t > * [EventAcquisitionTriggerTimestamp](#)
Returns the Timestamp of the Acquisition Trigger Event.
- GenicamFeature< int64_t > * [EventAcquisitionTriggerFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Acquisition Trigger Event.
- GenicamFeature< int64_t > * [EventAcquisitionTriggerMissed](#)
Returns the unique Identifier of the Acquisition Trigger Missed type of Event.
- GenicamFeature< int64_t > * [EventAcquisitionTriggerMissedTimestamp](#)
Returns the Timestamp of the Acquisition Trigger Missed Event.
- GenicamFeature< int64_t > * [EventAcquisitionTriggerMissedFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Acquisition Trigger Missed Event.
- GenicamFeature< int64_t > * [EventAcquisitionStart](#)
Returns the unique Identifier of the Acquisition Start type of Event.
- GenicamFeature< int64_t > * [EventAcquisitionStartTimestamp](#)
Returns the Timestamp of the Acquisition Start Event.
- GenicamFeature< int64_t > * [EventAcquisitionStartFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Acquisition Start Event.
- GenicamFeature< int64_t > * [EventAcquisitionEnd](#)
Returns the unique Identifier of the Acquisition End type of Event.
- GenicamFeature< int64_t > * [EventAcquisitionEndTimestamp](#)
Returns the Timestamp of the Acquisition End Event.
- GenicamFeature< int64_t > * [EventAcquisitionEndFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Acquisition End Event.
- GenicamFeature< int64_t > * [EventAcquisitionTransferStart](#)
Returns the unique Identifier of the Acquisition Transfer Start type of Event.
- GenicamFeature< int64_t > * [EventAcquisitionTransferStartTimestamp](#)
Returns the Timestamp of the Acquisition Transfer Start Event.
- GenicamFeature< int64_t > * [EventAcquisitionTransferStartFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Acquisition Transfer Start Event.
- GenicamFeature< int64_t > * [EventAcquisitionTransferEnd](#)
Returns the unique Identifier of the Acquisition Transfer End type of Event.
- GenicamFeature< int64_t > * [EventAcquisitionTransferEndTimestamp](#)
Returns the Timestamp of the Acquisition Transfer End Event.

- `GenicamFeature< int64_t > * EventAcquisitionTransferEndFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Acquisition Transfer End Event.
- `GenicamFeature< int64_t > * EventAcquisitionError`
Returns the unique Identifier of the Acquisition Error type of Event.
- `GenicamFeature< int64_t > * EventAcquisitionErrorTimestamp`
Returns the Timestamp of the Acquisition Error Event.
- `GenicamFeature< int64_t > * EventAcquisitionErrorFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Acquisition Error Event.
- `GenicamFeature< int64_t > * EventFrameBurstStart`
Returns the unique Identifier of the Frame Burst Start type of Event.
- `GenicamFeature< int64_t > * EventFrameBurstStartTimestamp`
Returns the Timestamp of the Frame Burst Start Event.
- `GenicamFeature< int64_t > * EventFrameBurstStartFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Frame Burst Start Event.
- `GenicamFeature< int64_t > * EventFrameBurstEnd`
Returns the unique Identifier of the Frame Burst End type of Event.
- `GenicamFeature< int64_t > * EventFrameBurstEndTimestamp`
Returns the Timestamp of the Frame Burst End Event.
- `GenicamFeature< int64_t > * EventFrameBurstEndFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Frame Burst End Event.
- `GenicamFeature< int64_t > * EventFrameTrigger`
Returns the unique Identifier of the Frame Trigger type of Event.
- `GenicamFeature< int64_t > * EventFrameTriggerTimestamp`
Returns the Timestamp of the Frame Trigger Event.
- `GenicamFeature< int64_t > * EventFrameTriggerFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Frame Trigger Event.
- `GenicamFeature< int64_t > * EventFrameTriggerMissed`
Returns the unique Identifier of the Frame Trigger Missed type of Event.
- `GenicamFeature< int64_t > * EventFrameTriggerMissedTimestamp`
Returns the Timestamp of the Frame Trigger Missed Event.
- `GenicamFeature< int64_t > * EventFrameTriggerMissedFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Frame Trigger Missed Event.
- `GenicamFeature< int64_t > * EventFrameStart`
Returns the unique Identifier of the Frame Start type of Event.
- `GenicamFeature< int64_t > * EventFrameStartTimestamp`
Returns the Timestamp of the Frame Start Event.
- `GenicamFeature< int64_t > * EventFrameStartFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Frame Start Event.
- `GenicamFeature< int64_t > * EventFrameEnd`
Returns the unique Identifier of the Frame End type of Event.
- `GenicamFeature< int64_t > * EventFrameEndTimestamp`
Returns the Timestamp of the Frame End Event.
- `GenicamFeature< int64_t > * EventFrameEndFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Frame End Event.
- `GenicamFeature< int64_t > * EventFrameTransferStart`
Returns the unique Identifier of the Frame Transfer Start type of Event.
- `GenicamFeature< int64_t > * EventFrameTransferStartTimestamp`
Returns the Timestamp of the Frame Transfer Start Event.
- `GenicamFeature< int64_t > * EventFrameTransferStartFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Frame Transfer Start Event.
- `GenicamFeature< int64_t > * EventFrameTransferEnd`

- Returns the unique Identifier of the Frame Transfer End type of Event.*

 - `GenicamFeature< int64_t > * EventFrameTransferEndTimestamp`
Returns the Timestamp of the Frame Transfer End Event.
 - `GenicamFeature< int64_t > * EventFrameTransferEndFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Frame Transfer End Event.
 - `GenicamFeature< int64_t > * EventLineTrigger`
Returns the unique Identifier of the Line Trigger type of Event.
 - `GenicamFeature< int64_t > * EventLineTriggerTimestamp`
Returns the Timestamp of the Line Trigger Event.
 - `GenicamFeature< int64_t > * EventLineTriggerFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Line Trigger Event.
 - `GenicamFeature< int64_t > * EventLineTriggerMissed`
Returns the unique Identifier of the Line Trigger Missed type of Event.
 - `GenicamFeature< int64_t > * EventLineTriggerMissedTimestamp`
Returns the Timestamp of the Line Trigger Missed Event.
 - `GenicamFeature< int64_t > * EventLineTriggerMissedFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Line Trigger Missed Event.
 - `GenicamFeature< int64_t > * EventLineStart`
Returns the unique Identifier of the Line Start type of Event.
 - `GenicamFeature< int64_t > * EventLineStartTimestamp`
Returns the Timestamp of the Line Start Event.
 - `GenicamFeature< int64_t > * EventLineStartFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Line Start Event.
 - `GenicamFeature< int64_t > * EventLineEnd`
Returns the unique Identifier of the Line End type of Event.
 - `GenicamFeature< int64_t > * EventLineEndTimestamp`
Returns the Timestamp of the Line End Event.
 - `GenicamFeature< int64_t > * EventLineEndFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Line End Event.
 - `GenicamFeature< int64_t > * EventExposureStart`
Returns the unique Identifier of the Exposure Start type of Event.
 - `GenicamFeature< int64_t > * EventExposureStartTimestamp`
Returns the Timestamp of the Exposure Start Event.
 - `GenicamFeature< int64_t > * EventExposureStartFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Exposure Start Event.
 - `GenicamFeature< int64_t > * EventExposureEnd`
Returns the unique Identifier of the Exposure End type of Event.
 - `GenicamFeature< int64_t > * EventExposureEndTimestamp`
Returns the Timestamp of the Exposure End Event.
 - `GenicamFeature< int64_t > * EventExposureEndFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Exposure End Event.
 - `GenicamFeature< int64_t > * EventStream0TransferStart`
Returns the unique Identifier of the Stream 0 Transfer Start type of Event.
 - `GenicamFeature< int64_t > * EventStream0TransferStartTimestamp`
Returns the Timestamp of the Stream 0 Transfer Start Event.
 - `GenicamFeature< int64_t > * EventStream0TransferStartFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Start Event.
 - `GenicamFeature< int64_t > * EventStream0TransferEnd`
Returns the unique Identifier of the Stream 0 Transfer End type of Event.
 - `GenicamFeature< int64_t > * EventStream0TransferEndTimestamp`
Returns the Timestamp of the Stream 0 Transfer End Event.

- [GenicamFeature< int64_t > * EventStream0TransferEndFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer End Event.
- [GenicamFeature< int64_t > * EventStream0TransferPause](#)
Returns the unique Identifier of the Stream 0 Transfer Pause type of Event.
- [GenicamFeature< int64_t > * EventStream0TransferPauseTimestamp](#)
Returns the Timestamp of the Stream 0 Transfer Pause Event.
- [GenicamFeature< int64_t > * EventStream0TransferPauseFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Pause Event.
- [GenicamFeature< int64_t > * EventStream0TransferResume](#)
Returns the unique Identifier of the Stream 0 Transfer Resume type of Event.
- [GenicamFeature< int64_t > * EventStream0TransferResumeTimestamp](#)
Returns the Timestamp of the Stream 0 Transfer Resume Event.
- [GenicamFeature< int64_t > * EventStream0TransferResumeFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Resume Event.
- [GenicamFeature< int64_t > * EventStream0TransferBlockStart](#)
Returns the unique Identifier of the Stream 0 Transfer Block Start type of Event.
- [GenicamFeature< int64_t > * EventStream0TransferBlockStartTimestamp](#)
Returns the Timestamp of the Stream 0 Transfer Block Start Event.
- [GenicamFeature< int64_t > * EventStream0TransferBlockStartFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Block Start Event.
- [GenicamFeature< int64_t > * EventStream0TransferBlockEnd](#)
Returns the unique Identifier of the Stream 0 Transfer Block End type of Event.
- [GenicamFeature< int64_t > * EventStream0TransferBlockEndTimestamp](#)
Returns the Timestamp of the Stream 0 Transfer Block End Event.
- [GenicamFeature< int64_t > * EventStream0TransferBlockEndFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Block End Event.
- [GenicamFeature< int64_t > * EventStream0TransferBlockTrigger](#)
Returns the unique Identifier of the Stream 0 Transfer Block Trigger type of Event.
- [GenicamFeature< int64_t > * EventStream0TransferBlockTriggerTimestamp](#)
Returns the Timestamp of the Stream 0 Transfer Block Trigger Event.
- [GenicamFeature< int64_t > * EventStream0TransferBlockTriggerFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Block Trigger Event.
- [GenicamFeature< int64_t > * EventStream0TransferBurstStart](#)
Returns the unique Identifier of the Stream 0 Transfer Burst Start type of Event.
- [GenicamFeature< int64_t > * EventStream0TransferBurstStartTimestamp](#)
Returns the Timestamp of the Stream 0 Transfer Burst Start Event.
- [GenicamFeature< int64_t > * EventStream0TransferBurstStartFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Burst Start Event.
- [GenicamFeature< int64_t > * EventStream0TransferBurstEnd](#)
Returns the unique Identifier of the Stream 0 Transfer Burst End type of Event.
- [GenicamFeature< int64_t > * EventStream0TransferBurstEndTimestamp](#)
Returns the Timestamp of the Stream 0 Transfer Burst End Event.
- [GenicamFeature< int64_t > * EventStream0TransferBurstEndFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Burst End Event.
- [GenicamFeature< int64_t > * EventStream0TransferOverflow](#)
Returns the unique Identifier of the Stream 0 Transfer Overflow type of Event.
- [GenicamFeature< int64_t > * EventStream0TransferOverflowTimestamp](#)
Returns the Timestamp of the Stream 0 Transfer Overflow Event.
- [GenicamFeature< int64_t > * EventStream0TransferOverflowFrameID](#)
Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Overflow Event.
- [GenicamFeature< int64_t > * EventSequencerSetChange](#)

- Returns the unique Identifier of the Sequencer Set Change type of Event.*

 - `GenicamFeature< int64_t > * EventSequencerSetChangeTimestamp`

Returns the Timestamp of the Sequencer Set Change Event.

 - `GenicamFeature< int64_t > * EventSequencerSetChangeFrameID`

Returns the unique Identifier of the Frame (or image) that generated the Sequencer Set Change Event.

 - `GenicamFeature< int64_t > * EventCounter0Start`

Returns the unique Identifier of the Counter 0 Start type of Event.

 - `GenicamFeature< int64_t > * EventCounter0StartTimestamp`

Returns the Timestamp of the Counter 0 Start Event.

 - `GenicamFeature< int64_t > * EventCounter0StartFrameID`

Returns the unique Identifier of the Frame (or image) that generated the Counter 0 Start Event.

 - `GenicamFeature< int64_t > * EventCounter1Start`

Returns the unique Identifier of the Counter 1 Start type of Event.

 - `GenicamFeature< int64_t > * EventCounter1StartTimestamp`

Returns the Timestamp of the Counter 1 Start Event.

 - `GenicamFeature< int64_t > * EventCounter1StartFrameID`

Returns the unique Identifier of the Frame (or image) that generated the Counter 1 Start Event.

 - `GenicamFeature< int64_t > * EventCounter0End`

Returns the unique Identifier of the Counter 0 End type of Event.

 - `GenicamFeature< int64_t > * EventCounter0EndTimestamp`

Returns the Timestamp of the Counter 0 End Event.

 - `GenicamFeature< int64_t > * EventCounter0EndFrameID`

Returns the unique Identifier of the Frame (or image) that generated the Counter 0 End Event.

 - `GenicamFeature< int64_t > * EventCounter1End`

Returns the unique Identifier of the Counter 1 End type of Event.

 - `GenicamFeature< int64_t > * EventCounter1EndTimestamp`

Returns the Timestamp of the Counter 1 End Event.

 - `GenicamFeature< int64_t > * EventCounter1EndFrameID`

Returns the unique Identifier of the Frame (or image) that generated the Counter 1 End Event.

 - `GenicamFeature< int64_t > * EventTimer0Start`

Returns the unique Identifier of the Timer 0 Start type of Event.

 - `GenicamFeature< int64_t > * EventTimer0StartTimestamp`

Returns the Timestamp of the Timer 0 Start Event.

 - `GenicamFeature< int64_t > * EventTimer0StartFrameID`

Returns the unique Identifier of the Frame (or image) that generated the Timer 0 Start Event.

 - `GenicamFeature< int64_t > * EventTimer1Start`

Returns the unique Identifier of the Timer 1 Start type of Event.

 - `GenicamFeature< int64_t > * EventTimer1StartTimestamp`

Returns the Timestamp of the Timer 1 Start Event.

 - `GenicamFeature< int64_t > * EventTimer1StartFrameID`

Returns the unique Identifier of the Frame (or image) that generated the Timer 1 Start Event.

 - `GenicamFeature< int64_t > * EventTimer0End`

Returns the unique Identifier of the Timer 0 End type of Event.

 - `GenicamFeature< int64_t > * EventTimer0EndTimestamp`

Returns the Timestamp of the Timer 0 End Event.

 - `GenicamFeature< int64_t > * EventTimer0EndFrameID`

Returns the unique Identifier of the Frame (or image) that generated the Timer 0 End Event.

 - `GenicamFeature< int64_t > * EventTimer1End`

Returns the unique Identifier of the Timer 1 End type of Event.

 - `GenicamFeature< int64_t > * EventTimer1EndTimestamp`

Returns the Timestamp of the Timer 1 End Event.

- `GenicamFeature< int64_t > * EventTimer1EndFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Timer 1 End Event.
- `GenicamFeature< int64_t > * EventEncoder0Stopped`
Returns the unique Identifier of the Encoder 0 Stopped type of Event.
- `GenicamFeature< int64_t > * EventEncoder0StoppedTimestamp`
Returns the Timestamp of the Encoder 0 Stopped Event.
- `GenicamFeature< int64_t > * EventEncoder0StoppedFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Encoder 0 Stopped Event.
- `GenicamFeature< int64_t > * EventEncoder1Stopped`
Returns the unique Identifier of the Encoder 1 Stopped type of Event.
- `GenicamFeature< int64_t > * EventEncoder1StoppedTimestamp`
Returns the Timestamp of the Encoder 1 Stopped Event.
- `GenicamFeature< int64_t > * EventEncoder1StoppedFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Encoder 1 Stopped Event.
- `GenicamFeature< int64_t > * EventEncoder0Restarted`
Returns the unique Identifier of the Encoder 0 Restarted type of Event.
- `GenicamFeature< int64_t > * EventEncoder0RestartedTimestamp`
Returns the Timestamp of the Encoder 0 Restarted Event.
- `GenicamFeature< int64_t > * EventEncoder0RestartedFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Encoder 0 Restarted Event.
- `GenicamFeature< int64_t > * EventEncoder1Restarted`
Returns the unique Identifier of the Encoder 1 Restarted type of Event.
- `GenicamFeature< int64_t > * EventEncoder1RestartedTimestamp`
Returns the Timestamp of the Encoder 1 Restarted Event.
- `GenicamFeature< int64_t > * EventEncoder1RestartedFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Encoder 1 Restarted Event.
- `GenicamFeature< int64_t > * EventLine0RisingEdge`
Returns the unique Identifier of the Line 0 Rising Edge type of Event.
- `GenicamFeature< int64_t > * EventLine0RisingEdgeTimestamp`
Returns the Timestamp of the Line 0 Rising Edge Event.
- `GenicamFeature< int64_t > * EventLine0RisingEdgeFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Line 0 Rising Edge Event.
- `GenicamFeature< int64_t > * EventLine1RisingEdge`
Returns the unique Identifier of the Line 1 Rising Edge type of Event.
- `GenicamFeature< int64_t > * EventLine1RisingEdgeTimestamp`
Returns the Timestamp of the Line 1 Rising Edge Event.
- `GenicamFeature< int64_t > * EventLine1RisingEdgeFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Line 1 Rising Edge Event.
- `GenicamFeature< int64_t > * EventLine0FallingEdge`
Returns the unique Identifier of the Line 0 Falling Edge type of Event.
- `GenicamFeature< int64_t > * EventLine0FallingEdgeTimestamp`
Returns the Timestamp of the Line 0 Falling Edge Event.
- `GenicamFeature< int64_t > * EventLine0FallingEdgeFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Line 0 Falling Edge Event.
- `GenicamFeature< int64_t > * EventLine1FallingEdge`
Returns the unique Identifier of the Line 1 Falling Edge type of Event.
- `GenicamFeature< int64_t > * EventLine1FallingEdgeTimestamp`
Returns the Timestamp of the Line 1 Falling Edge Event.
- `GenicamFeature< int64_t > * EventLine1FallingEdgeFrameID`
Returns the unique Identifier of the Frame (or image) that generated the Line 1 Falling Edge Event.
- `GenicamFeature< int64_t > * EventLine0AnyEdge`

- Returns the unique Identifier of the Line 0 Any Edge type of Event.*

 - `GenicamFeature< int64_t > * EventLine0AnyEdgeTimestamp`
- Returns the Timestamp of the Line 0 Any Edge Event.*

 - `GenicamFeature< int64_t > * EventLine0AnyEdgeFrameID`
- Returns the unique Identifier of the Frame (or image) that generated the Line 0 Any Edge Event.*

 - `GenicamFeature< int64_t > * EventLine1AnyEdge`
- Returns the unique Identifier of the Line 1 Any Edge type of Event.*

 - `GenicamFeature< int64_t > * EventLine1AnyEdgeTimestamp`
- Returns the Timestamp of the Line 1 Any Edge Event.*

 - `GenicamFeature< int64_t > * EventLine1AnyEdgeFrameID`
- Returns the unique Identifier of the Frame (or image) that generated the Line 1 Any Edge Event.*

 - `GenicamFeature< int64_t > * EventLinkTrigger0`
- Returns the unique Identifier of the Link Trigger 0 type of Event.*

 - `GenicamFeature< int64_t > * EventLinkTrigger0Timestamp`
- Returns the Timestamp of the Link Trigger 0 Event.*

 - `GenicamFeature< int64_t > * EventLinkTrigger0FrameID`
- Returns the unique Identifier of the Frame (or image) that generated the Link Trigger 0 Event.*

 - `GenicamFeature< int64_t > * EventLinkTrigger1`
- Returns the unique Identifier of the Link Trigger 1 type of Event.*

 - `GenicamFeature< int64_t > * EventLinkTrigger1Timestamp`
- Returns the Timestamp of the Link Trigger 1 Event.*

 - `GenicamFeature< int64_t > * EventLinkTrigger1FrameID`
- Returns the unique Identifier of the Frame (or image) that generated the Link Trigger 1 Event.*

 - `GenicamFeature< int64_t > * EventLinkSpeedChange`
- Returns the unique Identifier of the Link Speed Change type of Event.*

 - `GenicamFeature< int64_t > * EventLinkSpeedChangeTimestamp`
- Returns the Timestamp of the Link Speed Change Event.*

 - `GenicamFeature< int64_t > * EventLinkSpeedChangeFrameID`
- Returns the unique Identifier of the Frame (or image) that generated the Link Speed Change Event.*

 - `GenicamFeature< int64_t > * EventActionLate`
- Returns the unique Identifier of the Action Late type of Event.*

 - `GenicamFeature< int64_t > * EventActionLateTimestamp`
- Returns the Timestamp of the Action Late Event.*

 - `GenicamFeature< int64_t > * EventActionLateFrameID`
- Returns the unique Identifier of the Frame (or image) that generated the Action Late Event.*

 - `GenicamFeature< int64_t > * EventPrimaryApplicationSwitch`
- Returns the unique Identifier of the Primary Application Switch type of Event.*

 - `GenicamFeature< int64_t > * EventPrimaryApplicationSwitchTimestamp`
- Returns the Timestamp of the Primary Application Switch Event.*

 - `GenicamFeature< int64_t > * EventPrimaryApplicationSwitchFrameID`
- Returns the unique Identifier of the Frame (or image) that generated the Primary Application Switch Event.*

 - `GenicamFeature< int64_t > * EventError`
- Returns the unique identifier of the Error type of Event. It can be used to register a callback function to be notified of the Error event occurrence. Its value uniquely identifies that the event received was an Error.*

 - `GenicamFeature< int64_t > * EventErrorTimestamp`
- Returns the Timestamp of the Error Event. It can be used to determine when the event occurred.*

 - `GenicamFeature< int64_t > * EventErrorFrameID`
- If applicable, returns the unique Identifier of the Frame (or image) that generated the Error Event.*

 - `GenicamFeature< int64_t > * EventErrorCode`
- Returns an error code for the error(s) that happened.*

 - `GenicamFeature< int64_t > * EventTest`

Returns the unique identifier of the Event Test type of event generated using the TestEventGenerate command. It can be used to register a callback function to be notified of the EventTest event occurrence. Its value uniquely identifies that the event received was an Event Test.

- GenicamFeature< int64_t > * [EventTestTimestamp](#)
Returns the Timestamp of the Event Test event. It can be used to determine when the event occurred.
- GenicamFeature< FliSfncCameraEnum::UserSetSelectorEnum > * [UserSetSelector](#)
Selects the feature User Set to load, save or configure.
- GenicamFeature * [UserSetLoad](#)
Loads the User Set specified by UserSetSelector to the device and makes it active.
- GenicamFeature * [UserSetSave](#)
Save the User Set specified by UserSetSelector to the non-volatile memory of the device.
- GenicamFeature< FliSfncCameraEnum::UserSetDefaultEnum > * [UserSetDefault](#)
Selects the feature User Set to load and make active by default when the device is reset.
- GenicamFeature< FliSfncCameraEnum::UserSetFeatureSelectorEnum > * [UserSetFeatureSelector](#)
Selects which individual UserSet feature to control.
- GenicamFeature< bool > * [UserSetFeatureEnable](#)
Enables the selected feature and make it active in all the UserSets.
- GenicamFeature< FliSfncCameraEnum::SequencerModeEnum > * [SequencerMode](#)
Controls if the sequencer mechanism is active.
- GenicamFeature< FliSfncCameraEnum::SequencerConfigurationModeEnum > * [SequencerConfigurationMode](#)
Controls if the sequencer configuration mode is active.
- GenicamFeature< FliSfncCameraEnum::SequencerFeatureSelectorEnum > * [SequencerFeatureSelector](#)
Selects which sequencer features to control.
- GenicamFeature< bool > * [SequencerFeatureEnable](#)
Enables the selected feature and make it active in all the sequencer sets.
- GenicamFeature< int64_t > * [SequencerSetSelector](#)
Selects the sequencer set to which further feature settings applies.
- GenicamFeature * [SequencerSetSave](#)
Saves the current device state to the sequencer set selected by the SequencerSetSelector.
- GenicamFeature * [SequencerSetLoad](#)
Loads the sequencer set selected by SequencerSetSelector in the device. Even if SequencerMode is off, this will change the device state to the configuration of the selected set.
- GenicamFeature< int64_t > * [SequencerSetActive](#)
Contains the currently active sequencer set.
- GenicamFeature< int64_t > * [SequencerSetStart](#)
Sets the initial/start sequencer set, which is the first set used within a sequencer.
- GenicamFeature< int64_t > * [SequencerPathSelector](#)
Selects to which branching path further path settings applies.
- GenicamFeature< int64_t > * [SequencerSetNext](#)
Specifies the next sequencer set.
- GenicamFeature< FliSfncCameraEnum::SequencerTriggerSourceEnum > * [SequencerTriggerSource](#)
Specifies the internal signal or physical input line to use as the sequencer trigger source.
- GenicamFeature< FliSfncCameraEnum::SequencerTriggerActivationEnum > * [SequencerTriggerActivation](#)
Specifies the activation mode of the sequencer trigger.
- GenicamFeature< FliSfncCameraEnum::FileSelectorEnum > * [FileSelector](#)
Selects the target file in the device.
- GenicamFeature< FliSfncCameraEnum::FileOperationSelectorEnum > * [FileOperationSelector](#)
Selects the target operation for the selected file in the device. This Operation is executed when the FileOperation↔Execute feature is called.
- GenicamFeature * [FileOperationExecute](#)
Executes the operation selected by FileOperationSelector on the selected file.

- GenicamFeature< FliSfncCameraEnum::FileOpenModeEnum > * [FileOpenMode](#)
Selects the access mode in which a file is opened in the device.
- GenicamFeature< int64_t > * [FileAccessOffset](#)
Controls the Offset of the mapping between the device file storage and the FileAccessBuffer.
- GenicamFeature< int64_t > * [FileAccessLength](#)
Controls the Length of the mapping between the device file storage and the FileAccessBuffer.
- GenicamFeature< FliSfncCameraEnum::FileOperationStatusEnum > * [FileOperationStatus](#)
Represents the file operation execution status.
- GenicamFeature< int64_t > * [FileOperationResult](#)
Represents the file operation result. For Read or Write operations, the number of successfully read/written bytes is returned.
- GenicamFeature< int64_t > * [FileSize](#)
Represents the size of the selected file in bytes.
- GenicamFeature< int64_t > * [SourceCount](#)
Controls or returns the number of sources supported by the device.
- GenicamFeature< FliSfncCameraEnum::SourceSelectorEnum > * [SourceSelector](#)
Selects the source to control.
- GenicamFeature< int64_t > * [SourceIDValue](#)
Returns a unique Identifier value that correspond to the selected Source.
- GenicamFeature< FliSfncCameraEnum::TransferSelectorEnum > * [TransferSelector](#)
Selects which stream transfers are currently controlled by the selected Transfer features.
- GenicamFeature< FliSfncCameraEnum::TransferControlModeEnum > * [TransferControlMode](#)
Selects the control method for the transfers.
- GenicamFeature< FliSfncCameraEnum::TransferOperationModeEnum > * [TransferOperationMode](#)
Selects the operation mode of the transfer.
- GenicamFeature< int64_t > * [TransferBlockCount](#)
Specifies the number of data Blocks that the device should stream before stopping. This feature is only active if the TransferOperationMode is set to MultiBlock.
- GenicamFeature< int64_t > * [TransferBurstCount](#)
Number of Block(s) to transfer for each TransferBurstStart trigger.
- GenicamFeature< int64_t > * [TransferQueueMaxBlockCount](#)
Controls the maximum number of data blocks that can be stored in the block queue of the selected stream.
- GenicamFeature< int64_t > * [TransferQueueCurrentBlockCount](#)
Returns the number of Block(s) currently in the transfer queue.
- GenicamFeature< FliSfncCameraEnum::TransferQueueModeEnum > * [TransferQueueMode](#)
Specifies the operation mode of the transfer queue.
- GenicamFeature * [TransferStart](#)
Starts the streaming of data blocks out of the device. This feature must be available when the TransferControlMode is set to "UserControlled". If the TransferStart feature is not writable (locked), the application should not start the transfer and should avoid using the feature until it becomes writable again.
- GenicamFeature * [TransferStop](#)
Stops the streaming of data Block(s). The current block transmission will be completed. This feature must be available when the TransferControlMode is set to "UserControlled".
- GenicamFeature * [TransferAbort](#)
Aborts immediately the streaming of data block(s). Aborting the transfer will result in the lost of the data that is present or currently entering in the block queue. However, the next new block received will be stored in the queue and transferred to the host when the streaming is restarted. If implemented, this feature should be available when the TransferControlMode is set to "UserControlled".
- GenicamFeature * [TransferPause](#)
Pauses the streaming of data Block(s). Pausing the streaming will immediately suspend the ongoing data transfer even if a block is partially transfered. The device will resume its transmission at the reception of a TransferResume command.
- GenicamFeature * [TransferResume](#)

- Resumes a data Blocks streaming that was previously paused by a TransferPause command.*
- GenicamFeature< FliSfncCameraEnum::TransferTriggerSelectorEnum > * [TransferTriggerSelector](#)
Selects the type of transfer trigger to configure.
 - GenicamFeature< FliSfncCameraEnum::TransferTriggerModeEnum > * [TransferTriggerMode](#)
Controls if the selected trigger is active.
 - GenicamFeature< FliSfncCameraEnum::TransferTriggerSourceEnum > * [TransferTriggerSource](#)
Specifies the signal to use as the trigger source for transfers.
 - GenicamFeature< FliSfncCameraEnum::TransferTriggerActivationEnum > * [TransferTriggerActivation](#)
Specifies the activation mode of the transfer control trigger.
 - GenicamFeature< FliSfncCameraEnum::TransferStatusSelectorEnum > * [TransferStatusSelector](#)
Selects which status of the transfer module to read.
 - GenicamFeature< bool > * [TransferStatus](#)
Reads the status of the Transfer module signal selected by TransferStatusSelector.
 - GenicamFeature< FliSfncCameraEnum::TransferComponentSelectorEnum > * [TransferComponentSelector](#)
Selects the color component for the control of the TransferStreamChannel feature.
 - GenicamFeature< int64_t > * [TransferStreamChannel](#)
Selects the streaming channel that will be used to transfer the selected stream of data. In general, this feature can be omitted and the default streaming channel will be used.
 - GenicamFeature< FliSfncCameraEnum::Scan3dExtractionSelectorEnum > * [Scan3dExtractionSelector](#)
Selects the 3DExtraction processing module to control (if multiple ones are present).
 - GenicamFeature< FliSfncCameraEnum::Scan3dExtractionSourceEnum > * [Scan3dExtractionSource](#)
Selects the sensor's data source region for 3D Extraction module.
 - GenicamFeature< FliSfncCameraEnum::Scan3dExtractionMethodEnum > * [Scan3dExtractionMethod](#)
Selects the method for extracting 3D from the input sensor data.
 - GenicamFeature< FliSfncCameraEnum::Scan3dDistanceUnitEnum > * [Scan3dDistanceUnit](#)
Specifies the unit used when delivering (calibrated) distance data.
 - GenicamFeature< FliSfncCameraEnum::Scan3dCoordinateSystemEnum > * [Scan3dCoordinateSystem](#)
Specifies the Coordinate system to use for the device.
 - GenicamFeature< FliSfncCameraEnum::Scan3dOutputModeEnum > * [Scan3dOutputMode](#)
Controls the Calibration and data organization of the device and the coordinates transmitted.
 - GenicamFeature< FliSfncCameraEnum::Scan3dCoordinateSystemReferenceEnum > * [Scan3dCoordinateSystemReference](#)
Defines coordinate system reference location.
 - GenicamFeature< FliSfncCameraEnum::Scan3dCoordinateSelectorEnum > * [Scan3dCoordinateSelector](#)
Selects the individual coordinates in the vectors for 3D information/transformation.
 - GenicamFeature< double > * [Scan3dCoordinateScale](#)
Scale factor when transforming a pixel from relative coordinates to world coordinates.
 - GenicamFeature< double > * [Scan3dCoordinateOffset](#)
Offset when transforming a pixel from relative coordinates to world coordinates.
 - GenicamFeature< bool > * [Scan3dInvalidDataFlag](#)
Enables the definition of a non-valid flag value in the data stream. Note that the confidence output is an alternate recommended way to identify non-valid pixels. Using a Scan3dInvalidDataValue may give processing penalties due to special handling.
 - GenicamFeature< double > * [Scan3dInvalidDataValue](#)
Value which identifies a non-valid pixel if Scan3dInvalidDataFlag is enabled.
 - GenicamFeature< double > * [Scan3dAxisMin](#)
Minimum valid transmitted coordinate value of the selected Axis.
 - GenicamFeature< double > * [Scan3dAxisMax](#)
Maximum valid transmitted coordinate value of the selected Axis.
 - GenicamFeature< FliSfncCameraEnum::Scan3dCoordinateTransformSelectorEnum > * [Scan3dCoordinateTransformSelector](#)
Sets the index to read/write a coordinate transform value.
 - GenicamFeature< double > * [Scan3dTransformValue](#)

Specifies the transform value selected. For translations (`Scan3dCoordinateTransformSelector = TranslationX/Y/Z`) it is expressed in the distance unit of the system, for rotations (`Scan3dCoordinateTransformSelector = RotationX/Y/Z`) in degrees.

- `GenicamFeature< FliSfncCameraEnum::Scan3dCoordinateReferenceSelectorEnum > * Scan3dCoordinateReferenceSelector`
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.
- `GenicamFeature< double > * Scan3dCoordinateReferenceValue`
Returns the reference value selected. Reads the value of a rotation or translation value for the current (Anchor or Transformed) coordinate system transformation to the Reference system.
- `GenicamFeature< double > * Scan3dFocalLength`
Returns the focal length of the camera in pixel. The focal length depends on the selected region. The value of this feature takes into account horizontal binning, decimation, or any other function changing the image resolution.
- `GenicamFeature< double > * Scan3dBaseline`
Returns the baseline as the physical distance of two cameras in a stereo camera setup. The value of this feature can be used for 3D reconstruction from disparity images. In this case, the unit of the 3D coordinates corresponds to the unit of the baseline.
- `GenicamFeature< double > * Scan3dPrincipalPointU`
Returns the value of the horizontal position of the principal point, relative to the region origin, i.e. OffsetX. The value of this feature takes into account horizontal binning, decimation, or any other function changing the image resolution.
- `GenicamFeature< double > * Scan3dPrincipalPointV`
Returns the value of the vertical position of the principal point, relative to the region origin, i.e. OffsetY. The value of this feature takes into account vertical binning, decimation, or any other function changing the image resolution.
- `GenicamFeature< FliSfncCameraEnum::LightControllerSelectorEnum > * LightControllerSelector`
Selects the Light Controller to configure.
- `GenicamFeature< FliSfncCameraEnum::LightControllerSourceEnum > * LightControllerSource`
Selects the input source signal of the Light Controller.
- `GenicamFeature< double > * LightCurrentRating`
Set the current rating of the lighting output.
- `GenicamFeature< double > * LightVoltageRating`
Set the voltage rating of the lighting output.
- `GenicamFeature< double > * LightBrightness`
Set the brightness of the lighting output in percent. Can be greater than 100% for short overdrive period.
- `GenicamFeature< FliSfncCameraEnum::LightConnectionStatusEnum > * LightConnectionStatus`
Status of a light connected to the controller's output Line.
- `GenicamFeature< double > * LightCurrentMeasured`
The measured current applied to the lighting.
- `GenicamFeature< double > * LightVoltageMeasured`
The measured voltage applied to the lighting.
- `GenicamFeature< bool > * ChunkModeActive`
Activates the inclusion of Chunk data in the transmitted payload.
- `GenicamFeature< bool > * ChunkXMLEnable`
Activates the inclusion of the GenICam XML necessary to the chunk parser to decode all the Chunk data included in the transmitted payload.
- `GenicamFeature< FliSfncCameraEnum::ChunkSelectorEnum > * ChunkSelector`
Selects which Chunk to enable or control.
- `GenicamFeature< bool > * ChunkEnable`
Enables the inclusion of the selected Chunk data in the payload of the image.
- `GenicamFeature< FliSfncCameraEnum::ChunkRegionSelectorEnum > * ChunkRegionSelector`
Selects which Region to retrieve data from.
- `GenicamFeature< FliSfncCameraEnum::ChunkRegionIDEnum > * ChunkRegionID`
Returns the Identifier of Region that the image comes from.
- `GenicamFeature< int64_t > * ChunkRegionIDValue`
Returns the unique integer Identifier value of the Region that the image comes from.

- GenicamFeature< FliSfncCameraEnum::ChunkComponentSelectorEnum > * [ChunkComponentSelector](#)
Selects the Component from which to retrieve data from.
- GenicamFeature< FliSfncCameraEnum::ChunkComponentIDEnum > * [ChunkComponentID](#)
Returns the Identifier of the selected Component. This can be used to identify the image component type of a multi-component payload.
- GenicamFeature< int64_t > * [ChunkComponentIDValue](#)
Returns a unique Identifier value that corresponds to the selected chunk Component.
- GenicamFeature< FliSfncCameraEnum::ChunkGroupSelectorEnum > * [ChunkGroupSelector](#)
Selects the component Group from which to retrieve data from.
- GenicamFeature< FliSfncCameraEnum::ChunkGroupIDEnum > * [ChunkGroupID](#)
Returns a unique Identifier corresponding to the selected Group of components. This can be used to identify the component Group of a multi-group payload.
- GenicamFeature< int64_t > * [ChunkGroupIDValue](#)
Returns a unique Identifier value that corresponds to the Group of Components of the selected chunk Component.
- GenicamFeature< int64_t > * [ChunkOffsetX](#)
Returns the OffsetX of the image included in the payload.
- GenicamFeature< int64_t > * [ChunkOffsetY](#)
Returns the OffsetY of the image included in the payload.
- GenicamFeature< int64_t > * [ChunkWidth](#)
Returns the Width of the image included in the payload.
- GenicamFeature< int64_t > * [ChunkHeight](#)
Returns the Height of the image included in the payload.
- GenicamFeature< FliSfncCameraEnum::ChunkPixelFormatEnum > * [ChunkPixelFormat](#)
Returns the PixelFormat of the image included in the payload.
- GenicamFeature< int64_t > * [ChunkPixelDynamicRangeMin](#)
Returns the minimum value of dynamic range of the image included in the payload.
- GenicamFeature< int64_t > * [ChunkPixelDynamicRangeMax](#)
Returns the maximum value of dynamic range of the image included in the payload.
- GenicamFeature< int64_t > * [ChunkBinningHorizontal](#)
Number of horizontal photo-sensitive cells combined together.
- GenicamFeature< int64_t > * [ChunkBinningVertical](#)
Number of vertical photo-sensitive cells combined together.
- GenicamFeature< int64_t > * [ChunkDecimationHorizontal](#)
Horizontal sub-sampling of the image.
- GenicamFeature< int64_t > * [ChunkDecimationVertical](#)
Vertical sub-sampling of the image.
- GenicamFeature< bool > * [ChunkReverseX](#)
Flip horizontal of the image sent by the device.
- GenicamFeature< bool > * [ChunkReverseY](#)
Flip vertically of the image sent by the device.
- GenicamFeature< int64_t > * [ChunkTimestamp](#)
Returns the Timestamp of the image included in the payload at the time of the FrameStart internal event.
- GenicamFeature< int64_t > * [ChunkTimestampLatchValue](#)
Returns the last Timestamp latched with the TimestampLatch command.
- GenicamFeature< int64_t > * [ChunkLineStatusAll](#)
Returns the status of all the I/O lines at the time of the FrameStart internal event.
- GenicamFeature< FliSfncCameraEnum::ChunkCounterSelectorEnum > * [ChunkCounterSelector](#)
Selects which counter to retrieve data from.
- GenicamFeature< int64_t > * [ChunkCounterValue](#)
Returns the value of the selected Chunk counter at the time of the FrameStart event.
- GenicamFeature< FliSfncCameraEnum::ChunkTimerSelectorEnum > * [ChunkTimerSelector](#)

- Selects which Timer to retrieve data from.*

 - GenicamFeature< double > * [ChunkTimerValue](#)

Returns the value of the selected Timer at the time of the FrameStart internal event.
- GenicamFeature< int64_t > * [ChunkScanLineSelector](#)

Index for vector representation of one chunk value per line in an image.
- GenicamFeature< FliSfncCameraEnum::ChunkEncoderSelectorEnum > * [ChunkEncoderSelector](#)

Selects which Encoder to retrieve data from.
- GenicamFeature< int64_t > * [ChunkEncoderValue](#)

Returns the counter's value of the selected Encoder at the time of the FrameStart in area scan mode or the counter's value at the time of the LineStart selected by ChunkScanLineSelector in Linescan mode.
- GenicamFeature< FliSfncCameraEnum::ChunkEncoderStatusEnum > * [ChunkEncoderStatus](#)

Returns the motion status of the selected encoder.
- GenicamFeature< FliSfncCameraEnum::ChunkExposureTimeSelectorEnum > * [ChunkExposureTimeSelector](#)

Selects which exposure time is read by the ChunkExposureTime feature.
- GenicamFeature< double > * [ChunkExposureTime](#)

Returns the exposure time used to capture the image.
- GenicamFeature< FliSfncCameraEnum::ChunkGainSelectorEnum > * [ChunkGainSelector](#)

Selects which Gain to return.
- GenicamFeature< double > * [ChunkGain](#)

Returns the gain used to capture the image.
- GenicamFeature< FliSfncCameraEnum::ChunkBlackLevelSelectorEnum > * [ChunkBlackLevelSelector](#)

Selects which Black Level to return. Possible values are:
- GenicamFeature< double > * [ChunkBlackLevel](#)

Returns the black level used to capture the image included in the payload.
- GenicamFeature< int64_t > * [ChunkLinePitch](#)

Returns the LinePitch of the image included in the payload.
- GenicamFeature< int64_t > * [ChunkFrameID](#)

Returns the unique Identifier of the frame (or image) included in the payload.
- GenicamFeature< FliSfncCameraEnum::ChunkSourceSelectorEnum > * [ChunkSourceSelector](#)

Selects which Source to retrieve data from.
- GenicamFeature< FliSfncCameraEnum::ChunkSourceIDEnum > * [ChunkSourceID](#)

Returns the Identifier of Source that the image comes from.
- GenicamFeature< int64_t > * [ChunkSourceIDValue](#)

Returns the unique integer Identifier value of the Source that the image comes from.
- GenicamFeature< int64_t > * [ChunkTransferBlockID](#)

Returns the unique identifier of the transfer block used to transport the payload.
- GenicamFeature< FliSfncCameraEnum::ChunkTransferStreamIDEnum > * [ChunkTransferStreamID](#)

Returns identifier of the stream that generated this block.
- GenicamFeature< int64_t > * [ChunkTransferQueueCurrentBlockCount](#)

Returns the current number of blocks in the transfer queue.
- GenicamFeature< int64_t > * [ChunkStreamChannelID](#)

Returns identifier of the stream channel used to carry the block.
- GenicamFeature< int64_t > * [ChunkSequencerSetActive](#)

Return the index of the active set of the running sequencer included in the payload.
- GenicamFeature< FliSfncCameraEnum::ChunkScan3dDistanceUnitEnum > * [ChunkScan3dDistanceUnit](#)

Returns the Distance Unit of the payload image.
- GenicamFeature< FliSfncCameraEnum::ChunkScan3dOutputModeEnum > * [ChunkScan3dOutputMode](#)

Returns the Calibrated Mode of the payload image.
- GenicamFeature< FliSfncCameraEnum::ChunkScan3dCoordinateSystemEnum > * [ChunkScan3dCoordinateSystem](#)

Returns the Coordinate System of the image included in the payload.
- GenicamFeature< FliSfncCameraEnum::ChunkScan3dCoordinateSystemReferenceEnum > * [ChunkScan3dCoordinateSystemReference](#)

- Returns the Coordinate System Position of the image included in the payload.*

 - GenicamFeature< FliSfncCameraEnum::ChunkScan3dCoordinateSelectorEnum > * [ChunkScan3dCoordinateSelector](#)
Selects which Coordinate to retrieve data from.
 - GenicamFeature< double > * [ChunkScan3dCoordinateScale](#)
Returns the Scale for the selected coordinate axis of the image included in the payload.
 - GenicamFeature< double > * [ChunkScan3dCoordinateOffset](#)
Returns the Offset for the selected coordinate axis of the image included in the payload.
 - GenicamFeature< bool > * [ChunkScan3dInvalidDataFlag](#)
Returns if a specific non-valid data flag is used in the data stream.
 - GenicamFeature< double > * [ChunkScan3dInvalidDataValue](#)
Returns the Invalid Data Value used for the image included in the payload.
 - GenicamFeature< double > * [ChunkScan3dAxisMin](#)
Returns the Minimum Axis value for the selected coordinate axis of the image included in the payload.
 - GenicamFeature< double > * [ChunkScan3dAxisMax](#)
Returns the Maximum Axis value for the selected coordinate axis of the image included in the payload.
 - GenicamFeature< FliSfncCameraEnum::ChunkScan3dCoordinateTransformSelectorEnum > * [ChunkScan3dCoordinateTransformSelector](#)
Selector for transform values.
 - GenicamFeature< double > * [ChunkScan3dTransformValue](#)
Returns the transform value.
 - GenicamFeature< FliSfncCameraEnum::ChunkScan3dCoordinateReferenceSelectorEnum > * [ChunkScan3dCoordinateReferenceSelector](#)
Selector to read a coordinate system reference value defining the transform of a point from one system to the other.
 - GenicamFeature< double > * [ChunkScan3dCoordinateReferenceValue](#)
Returns the value of a position or pose coordinate for the anchor or transformed coordinate systems relative to the reference point.
 - GenicamFeature< double > * [ChunkScan3dFocalLength](#)
Returns the focal length of the camera in pixel. The focal length depends on the selected region. The value of this feature takes into account horizontal binning, decimation, or any other function changing the image resolution.
 - GenicamFeature< double > * [ChunkScan3dBaseline](#)
Returns the baseline as the physical distance of two cameras in a stereo camera setup. The value of this feature can be used for 3D reconstruction from disparity images. In this case, the unit of the 3D coordinates corresponds to the unit of the baseline.
 - GenicamFeature< double > * [ChunkScan3dPrincipalPointU](#)
Returns the value of this feature gives the horizontal position of the principal point, relative to the region origin, i.e. OffsetX. The value of this feature takes into account horizontal binning, decimation, or any other function changing the image resolution.
 - GenicamFeature< double > * [ChunkScan3dPrincipalPointV](#)
Returns the value of this feature gives the vertical position of the principal point, relative to the region origin, i.e. OffsetY. The value of this feature takes into account vertical binning, decimation, or any other function changing the image resolution.
 - GenicamFeature< int64_t > * [TestPendingAck](#)
Tests the device's pending acknowledge feature. When this feature is written, the device waits a time period corresponding to the value of TestPendingAck before acknowledging the write.
 - GenicamFeature * [TestEventGenerate](#)
Generates a Test Event.
 - GenicamFeature< FliSfncCameraEnum::TestPayloadFormatModeEnum > * [TestPayloadFormatMode](#)
This feature allows setting a device in test mode and to output a specific payload format for validation of data streaming. This feature is intended solely for test purposes. The data can be real acquired data or any test pattern.
 - GenicamFeature< int64_t > * [TLParamsLocked](#)
Used by the Transport Layer to prevent critical features from changing during acquisition.
 - GenicamFeature< FliSfncCameraEnum::TLParamsLockedSelectorEnum > * [TLParamsLockedSelector](#)
Selects the type of feature for which the locking behavior will be configured.
 - GenicamFeature< bool > * [TLParamsLockedState](#)
Controls if the selected parameters are locked during acquisition.

- [GenicamFeature< int64_t > * PayloadSize](#)
Provides the number of bytes transferred for each data buffer or chunk on the stream channel. This includes any end-of-line, end-of-frame statistics or other stamp data. This is the total size of data payload for a data block.
- [GenicamFeature< FliSfncCameraEnum::GenDCStreamingModeEnum > * GenDCStreamingMode](#)
Controls the device's streaming format.
- [GenicamFeature< FliSfncCameraEnum::GenDCStreamingStatusEnum > * GenDCStreamingStatus](#)
Returns whether the current device data streaming format is GenDC. This value is conditioned by the GenDC↔StreamingMode.
- [GenicamFeature< FliSfncCameraEnum::DeviceTapGeometryEnum > * DeviceTapGeometry](#)
This device tap geometry feature describes the geometrical properties characterizing the taps of a camera as presented at the output of the device.
- [GenicamFeature< bool > * PtpEnable](#)
Enables the Precision Time Protocol (PTP).
- [GenicamFeature< FliSfncCameraEnum::PtpClockAccuracyEnum > * PtpClockAccuracy](#)
Indicates the expected accuracy of the device PTP clock when it is the grandmaster, or in the event it becomes the grandmaster.
- [GenicamFeature * PtpDataSetLatch](#)
Latches the current values from the device's PTP clock data set.
- [GenicamFeature< FliSfncCameraEnum::PtpStatusEnum > * PtpStatus](#)
Returns the latched state of the PTP clock.
- [GenicamFeature< FliSfncCameraEnum::PtpServoStatusEnum > * PtpServoStatus](#)
Returns the latched state of the clock servo. When the servo is in a locked state, the value returned is 'Locked'. When the servo is in a non-locked state, a device-specific value can be returned to give specific information. If no device-specific value is available to describe the current state of the clock servo, the value should be 'Unknown'.
- [GenicamFeature< int64_t > * PtpOffsetFromMaster](#)
Returns the latched offset from the PTP master clock in nanoseconds.
- [GenicamFeature< int64_t > * PtpClockID](#)
Returns the latched clock ID of the PTP device.
- [GenicamFeature< int64_t > * PtpParentClockID](#)
Returns the latched parent clock ID of the PTP device. The parent clock ID is the clock ID of the current master clock.
- [GenicamFeature< int64_t > * PtpGrandmasterClockID](#)
Returns the latched grandmaster clock ID of the PTP device. The grandmaster clock ID is the clock ID of the current grandmaster clock.
- [GenicamFeature< FliSfncCameraEnum::GevPhysicalLinkConfigurationEnum > * GevPhysicalLinkConfiguration](#)
Controls the principal physical link configuration to use on next restart/power-up of the device.
- [GenicamFeature< FliSfncCameraEnum::GevCurrentPhysicalLinkConfigurationEnum > * GevCurrentPhysicalLinkConfiguration](#)
Indicates the current physical link configuration of the device.
- [GenicamFeature< int64_t > * GevActiveLinkCount](#)
Indicates the current number of active logical links.
- [GenicamFeature< FliSfncCameraEnum::GevSupportedOptionSelectorEnum > * GevSupportedOptionSelector](#)
Selects the GEV option to interrogate for existing support.
- [GenicamFeature< bool > * GevSupportedOption](#)
Returns if the selected GEV option is supported.
- [GenicamFeature< int64_t > * GevInterfaceSelector](#)
Selects which logical link to control.
- [GenicamFeature< int64_t > * GevMACAddress](#)
MAC address of the logical link.
- [GenicamFeature< bool > * GevPAUSEFrameReception](#)
Controls whether incoming PAUSE Frames are handled on the given logical link.
- [GenicamFeature< bool > * GevPAUSEFrameTransmission](#)
Controls whether PAUSE Frames can be generated on the given logical link.
- [GenicamFeature< bool > * GevCurrentIPConfigurationLLA](#)

- Controls whether the Link Local Address IP configuration scheme is activated on the given logical link.*

 - GenicamFeature< bool > * [GevCurrentIPConfigurationDHCP](#)
- Controls whether the DHCP IP configuration scheme is activated on the given logical link.*

 - GenicamFeature< bool > * [GevCurrentIPConfigurationPersistentIP](#)
- Controls whether the PersistentIP configuration scheme is activated on the given logical link.*

 - GenicamFeature< int64_t > * [GevCurrentIPAddress](#)
- Reports the IP address for the given logical link.*

 - GenicamFeature< int64_t > * [GevCurrentSubnetMask](#)
- Reports the subnet mask of the given logical link.*

 - GenicamFeature< int64_t > * [GevCurrentDefaultGateway](#)
- Reports the default gateway IP address of the given logical link.*

 - GenicamFeature< FliSfncCameraEnum::GevIPConfigurationStatusEnum > * [GevIPConfigurationStatus](#)
- Reports the current IP configuration status.*

 - GenicamFeature< std::string > * [GevFirstURL](#)
- Indicates the first URL to the GenICam XML device description file. The First URL is used as the first choice by the application to retrieve the GenICam XML device description file.*

 - GenicamFeature< std::string > * [GevSecondURL](#)
- Indicates the second URL to the GenICam XML device description file. This URL is an alternative if the application was unsuccessful to retrieve the device description file using the first URL.*

 - GenicamFeature< int64_t > * [GevPersistentIPAddress](#)
- Controls the Persistent IP address for this logical link. It is only used when the device boots with the Persistent IP configuration scheme.*

 - GenicamFeature< int64_t > * [GevPersistentSubnetMask](#)
- Controls the Persistent subnet mask associated with the Persistent IP address on this logical link. It is only used when the device boots with the Persistent IP configuration scheme.*

 - GenicamFeature< int64_t > * [GevPersistentDefaultGateway](#)
- Controls the persistent default gateway for this logical link. It is only used when the device boots with the Persistent IP configuration scheme.*

 - GenicamFeature< int64_t > * [GevDiscoveryAckDelay](#)
- Indicates the maximum randomized delay the device will wait to acknowledge a discovery command.*

 - GenicamFeature< FliSfncCameraEnum::GevGVCPExtendedStatusCodesSelectorEnum > * [GevGVCPExtendedStatusCodes](#)
- Selects the GigE Vision version to control extended status codes for.*

 - GenicamFeature< bool > * [GevGVCPExtendedStatusCodes](#)
- Enables the generation of extended status codes.*

 - GenicamFeature< bool > * [GevGVCPPendingAck](#)
- Enables the generation of PENDING_ACK.*

 - GenicamFeature< int64_t > * [GevPrimaryApplicationSwitchoverKey](#)
- Controls the key to use to authenticate primary application switchover requests.*

 - GenicamFeature< FliSfncCameraEnum::GevGVSPExtendedIDModeEnum > * [GevGVSPExtendedIDMode](#)
- Enables the extended IDs mode.*

 - GenicamFeature< FliSfncCameraEnum::GevCCPEnum > * [GevCCP](#)
- Controls the device access privilege of an application.*

 - GenicamFeature< int64_t > * [GevPrimaryApplicationSocket](#)
- Returns the UDP source port of the primary application.*

 - GenicamFeature< int64_t > * [GevPrimaryApplicationIPAddress](#)
- Returns the address of the primary application.*

 - GenicamFeature< int64_t > * [GevMCPHostPort](#)
- Controls the port to which the device must send messages. Setting this value to 0 closes the message channel.*

 - GenicamFeature< int64_t > * [GevMCDA](#)
- Controls the destination IP address for the message channel.*

 - GenicamFeature< int64_t > * [GevMCTT](#)
- Provides the transmission timeout value in milliseconds.*

- `GenicamFeature< int64_t > * GevMCRC`
Controls the number of retransmissions allowed when a message channel message times out.
- `GenicamFeature< int64_t > * GevMCSP`
This feature indicates the source port for the message channel.
- `GenicamFeature< int64_t > * GevStreamChannelSelector`
Selects the stream channel to control.
- `GenicamFeature< bool > * GevSCCFGPacketResendDestination`
Enables the alternate IP destination for stream packets resent due to a packet resend request. When True, the source IP address provided in the packet resend command packet is used. When False, the value set in the [GevSCDA\[GenicamFeature::StreamChannelSelector\]](#) feature is used.
- `GenicamFeature< bool > * GevSCCFGAllInTransmission`
Enables the selected GVSP transmitter to use the single packet per data block All-in Transmission mode.
- `GenicamFeature< bool > * GevSCCFGUnconditionalStreaming`
Enables the camera to continue to stream, for this stream channel, if its control channel is closed or regardless of the reception of any ICMP messages (such as destination unreachable messages).
- `GenicamFeature< bool > * GevSCCFGExtendedChunkData`
Enables cameras to use the extended chunk data payload type for this stream channel.
- `GenicamFeature< int64_t > * GevSCPInterfaceIndex`
Index of the logical link to use.
- `GenicamFeature< int64_t > * GevSCPHostPort`
Controls the port of the selected channel to which a GVSP transmitter must send data stream or the port from which a GVSP receiver may receive data stream. Setting this value to 0 closes the stream channel.
- `GenicamFeature< bool > * GevSCPSFireTestPacket`
Sends a test packet. When this feature is set, the device will fire one test packet.
- `GenicamFeature< bool > * GevSCPSDoNotFragment`
The state of this feature is copied into the "do not fragment" bit of IP header of each stream packet. It can be used by the application to prevent IP fragmentation of packets on the stream channel.
- `GenicamFeature< int64_t > * GevSCPSPacketSize`
This GigE Vision specific feature corresponds to [DeviceStreamChannelPacketSize](#) and should be kept in sync with it. It specifies the stream packet size, in bytes, to send on the selected channel for a GVSP transmitter or specifies the maximum packet size supported by a GVSP receiver.
- `GenicamFeature< int64_t > * GevSCPD`
Controls the delay (in GEV timestamp counter unit) to insert between each packet for this stream channel. This can be used as a crude flow-control mechanism if the application or the network infrastructure cannot keep up with the packets coming from the device.
- `GenicamFeature< int64_t > * GevSCDA`
Controls the destination IP address of the selected stream channel to which a GVSP transmitter must send data stream or the destination IP address from which a GVSP receiver may receive data stream.
- `GenicamFeature< int64_t > * GevSCSP`
Indicates the source port of the stream channel.
- `GenicamFeature< int64_t > * GevSCZoneCount`
Reports the number of zones per block transmitted on the selected stream channel.
- `GenicamFeature< int64_t > * GevSCZoneDirectionAll`
Reports the transmission direction of each zone transmitted on the selected stream channel.
- `GenicamFeature< bool > * GevSCZoneConfigurationLock`
Controls whether the selected stream channel multi-zone configuration is locked. When locked, the GVSP transmitter is not allowed to change the number of zones and their direction during block acquisition and transmission.
- `GenicamFeature< int64_t > * aPAUSEMACCtrlFramesTransmitted`
Reports the number of transmitted PAUSE frames.
- `GenicamFeature< int64_t > * aPAUSEMACCtrlFramesReceived`
Reports the number of received PAUSE frames.
- `GenicamFeature< FlisfncCameraEnum::CIConfigurationEnum > * CIConfiguration`

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, SensorDigitizationTaps, and DeviceTapGeometry do not provide enough information for interpretation of the image data provided by the camera.

- GenicamFeature< FliSfncCameraEnum::CITimeSlotsCountEnum > * [CITimeSlotsCount](#)
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.
- GenicamFeature< FliSfncCameraEnum::CxpLinkConfigurationStatusEnum > * [CxpLinkConfigurationStatus](#)
This feature indicates the current and active Link configuration used by the Device.
- GenicamFeature< FliSfncCameraEnum::CxpLinkConfigurationPreferredEnum > * [CxpLinkConfigurationPreferred](#)
Provides the Link configuration that allows the Transmitter Device to operate in its default mode.
- GenicamFeature< FliSfncCameraEnum::CxpLinkConfigurationEnum > * [CxpLinkConfiguration](#)
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.
- GenicamFeature< bool > * [CxpLinkSharingEnable](#)
Enable or disable the link sharing functionality of the device.
- GenicamFeature< int64_t > * [CxpLinkSharingSubDeviceSelector](#)
Index of the sub device used in the Link Sharing.
- GenicamFeature< FliSfncCameraEnum::CxpLinkSharingStatusEnum > * [CxpLinkSharingStatus](#)
This feature provides the data sharing status for the selected sub device.
- GenicamFeature< FliSfncCameraEnum::CxpLinkSharingSubDeviceTypeEnum > * [CxpLinkSharingSubDeviceType](#)
This feature provides the type of sub device.
- GenicamFeature< int64_t > * [CxpLinkSharingHorizontalStripeCount](#)
This feature provides the number of horizontal stripes that the device implements.
- GenicamFeature< int64_t > * [CxpLinkSharingVerticalStripeCount](#)
This feature provides the number of vertical stripes that the device implements.
- GenicamFeature< int64_t > * [CxpLinkSharingHorizontalOverlap](#)
This feature provides the number of pixel overlap in the horizontal stripes that the device implements.
- GenicamFeature< int64_t > * [CxpLinkSharingVerticalOverlap](#)
This feature provides the number of pixel overlap in the vertical stripes that the device implements.
- GenicamFeature< int64_t > * [CxpLinkSharingDuplicateStripe](#)
This feature provides the duplicate count in striped system. A non-zero value sets the number of duplicate images sent to sub-Devices.
- GenicamFeature< int64_t > * [CxpConnectionSelector](#)
Selects the CoaXPress physical connection to control.
- GenicamFeature< FliSfncCameraEnum::CxpConnectionTestModeEnum > * [CxpConnectionTestMode](#)
Enables the test mode for an individual physical connection of the Device.
- GenicamFeature< int64_t > * [CxpConnectionTestErrorCount](#)
Reports the current connection error count for test packets received by the device on the connection selected by CxpConnectionSelector.
- GenicamFeature< FliSfncCameraEnum::CxpSendReceiveSelectorEnum > * [CxpSendReceiveSelector](#)
Selects which one of the send or receive features to control.
- GenicamFeature< int64_t > * [CxpConnectionTestPacketCount](#)
Reports the current count for the test packets on the connection selected by CxpConnectionSelector.
- GenicamFeature< FliSfncCameraEnum::CxpErrorCounterSelectorEnum > * [CxpErrorCounterSelector](#)
Selects which Cxp Error Counter to read or reset.
- GenicamFeature * [CxpErrorCounterReset](#)
Resets the selected Cxp Error Counter on the connection selected by CxpConnectionSelector. The counter starts counting events immediately after the reset.
- GenicamFeature< int64_t > * [CxpErrorCounterValue](#)
Reads the current value of the selected Cxp Error Counter on the connection selected by CxpConnectionSelector.
- GenicamFeature< FliSfncCameraEnum::CxpErrorCounterStatusEnum > * [CxpErrorCounterStatus](#)

- Returns the current status of the selected Cxp Error Counter on the connection selected by CxpConnectionSelector.*
- GenicamFeature * [CxpPoCxpAuto](#)
Activate automatic control of the Power over CoaXPress (PoCXP) for the Link.
 - GenicamFeature * [CxpPoCxpTurnOff](#)
Disable Power over CoaXPress (PoCXP) for the Link.
 - GenicamFeature * [CxpPoCxpTripReset](#)
Reset the Power over CoaXPress (PoCXP) Link after an over-current trip on the Device connection(s).
 - GenicamFeature< FliSfncCameraEnum::CxpPoCxpStatusEnum > * [CxpPoCxpStatus](#)
Returns the Power over CoaXPress (PoCXP) status of the Device.

Additional Inherited Members

6.14.1 Detailed Description

This class defined all the register of an SFNC compliant camera.

Attention

C-BLUE family cameras do not implement all these registers.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 FliSfncCamera()

```
FliSfncCamera::FliSfncCamera (
    IFrameGrabberGenicam * grabber ) [explicit]
```

6.14.2.2 ~FliSfncCamera()

```
virtual FliSfncCamera::~~FliSfncCamera ( ) [virtual]
```

6.14.3 Member Data Documentation

6.14.3.1 AcquisitionAbort

```
GenicamFeature* FliSfncCamera::AcquisitionAbort
```

Aborts the Acquisition immediately. This will end the capture without completing the current Frame or waiting on a trigger. If no Acquisition is in progress, the command is ignored.

6.14.3.2 AcquisitionArm

```
GenicamFeature* FliSfncCamera::AcquisitionArm
```

Arms the device before an AcquisitionStart command. This optional command validates all the current features for consistency and prepares the device for a fast start of the Acquisition.

6.14.3.3 AcquisitionBurstFrameCount

```
GenicamFeature<int64_t>* FliSfncCamera::AcquisitionBurstFrameCount
```

Number of frames to acquire for each FrameBurstStart trigger.

6.14.3.4 AcquisitionFrameCount

```
GenicamFeature<int64_t>* FliSfncCamera::AcquisitionFrameCount
```

Number of frames to acquire in MultiFrame Acquisition mode.

6.14.3.5 AcquisitionFrameRate

```
GenicamFeature<double>* FliSfncCamera::AcquisitionFrameRate
```

Controls the acquisition rate (in Hertz) at which the frames are captured.

6.14.3.6 AcquisitionFrameRateEnable

```
GenicamFeature<bool>* FliSfncCamera::AcquisitionFrameRateEnable
```

Controls if the AcquisitionFrameRate feature is writable and used to control the acquisition rate. Otherwise, the acquisition rate is implicitly controlled by the combination of other features like ExposureTime, etc...

6.14.3.7 AcquisitionLineRate

```
GenicamFeature<double>* FliSfncCamera::AcquisitionLineRate
```

Controls the rate (in Hertz) at which the Lines in a Frame are captured.

6.14.3.8 AcquisitionLineRateEnable

```
GenicamFeature<bool>* FliSfncCamera::AcquisitionLineRateEnable
```

Controls if the AcquisitionLineRate feature is writable and used to control the acquisition rate. Otherwise, the acquisition rate is implicitly controlled by the combination of other features like ExposureTime, etc...

6.14.3.9 AcquisitionMode

```
GenicamFeature<FliSfncCameraEnum::AcquisitionModeEnum>* FliSfncCamera::AcquisitionMode
```

Sets the acquisition mode of the device. It defines mainly the number of frames to capture during an acquisition and the way the acquisition stops.

6.14.3.10 AcquisitionStart

```
GenicamFeature* FliSfncCamera::AcquisitionStart
```

Starts the Acquisition of the device. The number of frames captured is specified by AcquisitionMode.

6.14.3.11 AcquisitionStatus

```
GenicamFeature<bool>* FliSfncCamera::AcquisitionStatus
```

Reads the state of the internal acquisition signal selected using AcquisitionStatusSelector.

6.14.3.12 AcquisitionStatusSelector

```
GenicamFeature<FliSfncCameraEnum::AcquisitionStatusSelectorEnum>* FliSfncCamera::Acquisition↔  
StatusSelector
```

Selects the internal acquisition signal to read using AcquisitionStatus.

6.14.3.13 AcquisitionStop

```
GenicamFeature* FliSfncCamera::AcquisitionStop
```

Stops the Acquisition of the device at the end of the current Frame. It is mainly used when AcquisitionMode is Continuous but can be used in any acquisition mode.

6.14.3.14 AcquisitionStopMode

GenicamFeature<FliSfncCameraEnum::AcquisitionStopModeEnum>* FliSfncCamera::AcquisitionStopMode

Controls how the AcquisitionStop command and the acquisition stopped using a trigger (e.g. AcquisitionActive, FrameBurstActive, FrameActive or FrameEnd trigger), ends an ongoing frame. This feature is mainly used in Linescan devices where each line in a frame is acquired sequentially.

6.14.3.15 ActionDeviceKey

GenicamFeature<int64_t>* FliSfncCamera::ActionDeviceKey

Provides the device key that allows the device to check the validity of action commands. The device internal assertion of an action signal is only authorized if the ActionDeviceKey and the action device key value in the protocol message are equal.

6.14.3.16 ActionGroupKey

GenicamFeature<int64_t>* FliSfncCamera::ActionGroupKey

Provides the key that the device will use to validate the action on reception of the action protocol message.

6.14.3.17 ActionGroupMask

GenicamFeature<int64_t>* FliSfncCamera::ActionGroupMask

Provides the mask that the device will use to validate the action on reception of the action protocol message.

6.14.3.18 ActionQueueSize

GenicamFeature<int64_t>* FliSfncCamera::ActionQueueSize

Indicates the size of the scheduled action commands queue. This number represents the maximum number of scheduled action commands that can be pending at a given point in time.

6.14.3.19 ActionSelector

GenicamFeature<int64_t>* FliSfncCamera::ActionSelector

Selects to which Action Signal further Action settings apply.

6.14.3.20 ActionUnconditionalMode

GenicamFeature<FliSfncCameraEnum::ActionUnconditionalModeEnum>* FliSfncCamera::ActionUnconditionalMode

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

6.14.3.21 aPAUSEMACCtrlFramesReceived

GenicamFeature<int64_t>* FliSfncCamera::aPAUSEMACCtrlFramesReceived

Reports the number of received PAUSE frames.

6.14.3.22 aPAUSEMACCtrlFramesTransmitted

GenicamFeature<int64_t>* FliSfncCamera::aPAUSEMACCtrlFramesTransmitted

Reports the number of transmitted PAUSE frames.

6.14.3.23 BalanceRatio

GenicamFeature<double>* FliSfncCamera::BalanceRatio

Controls ratio of the selected color component to a reference color component. It is used for white balancing.

6.14.3.24 BalanceRatioSelector

GenicamFeature<FliSfncCameraEnum::BalanceRatioSelectorEnum>* FliSfncCamera::BalanceRatioSelector

Selects which Balance ratio to control.

6.14.3.25 BalanceWhiteAuto

GenicamFeature<FliSfncCameraEnum::BalanceWhiteAutoEnum>* FliSfncCamera::BalanceWhiteAuto

Controls the mode for automatic white balancing between the color channels. The white balancing ratios are automatically adjusted.

6.14.3.26 BinningHorizontal

```
GenicamFeature<int64_t>* FliSfncCamera::BinningHorizontal
```

Number of horizontal photo-sensitive cells to combine together. This reduces the horizontal resolution (width) of the image.

6.14.3.27 BinningHorizontalMode

```
GenicamFeature<FliSfncCameraEnum::BinningHorizontalModeEnum>* FliSfncCamera::BinningHorizontalMode
```

Sets the mode to use to combine horizontal photo-sensitive cells together when BinningHorizontal is used.

6.14.3.28 BinningSelector

```
GenicamFeature<FliSfncCameraEnum::BinningSelectorEnum>* FliSfncCamera::BinningSelector
```

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

6.14.3.29 BinningVertical

```
GenicamFeature<int64_t>* FliSfncCamera::BinningVertical
```

Number of vertical photo-sensitive cells to combine together. This reduces the vertical resolution (height) of the image.

6.14.3.30 BinningVerticalMode

```
GenicamFeature<FliSfncCameraEnum::BinningVerticalModeEnum>* FliSfncCamera::BinningVerticalMode
```

Sets the mode to use to combine vertical photo-sensitive cells together when BinningVertical is used.

6.14.3.31 BlackLevel

```
GenicamFeature<double>* FliSfncCamera::BlackLevel
```

Controls the analog black level as an absolute physical value. This represents a DC offset applied to the video signal.

6.14.3.32 BlackLevelAuto

```
GenicamFeature<FliSfncCameraEnum::BlackLevelAutoEnum>* FliSfncCamera::BlackLevelAuto
```

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

6.14.3.33 BlackLevelAutoBalance

```
GenicamFeature<FliSfncCameraEnum::BlackLevelAutoBalanceEnum>* FliSfncCamera::BlackLevelAuto↔  
Balance
```

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.

6.14.3.34 BlackLevelSelector

```
GenicamFeature<FliSfncCameraEnum::BlackLevelSelectorEnum>* FliSfncCamera::BlackLevelSelector
```

Selects which Black Level is controlled by the various Black Level features.

6.14.3.35 CameraPresence

```
GenicamFeature<bool>* FliSfncCamera::CameraPresence
```

6.14.3.36 ChunkBinningHorizontal

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkBinningHorizontal
```

Number of horizontal photo-sensitive cells combined together.

6.14.3.37 ChunkBinningVertical

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkBinningVertical
```

Number of vertical photo-sensitive cells combined together.

6.14.3.38 ChunkBlackLevel

```
GenicamFeature<double>* FliSfncCamera::ChunkBlackLevel
```

Returns the black level used to capture the image included in the payload.

6.14.3.39 ChunkBlackLevelSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkBlackLevelSelectorEnum>* FliSfncCamera::ChunkBlackLevelSelector
```

Selects which Black Level to return. Possible values are:

6.14.3.40 ChunkComponentID

```
GenicamFeature<FliSfncCameraEnum::ChunkComponentIDEnum>* FliSfncCamera::ChunkComponentID
```

Returns the Identifier of the selected Component. This can be used to identify the image component type of a multi-component payload.

6.14.3.41 ChunkComponentIDValue

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkComponentIDValue
```

Returns a unique Identifier value that corresponds to the selected chunk Component.

6.14.3.42 ChunkComponentSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkComponentSelectorEnum>* FliSfncCamera::ChunkComponentSelector
```

Selects the Component from which to retrieve data from.

6.14.3.43 ChunkCounterSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkCounterSelectorEnum>* FliSfncCamera::ChunkCounterSelector
```

Selects which counter to retrieve data from.

6.14.3.44 ChunkCounterValue

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkCounterValue
```

Returns the value of the selected Chunk counter at the time of the FrameStart event.

6.14.3.45 ChunkDecimationHorizontal

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkDecimationHorizontal
```

Horizontal sub-sampling of the image.

6.14.3.46 ChunkDecimationVertical

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkDecimationVertical
```

Vertical sub-sampling of the image.

6.14.3.47 ChunkEnable

```
GenicamFeature<bool>* FliSfncCamera::ChunkEnable
```

Enables the inclusion of the selected Chunk data in the payload of the image.

6.14.3.48 ChunkEncoderSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkEncoderSelectorEnum>* FliSfncCamera::ChunkEncoderSelector
```

Selects which Encoder to retrieve data from.

6.14.3.49 ChunkEncoderStatus

```
GenicamFeature<FliSfncCameraEnum::ChunkEncoderStatusEnum>* FliSfncCamera::ChunkEncoderStatus
```

Returns the motion status of the selected encoder.

6.14.3.50 ChunkEncoderValue

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkEncoderValue
```

Returns the counter's value of the selected Encoder at the time of the FrameStart in area scan mode or the counter's value at the time of the LineStart selected by ChunkScanLineSelector in Linescan mode.

6.14.3.51 ChunkExposureTime

```
GenicamFeature<double>* FliSfncCamera::ChunkExposureTime
```

Returns the exposure time used to capture the image.

6.14.3.52 ChunkExposureTimeSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkExposureTimeSelectorEnum>* FliSfncCamera::ChunkExposure←  
TimeSelector
```

Selects which exposure time is read by the ChunkExposureTime feature.

6.14.3.53 ChunkFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkFrameID
```

Returns the unique Identifier of the frame (or image) included in the payload.

6.14.3.54 ChunkGain

```
GenicamFeature<double>* FliSfncCamera::ChunkGain
```

Returns the gain used to capture the image.

6.14.3.55 ChunkGainSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkGainSelectorEnum>* FliSfncCamera::ChunkGainSelector
```

Selects which Gain to return.

6.14.3.56 ChunkGroupID

```
GenicamFeature<FliSfncCameraEnum::ChunkGroupIDEnum>* FliSfncCamera::ChunkGroupID
```

Returns a unique Identifier corresponding to the selected Group of components. This can be used to identify the component Group of a multi-group payload.

6.14.3.57 ChunkGroupIDValue

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkGroupIDValue
```

Returns a unique Identifier value that corresponds to the Group of Components of the selected chunk Component.

6.14.3.58 ChunkGroupSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkGroupSelectorEnum>* FliSfncCamera::ChunkGroupSelector
```

Selects the component Group from which to retrieve data from.

6.14.3.59 ChunkHeight

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkHeight
```

Returns the Height of the image included in the payload.

6.14.3.60 ChunkLinePitch

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkLinePitch
```

Returns the LinePitch of the image included in the payload.

6.14.3.61 ChunkLineStatusAll

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkLineStatusAll
```

Returns the status of all the I/O lines at the time of the FrameStart internal event.

6.14.3.62 ChunkModeActive

```
GenicamFeature<bool>* FliSfncCamera::ChunkModeActive
```

Activates the inclusion of Chunk data in the transmitted payload.

6.14.3.63 ChunkOffsetX

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkOffsetX
```

Returns the OffsetX of the image included in the payload.

6.14.3.64 ChunkOffsetY

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkOffsetY
```

Returns the OffsetY of the image included in the payload.

6.14.3.65 ChunkPixelDynamicRangeMax

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkPixelDynamicRangeMax
```

Returns the maximum value of dynamic range of the image included in the payload.

6.14.3.66 ChunkPixelDynamicRangeMin

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkPixelDynamicRangeMin
```

Returns the minimum value of dynamic range of the image included in the payload.

6.14.3.67 ChunkPixelFormat

```
GenicamFeature<FliSfncCameraEnum::ChunkPixelFormatEnum>* FliSfncCamera::ChunkPixelFormat
```

Returns the PixelFormat of the image included in the payload.

6.14.3.68 ChunkRegionID

```
GenicamFeature<FliSfncCameraEnum::ChunkRegionIDEnum>* FliSfncCamera::ChunkRegionID
```

Returns the Identifier of Region that the image comes from.

6.14.3.69 ChunkRegionIDValue

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkRegionIDValue
```

Returns the unique integer Identifier value of the Region that the image comes from.

6.14.3.70 ChunkRegionSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkRegionSelectorEnum>* FliSfncCamera::ChunkRegionSelector
```

Selects which Region to retrieve data from.

6.14.3.71 ChunkReverseX

```
GenicamFeature<bool>* FliSfncCamera::ChunkReverseX
```

Flip horizontal of the image sent by the device.

6.14.3.72 ChunkReverseY

```
GenicamFeature<bool>* FliSfncCamera::ChunkReverseY
```

Flip vertically of the image sent by the device.

6.14.3.73 ChunkScan3dAxisMax

```
GenicamFeature<double>* FliSfncCamera::ChunkScan3dAxisMax
```

Returns the Maximum Axis value for the selected coordinate axis of the image included in the payload.

6.14.3.74 ChunkScan3dAxisMin

```
GenicamFeature<double>* FliSfncCamera::ChunkScan3dAxisMin
```

Returns the Minimum Axis value for the selected coordinate axis of the image included in the payload.

6.14.3.75 ChunkScan3dBaseline

```
GenicamFeature<double>* FliSfncCamera::ChunkScan3dBaseline
```

Returns the baseline as the physical distance of two cameras in a stereo camera setup. The value of this feature can be used for 3D reconstruction from disparity images. In this case, the unit of the 3D coordinates corresponds to the unit of the baseline.

6.14.3.76 ChunkScan3dCoordinateOffset

```
GenicamFeature<double>* FliSfncCamera::ChunkScan3dCoordinateOffset
```

Returns the Offset for the selected coordinate axis of the image included in the payload.

6.14.3.77 ChunkScan3dCoordinateReferenceSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkScan3dCoordinateReferenceSelectorEnum>* FliSfncCamera↔  
::ChunkScan3dCoordinateReferenceSelector
```

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

6.14.3.78 ChunkScan3dCoordinateReferenceValue

```
GenicamFeature<double>* FliSfncCamera::ChunkScan3dCoordinateReferenceValue
```

Returns the value of a position or pose coordinate for the anchor or transformed coordinate systems relative to the reference point.

6.14.3.79 ChunkScan3dCoordinateScale

```
GenicamFeature<double>* FliSfncCamera::ChunkScan3dCoordinateScale
```

Returns the Scale for the selected coordinate axis of the image included in the payload.

6.14.3.80 ChunkScan3dCoordinateSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkScan3dCoordinateSelectorEnum>* FliSfncCamera::ChunkScan3dCoordinateSelector
```

Selects which Coordinate to retrieve data from.

6.14.3.81 ChunkScan3dCoordinateSystem

```
GenicamFeature<FliSfncCameraEnum::ChunkScan3dCoordinateSystemEnum>* FliSfncCamera::ChunkScan3dCoordinateSystem
```

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

6.14.3.82 ChunkScan3dCoordinateSystemReference

```
GenicamFeature<FliSfncCameraEnum::ChunkScan3dCoordinateSystemReferenceEnum>* FliSfncCamera::ChunkScan3dCoordinateSystemReference
```

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

6.14.3.83 ChunkScan3dCoordinateTransformSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkScan3dCoordinateTransformSelectorEnum>* FliSfncCamera::ChunkScan3dCoordinateTransformSelector
```

Selector for transform values.

6.14.3.84 ChunkScan3dDistanceUnit

```
GenicamFeature<FliSfncCameraEnum::ChunkScan3dDistanceUnitEnum>* FliSfncCamera::ChunkScan3dDistanceUnit
```

Returns the Distance Unit of the payload image.

6.14.3.85 ChunkScan3dFocalLength

```
GenicamFeature<double>* FliSfncCamera::ChunkScan3dFocalLength
```

Returns the focal length of the camera in pixel. The focal length depends on the selected region. The value of this feature takes into account horizontal binning, decimation, or any other function changing the image resolution.

6.14.3.86 ChunkScan3dInvalidDataFlag

```
GenicamFeature<bool>* FliSfncCamera::ChunkScan3dInvalidDataFlag
```

Returns if a specific non-valid data flag is used in the data stream.

6.14.3.87 ChunkScan3dInvalidDataValue

```
GenicamFeature<double>* FliSfncCamera::ChunkScan3dInvalidDataValue
```

Returns the Invalid Data Value used for the image included in the payload.

6.14.3.88 ChunkScan3dOutputMode

```
GenicamFeature<FliSfncCameraEnum::ChunkScan3dOutputModeEnum>* FliSfncCamera::ChunkScan3d↵  
OutputMode
```

Returns the Calibrated Mode of the payload image.

6.14.3.89 ChunkScan3dPrincipalPointU

```
GenicamFeature<double>* FliSfncCamera::ChunkScan3dPrincipalPointU
```

Returns the value of this feature gives the horizontal position of the principal point, relative to the region origin, i.e. OffsetX. The value of this feature takes into account horizontal binning, decimation, or any other function changing the image resolution.

6.14.3.90 ChunkScan3dPrincipalPointV

```
GenicamFeature<double>* FliSfncCamera::ChunkScan3dPrincipalPointV
```

Returns the value of this feature gives the vertical position of the principal point, relative to the region origin, i.e. OffsetY. The value of this feature takes into account vertical binning, decimation, or any other function changing the image resolution.

6.14.3.91 ChunkScan3dTransformValue

```
GenicamFeature<double>* FliSfncCamera::ChunkScan3dTransformValue
```

Returns the transform value.

6.14.3.92 ChunkScanLineSelector

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkScanLineSelector
```

Index for vector representation of one chunk value per line in an image.

6.14.3.93 ChunkSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkSelectorEnum>* FliSfncCamera::ChunkSelector
```

Selects which Chunk to enable or control.

6.14.3.94 ChunkSequencerSetActive

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkSequencerSetActive
```

Return the index of the active set of the running sequencer included in the payload.

6.14.3.95 ChunkSourceID

```
GenicamFeature<FliSfncCameraEnum::ChunkSourceIDEnum>* FliSfncCamera::ChunkSourceID
```

Returns the Identifier of Source that the image comes from.

6.14.3.96 ChunkSourceIDValue

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkSourceIDValue
```

Returns the unique integer Identifier value of the Source that the image comes from.

6.14.3.97 ChunkSourceSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkSourceSelectorEnum>* FliSfncCamera::ChunkSourceSelector
```

Selects which Source to retrieve data from.

6.14.3.98 ChunkStreamChannelID

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkStreamChannelID
```

Returns identifier of the stream channel used to carry the block.

6.14.3.99 ChunkTimerSelector

```
GenicamFeature<FliSfncCameraEnum::ChunkTimerSelectorEnum>* FliSfncCamera::ChunkTimerSelector
```

Selects which Timer to retrieve data from.

6.14.3.100 ChunkTimerValue

```
GenicamFeature<double>* FliSfncCamera::ChunkTimerValue
```

Returns the value of the selected Timer at the time of the FrameStart internal event.

6.14.3.101 ChunkTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkTimestamp
```

Returns the Timestamp of the image included in the payload at the time of the FrameStart internal event.

6.14.3.102 ChunkTimestampLatchValue

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkTimestampLatchValue
```

Returns the last Timestamp latched with the TimestampLatch command.

6.14.3.103 ChunkTransferBlockID

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkTransferBlockID
```

Returns the unique identifier of the transfer block used to transport the payload.

6.14.3.104 ChunkTransferQueueCurrentBlockCount

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkTransferQueueCurrentBlockCount
```

Returns the current number of blocks in the transfer queue.

6.14.3.105 ChunkTransferStreamID

```
GenicamFeature<FliSfncCameraEnum::ChunkTransferStreamIDEnum>* FliSfncCamera::ChunkTransfer↔  
StreamID
```

Returns identifier of the stream that generated this block.

6.14.3.106 ChunkWidth

```
GenicamFeature<int64_t>* FliSfncCamera::ChunkWidth
```

Returns the Width of the image included in the payload.

6.14.3.107 ChunkXMLEnable

```
GenicamFeature<bool>* FliSfncCamera::ChunkXMLEnable
```

Activates the inclusion of the GenICam XML necessary to the chunk parser to decode all the Chunk data included in the transmitted payload.

6.14.3.108 ClConfiguration

```
GenicamFeature<FliSfncCameraEnum::ClConfigurationEnum>* FliSfncCamera::ClConfiguration
```

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.

6.14.3.109 ClTimeSlotsCount

```
GenicamFeature<FliSfncCameraEnum::ClTimeSlotsCountEnum>* FliSfncCamera::ClTimeSlotsCount
```

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.

6.14.3.110 ColorTransformationEnable

```
GenicamFeature<bool>* FliSfncCamera::ColorTransformationEnable
```

Activates the selected Color Transformation module.

6.14.3.111 ColorTransformationSelector

```
GenicamFeature<FliSfncCameraEnum::ColorTransformationSelectorEnum>* FliSfncCamera::ColorTransformationSelector
```

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

6.14.3.112 ColorTransformationValue

```
GenicamFeature<double>* FliSfncCamera::ColorTransformationValue
```

Represents the value of the selected Gain factor or Offset inside the Transformation matrix.

6.14.3.113 ColorTransformationValueSelector

```
GenicamFeature<FliSfncCameraEnum::ColorTransformationValueSelectorEnum>* FliSfncCamera::ColorTransformationValueSelector
```

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

6.14.3.114 ComponentEnable

```
GenicamFeature<bool>* FliSfncCamera::ComponentEnable
```

Controls if the selected component streaming is active.

6.14.3.115 ComponentIDValue

```
GenicamFeature<int64_t>* FliSfncCamera::ComponentIDValue
```

Returns a unique Identifier value that corresponds to type of the component selected by ComponentSelector.

6.14.3.116 ComponentSelector

```
GenicamFeature<FliSfncCameraEnum::ComponentSelectorEnum>* FliSfncCamera::ComponentSelector
```

Selects a component to activate/deactivate its data streaming.

6.14.3.117 CounterDuration

```
GenicamFeature<int64_t>* FliSfncCamera::CounterDuration
```

Sets the duration (or number of events) before the CounterEnd event is generated.

6.14.3.118 CounterEventActivation

```
GenicamFeature<FliSfncCameraEnum::CounterEventActivationEnum>* FliSfncCamera::CounterEvent←  
Activation
```

Selects the Activation mode Event Source signal.

6.14.3.119 CounterEventSource

```
GenicamFeature<FliSfncCameraEnum::CounterEventSourceEnum>* FliSfncCamera::CounterEventSource
```

Select the events that will be the source to increment the Counter.

6.14.3.120 CounterReset

```
GenicamFeature* FliSfncCamera::CounterReset
```

Does a software reset of the selected Counter and starts it. The counter starts counting events immediately after the reset unless a Counter trigger is active. CounterReset can be used to reset the Counter independently from the CounterResetSource. To disable the counter temporarily, set CounterEventSource to Off.

6.14.3.121 CounterResetActivation

```
GenicamFeature<FliSfncCameraEnum::CounterResetActivationEnum>* FliSfncCamera::CounterReset←  
Activation
```

Selects the Activation mode of the Counter Reset Source signal.

6.14.3.122 CounterResetSource

```
GenicamFeature<FliSfncCameraEnum::CounterResetSourceEnum>* FliSfncCamera::CounterResetSource
```

Selects the signals that will be the source to reset the Counter.

6.14.3.123 CounterSelector

```
GenicamFeature<FliSfncCameraEnum::CounterSelectorEnum>* FliSfncCamera::CounterSelector
```

Selects which Counter to configure.

6.14.3.124 CounterStatus

```
GenicamFeature<FliSfncCameraEnum::CounterStatusEnum>* FliSfncCamera::CounterStatus
```

Returns the current status of the Counter.

6.14.3.125 CounterTriggerActivation

```
GenicamFeature<FliSfncCameraEnum::CounterTriggerActivationEnum>* FliSfncCamera::Counter↔  
TriggerActivation
```

Selects the activation mode of the trigger to start the Counter.

6.14.3.126 CounterTriggerSource

```
GenicamFeature<FliSfncCameraEnum::CounterTriggerSourceEnum>* FliSfncCamera::CounterTrigger↔  
Source
```

Selects the source to start the Counter.

6.14.3.127 CounterValue

```
GenicamFeature<int64_t>* FliSfncCamera::CounterValue
```

Reads or writes the current value of the selected Counter.

6.14.3.128 CounterValueAtReset

```
GenicamFeature<int64_t>* FliSfncCamera::CounterValueAtReset
```

Reads the value of the selected Counter when it was reset by a trigger or by an explicit CounterReset command.

6.14.3.129 CxpConnectionSelector

```
GenicamFeature<int64_t>* FliSfncCamera::CxpConnectionSelector
```

Selects the CoaXPress physical connection to control.

6.14.3.130 CxpConnectionTestErrorCount

```
GenicamFeature<int64_t>* FliSfncCamera::CxpConnectionTestErrorCount
```

Reports the current connection error count for test packets received by the device on the connection selected by CxpConnectionSelector.

6.14.3.131 CxpConnectionTestMode

```
GenicamFeature<FliSfncCameraEnum::CxpConnectionTestModeEnum>* FliSfncCamera::CxpConnection↔  
TestMode
```

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

6.14.3.132 CxpConnectionTestPacketCount

```
GenicamFeature<int64_t>* FliSfncCamera::CxpConnectionTestPacketCount
```

Reports the current count for the test packets on the connection selected by CxpConnectionSelector.

6.14.3.133 CxpErrorCounterReset

```
GenicamFeature* FliSfncCamera::CxpErrorCounterReset
```

Resets the selected Cxp Error Counter on the connection selected by CxpConnectionSelector. The counter starts counting events immediately after the reset.

6.14.3.134 CxpErrorCounterSelector

```
GenicamFeature<FliSfncCameraEnum::CxpErrorCounterSelectorEnum>* FliSfncCamera::CxpErrorCounterSelector
```

Selects which Cxp Error Counter to read or reset.

6.14.3.135 CxpErrorCounterStatus

```
GenicamFeature<FliSfncCameraEnum::CxpErrorCounterStatusEnum>* FliSfncCamera::CxpErrorCounterStatus
```

Returns the current status of the selected Cxp Error Counter on the connection selected by CxpConnectionSelector.

6.14.3.136 CxpErrorCounterValue

```
GenicamFeature<int64_t>* FliSfncCamera::CxpErrorCounterValue
```

Reads the current value of the selected Cxp Error Counter on the connection selected by CxpConnectionSelector.

6.14.3.137 CxpFirstLineTriggerWithFrameStart

```
GenicamFeature<bool>* FliSfncCamera::CxpFirstLineTriggerWithFrameStart
```

Specifies if a FrameStart trigger also triggers the first LineStart at the same time.

6.14.3.138 CxpLinkConfiguration

```
GenicamFeature<FliSfncCameraEnum::CxpLinkConfigurationEnum>* FliSfncCamera::CxpLinkConfiguration
```

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.

6.14.3.139 CxpLinkConfigurationPreferred

```
GenicamFeature<FliSfncCameraEnum::CxpLinkConfigurationPreferredEnum>* FliSfncCamera::CxpLinkConfigurationPreferred
```

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

6.14.3.140 CxpLinkConfigurationStatus

```
GenicamFeature<FliSfncCameraEnum::CxpLinkConfigurationStatusEnum>* FliSfncCamera::CxpLink↔  
ConfigurationStatus
```

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

6.14.3.141 CxpLinkSharingDuplicateStripe

```
GenicamFeature<int64_t>* FliSfncCamera::CxpLinkSharingDuplicateStripe
```

This feature provides the duplicate count in striped system. A non-zero value sets the number of duplicate images sent to sub-Devices.

6.14.3.142 CxpLinkSharingEnable

```
GenicamFeature<bool>* FliSfncCamera::CxpLinkSharingEnable
```

Enable or disable the link sharing functionality of the device.

6.14.3.143 CxpLinkSharingHorizontalOverlap

```
GenicamFeature<int64_t>* FliSfncCamera::CxpLinkSharingHorizontalOverlap
```

This feature provides the number of pixel overlap in the horizontal stripes that the device implements.

6.14.3.144 CxpLinkSharingHorizontalStripeCount

```
GenicamFeature<int64_t>* FliSfncCamera::CxpLinkSharingHorizontalStripeCount
```

This feature provides the number of horizontal stripes that the device implements.

6.14.3.145 CxpLinkSharingStatus

```
GenicamFeature<FliSfncCameraEnum::CxpLinkSharingStatusEnum>* FliSfncCamera::CxpLinkSharing↔  
Status
```

This feature provides the data sharing status for the selected sub device.

6.14.3.146 CxpLinkSharingSubDeviceSelector

```
GenicamFeature<int64_t>* FliSfncCamera::CxpLinkSharingSubDeviceSelector
```

Index of the sub device used in the Link Sharing.

6.14.3.147 CxpLinkSharingSubDeviceType

```
GenicamFeature<FliSfncCameraEnum::CxpLinkSharingSubDeviceTypeEnum>* FliSfncCamera::CxpLinkSharingSubDeviceType
```

This feature provides the type of sub device.

6.14.3.148 CxpLinkSharingVerticalOverlap

```
GenicamFeature<int64_t>* FliSfncCamera::CxpLinkSharingVerticalOverlap
```

This feature provides the number of pixel overlap in the vertical stripes that the device implements.

6.14.3.149 CxpLinkSharingVerticalStripeCount

```
GenicamFeature<int64_t>* FliSfncCamera::CxpLinkSharingVerticalStripeCount
```

This feature provides the number of vertical stripes that the device implements.

6.14.3.150 CxpPoCxpAuto

```
GenicamFeature* FliSfncCamera::CxpPoCxpAuto
```

Activate automatic control of the Power over CoaXPress (PoCXP) for the Link.

6.14.3.151 CxpPoCxpStatus

```
GenicamFeature<FliSfncCameraEnum::CxpPoCxpStatusEnum>* FliSfncCamera::CxpPoCxpStatus
```

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

6.14.3.152 CxpPoCxpTripReset

```
GenicamFeature* FliSfncCamera::CxpPoCxpTripReset
```

Reset the Power over CoaXPress (PoCXP) Link after an over-current trip on the Device connection(s).

6.14.3.153 CxpPoCxpTurnOff

```
GenicamFeature* FliSfncCamera::CxpPoCxpTurnOff
```

Disable Power over CoaXPress (PoCXP) for the Link.

6.14.3.154 CxpSendReceiveSelector

```
GenicamFeature<FliSfncCameraEnum::CxpSendReceiveSelectorEnum>* FliSfncCamera::CxpSendReceive↔  
Selector
```

Selects which one of the send or receive features to control.

6.14.3.155 DecimationHorizontal

```
GenicamFeature<int64_t>* FliSfncCamera::DecimationHorizontal
```

Horizontal sub-sampling of the image. This reduces the horizontal resolution (width) of the image by the specified horizontal decimation factor.

6.14.3.156 DecimationHorizontalMode

```
GenicamFeature<FliSfncCameraEnum::DecimationHorizontalModeEnum>* FliSfncCamera::Decimation↔  
HorizontalMode
```

Sets the mode used to reduce the horizontal resolution when DecimationHorizontal is used.

6.14.3.157 DecimationVertical

```
GenicamFeature<int64_t>* FliSfncCamera::DecimationVertical
```

Vertical sub-sampling of the image. This reduces the vertical resolution (height) of the image by the specified vertical decimation factor.

6.14.3.158 DecimationVerticalMode

```
GenicamFeature<FliSfncCameraEnum::DecimationVerticalModeEnum>* FliSfncCamera::DecimationVerticalMode
```

Sets the mode used to reduce the Vertical resolution when DecimationVertical is used.

6.14.3.159 Deinterlacing

```
GenicamFeature<FliSfncCameraEnum::DeinterlacingEnum>* FliSfncCamera::Deinterlacing
```

Controls how the device performs de-interlacing.

6.14.3.160 DeviceCharacterSet

```
GenicamFeature<FliSfncCameraEnum::DeviceCharacterSetEnum>* FliSfncCamera::DeviceCharacterSet
```

Character set used by the strings of the device.

6.14.3.161 DeviceClockFrequency

```
GenicamFeature<double>* FliSfncCamera::DeviceClockFrequency
```

Returns the frequency of the selected Clock.

6.14.3.162 DeviceClockSelector

```
GenicamFeature<FliSfncCameraEnum::DeviceClockSelectorEnum>* FliSfncCamera::DeviceClockSelector
```

Selects the clock frequency to access from the device.

6.14.3.163 DeviceConnectionSelector

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceConnectionSelector
```

Selects which Connection of the device to control.

6.14.3.164 DeviceConnectionSpeed

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceConnectionSpeed
```

Indicates the speed of transmission of the specified Connection.

6.14.3.165 DeviceConnectionStatus

```
GenicamFeature<FliSfncCameraEnum::DeviceConnectionStatusEnum>* FliSfncCamera::DeviceConnection↔  
Status
```

Indicates the status of the specified Connection.

6.14.3.166 DeviceEventChannelCount

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceEventChannelCount
```

Indicates the number of event channels supported by the device.

6.14.3.167 DeviceFamilyName

```
GenicamFeature<std::string>* FliSfncCamera::DeviceFamilyName
```

Identifier of the product family of the device.

6.14.3.168 DeviceFeaturePersistenceEnd

```
GenicamFeature* FliSfncCamera::DeviceFeaturePersistenceEnd
```

Indicate to the device the end of feature persistence.

6.14.3.169 DeviceFeaturePersistenceStart

```
GenicamFeature* FliSfncCamera::DeviceFeaturePersistenceStart
```

Indicate to the device and GenICam XML to get ready for persisting of all streamable features.

6.14.3.170 DeviceFirmwareVersion

```
GenicamFeature<std::string>* FliSfncCamera::DeviceFirmwareVersion
```

Version of the firmware in the device.

6.14.3.171 DeviceGenCPVersionMajor

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceGenCPVersionMajor
```

Major version of the GenCP protocol supported by the device.

6.14.3.172 DeviceGenCPVersionMinor

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceGenCPVersionMinor
```

Minor version of the GenCP protocol supported by the device.

6.14.3.173 DeviceIndicatorMode

```
GenicamFeature<FliSfncCameraEnum::DeviceIndicatorModeEnum>* FliSfncCamera::DeviceIndicatorMode
```

Controls the behavior of the indicators (such as LEDs) showing the status of the Device.

6.14.3.174 DeviceLinkCommandTimeout

```
GenicamFeature<double>* FliSfncCamera::DeviceLinkCommandTimeout
```

Indicates the command timeout of the specified Link. This corresponds to the maximum response time of the device for a command sent on that link.

6.14.3.175 DeviceLinkConnectionCount

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceLinkConnectionCount
```

Returns the number of physical connection of the device used by a particular Link.

6.14.3.176 DeviceLinkHeartbeatMode

```
GenicamFeature<FliSfncCameraEnum::DeviceLinkHeartbeatModeEnum>* FliSfncCamera::DeviceLink↔  
HeartbeatMode
```

Activate or deactivate the Link's heartbeat.

6.14.3.177 DeviceLinkHeartbeatTimeout

```
GenicamFeature<double>* FliSfncCamera::DeviceLinkHeartbeatTimeout
```

Controls the current heartbeat timeout of the specific Link.

6.14.3.178 DeviceLinkSelector

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceLinkSelector
```

Selects which Link of the device to control.

6.14.3.179 DeviceLinkSpeed

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceLinkSpeed
```

Indicates the speed of transmission negotiated on the specified Link.

6.14.3.180 DeviceLinkThroughputLimit

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceLinkThroughputLimit
```

Limits the maximum bandwidth of the data that will be streamed out by the device on the selected Link. If necessary, delays will be uniformly inserted between transport layer packets in order to control the peak bandwidth.

6.14.3.181 DeviceLinkThroughputLimitMode

```
GenicamFeature<FliSfncCameraEnum::DeviceLinkThroughputLimitModeEnum>* FliSfncCamera::Device↔  
LinkThroughputLimitMode
```

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.

6.14.3.182 DeviceManifestEntrySelector

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceManifestEntrySelector
```

Selects the manifest entry to reference.

6.14.3.183 DeviceManifestPrimaryURL

```
GenicamFeature<std::string>* FliSfncCamera::DeviceManifestPrimaryURL
```

Indicates the first URL to the GenICam XML device description file of the selected manifest entry.

6.14.3.184 DeviceManifestSchemaMajorVersion

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceManifestSchemaMajorVersion
```

Indicates the major version number of the schema file of the selected manifest entry.

6.14.3.185 DeviceManifestSchemaMinorVersion

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceManifestSchemaMinorVersion
```

Indicates the minor version number of the schema file of the selected manifest entry.

6.14.3.186 DeviceManifestSecondaryURL

```
GenicamFeature<std::string>* FliSfncCamera::DeviceManifestSecondaryURL
```

Indicates the second URL to the GenICam XML device description file of the selected manifest entry.

6.14.3.187 DeviceManifestXMLMajorVersion

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceManifestXMLMajorVersion
```

Indicates the major version number of the GenICam XML file of the selected manifest entry.

6.14.3.188 DeviceManifestXMLMinorVersion

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceManifestXMLMinorVersion
```

Indicates the minor version number of the GenICam XML file of the selected manifest entry.

6.14.3.189 DeviceManifestXMLSubMinorVersion

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceManifestXMLSubMinorVersion
```

Indicates the subminor version number of the GenICam XML file of the selected manifest entry.

6.14.3.190 DeviceManufacturerInfo

```
GenicamFeature<std::string>* FliSfncCamera::DeviceManufacturerInfo
```

Manufacturer information about the device.

6.14.3.191 DeviceMaxThroughput

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceMaxThroughput
```

Maximum bandwidth of the data that can be streamed out of the device. This can be used to estimate if the physical connection(s) can sustain transfer of free-running images from the camera at its maximum speed.

6.14.3.192 DeviceModelName

```
GenicamFeature<std::string>* FliSfncCamera::DeviceModelName
```

Model of the device.

6.14.3.193 DeviceRegistersCheck

```
GenicamFeature* FliSfncCamera::DeviceRegistersCheck
```

Perform the validation of the current register set for consistency. This will update the DeviceRegistersValid flag.

6.14.3.194 DeviceRegistersEndianness

```
GenicamFeature<FliSfncCameraEnum::DeviceRegistersEndiannessEnum>* FliSfncCamera::DeviceRegistersEndianness
```

Endianness of the registers of the device.

6.14.3.195 DeviceRegistersStreamingEnd

```
GenicamFeature* FliSfncCamera::DeviceRegistersStreamingEnd
```

Announce the end of registers streaming. This will do a register set validation for consistency and activate it. This will also update the DeviceRegistersValid flag.

6.14.3.196 DeviceRegistersStreamingStart

```
GenicamFeature* FliSfncCamera::DeviceRegistersStreamingStart
```

Prepare the device for registers streaming without checking for consistency.

6.14.3.197 DeviceRegistersValid

```
GenicamFeature<bool>* FliSfncCamera::DeviceRegistersValid
```

Returns if the current register set is valid and consistent.

6.14.3.198 DeviceReset

```
GenicamFeature* FliSfncCamera::DeviceReset
```

Resets the device to its power up state. After reset, the device must be rediscovered.

6.14.3.199 DeviceScanType

```
GenicamFeature<FliSfncCameraEnum::DeviceScanTypeEnum>* FliSfncCamera::DeviceScanType
```

Scan type of the sensor of the device.

6.14.3.200 DeviceSerialNumber

```
GenicamFeature<std::string>* FliSfncCamera::DeviceSerialNumber
```

Device's serial number. This string is a unique identifier of the device.

6.14.3.201 DeviceSerialPortBaudRate

```
GenicamFeature<FliSfncCameraEnum::DeviceSerialPortBaudRateEnum>* FliSfncCamera::DeviceSerial↔  
PortBaudRate
```

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

6.14.3.202 DeviceSerialPortSelector

```
GenicamFeature<FliSfncCameraEnum::DeviceSerialPortSelectorEnum>* FliSfncCamera::DeviceSerial↔  
PortSelector
```

Selects which serial port of the device to control.

6.14.3.203 DeviceSFNCVersionMajor

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceSFNCVersionMajor
```

Major version of the Standard Features Naming Convention that was used to create the device's GenICam XML.

6.14.3.204 DeviceSFNCVersionMinor

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceSFNCVersionMinor
```

Minor version of the Standard Features Naming Convention that was used to create the device's GenICam XML.

6.14.3.205 DeviceSFNCVersionSubMinor

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceSFNCVersionSubMinor
```

Sub minor version of Standard Features Naming Convention that was used to create the device's GenICam XML.

6.14.3.206 DeviceStreamChannelCount

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceStreamChannelCount
```

Indicates the number of streaming channels supported by the device.

6.14.3.207 DeviceStreamChannelEndianness

```
GenicamFeature<FliSfncCameraEnum::DeviceStreamChannelEndiannessEnum>* FliSfncCamera::DeviceStreamChannelEndianness
```

Endianness of multi-byte pixel data for this stream.

6.14.3.208 DeviceStreamChannelLink

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceStreamChannelLink
```

Index of device's Link to use for streaming the specified stream channel.

6.14.3.209 DeviceStreamChannelPacketSize

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceStreamChannelPacketSize
```

Specifies the stream packet size, in bytes, to send on the selected channel for a Transmitter or specifies the maximum packet size supported by a receiver.

6.14.3.210 DeviceStreamChannelSelector

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceStreamChannelSelector
```

Selects the stream channel to control.

6.14.3.211 DeviceStreamChannelType

```
GenicamFeature<FliSfncCameraEnum::DeviceStreamChannelTypeEnum>* FliSfncCamera::DeviceStreamChannelType
```

Reports the type of the stream channel.

6.14.3.212 DeviceTapGeometry

```
GenicamFeature<FliSfncCameraEnum::DeviceTapGeometryEnum>* FliSfncCamera::DeviceTapGeometry
```

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

6.14.3.213 DeviceTemperature

```
GenicamFeature<double>* FliSfncCamera::DeviceTemperature
```

Device temperature in degrees Celsius (C). It is measured at the location selected by DeviceTemperatureSelector.

6.14.3.214 DeviceTemperatureSelector

```
GenicamFeature<FliSfncCameraEnum::DeviceTemperatureSelectorEnum>* FliSfncCamera::DeviceTemperatureSelector
```

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

6.14.3.215 DeviceTLType

```
GenicamFeature<FliSfncCameraEnum::DeviceTLTypeEnum>* FliSfncCamera::DeviceTLType
```

Transport Layer type of the device.

6.14.3.216 DeviceTLVersionMajor

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceTLVersionMajor
```

Major version of the Transport Layer of the device.

6.14.3.217 DeviceTLVersionMinor

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceTLVersionMinor
```

Minor version of the Transport Layer of the device.

6.14.3.218 DeviceTLVersionSubMinor

```
GenicamFeature<int64_t>* FliSfncCamera::DeviceTLVersionSubMinor
```

Sub minor version of the Transport Layer of the device.

6.14.3.219 DeviceType

```
GenicamFeature<FliSfncCameraEnum::DeviceTypeEnum>* FliSfncCamera::DeviceType
```

Returns the device type.

6.14.3.220 DeviceUserID

```
GenicamFeature<std::string>* FliSfncCamera::DeviceUserID
```

User-programmable device identifier.

6.14.3.221 DeviceVendorName

```
GenicamFeature<std::string>* FliSfncCamera::DeviceVendorName
```

Name of the manufacturer of the device.

6.14.3.222 DeviceVersion

```
GenicamFeature<std::string>* FliSfncCamera::DeviceVersion
```

Version of the device.

6.14.3.223 EncoderDivider

```
GenicamFeature<int64_t>* FliSfncCamera::EncoderDivider
```

Sets how many Encoder increments/decrements are needed to generate an Encoder output pulse signal.

6.14.3.224 EncoderMode

```
GenicamFeature<FliSfncCameraEnum::EncoderModeEnum>* FliSfncCamera::EncoderMode
```

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

6.14.3.225 EncoderOutputMode

```
GenicamFeature<FliSfncCameraEnum::EncoderOutputModeEnum>* FliSfncCamera::EncoderOutputMode
```

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

6.14.3.226 EncoderReset

```
GenicamFeature* FliSfncCamera::EncoderReset
```

Does a software reset of the selected Encoder and starts it. The Encoder starts counting events immediately after the reset. EncoderReset can be used to reset the Encoder independently from the EncoderResetSource.

6.14.3.227 EncoderResetActivation

```
GenicamFeature<FliSfncCameraEnum::EncoderResetActivationEnum>* FliSfncCamera::EncoderReset↔  
Activation
```

Selects the Activation mode of the Encoder Reset Source signal.

6.14.3.228 EncoderResetSource

```
GenicamFeature<FliSfncCameraEnum::EncoderResetSourceEnum>* FliSfncCamera::EncoderResetSource
```

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

6.14.3.229 EncoderResolution

```
GenicamFeature<double>* FliSfncCamera::EncoderResolution
```

Defines the resolution of one encoder step.

6.14.3.230 EncoderSelector

GenicamFeature<FliSfncCameraEnum::EncoderSelectorEnum>* FliSfncCamera::EncoderSelector

Selects which Encoder to configure.

6.14.3.231 EncoderSourceA

GenicamFeature<FliSfncCameraEnum::EncoderSourceAEnum>* FliSfncCamera::EncoderSourceA

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

6.14.3.232 EncoderSourceB

GenicamFeature<FliSfncCameraEnum::EncoderSourceBEnum>* FliSfncCamera::EncoderSourceB

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

6.14.3.233 EncoderStatus

GenicamFeature<FliSfncCameraEnum::EncoderStatusEnum>* FliSfncCamera::EncoderStatus

Returns the motion status of the encoder.

6.14.3.234 EncoderTimeout

GenicamFeature<double>* FliSfncCamera::EncoderTimeout

Sets the maximum time interval between encoder counter increments before the status turns to static.

6.14.3.235 EncoderValue

GenicamFeature<int64_t>* FliSfncCamera::EncoderValue

Reads or writes the current value of the position counter of the selected Encoder.

6.14.3.236 EncoderValueAtReset

```
GenicamFeature<int64_t>* FliSfncCamera::EncoderValueAtReset
```

Reads the value of the of the position counter of the selected Encoder when it was reset by a signal or by an explicit EncoderReset command.

6.14.3.237 EventAcquisitionEnd

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionEnd
```

Returns the unique Identifier of the Acquisition End type of Event.

6.14.3.238 EventAcquisitionEndFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionEndFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Acquisition End Event.

6.14.3.239 EventAcquisitionEndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionEndTimestamp
```

Returns the Timestamp of the Acquisition End Event.

6.14.3.240 EventAcquisitionError

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionError
```

Returns the unique Identifier of the Acquisition Error type of Event.

6.14.3.241 EventAcquisitionErrorFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionErrorFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Acquisition Error Event.

6.14.3.242 EventAcquisitionErrorTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionErrorTimestamp
```

Returns the Timestamp of the Acquisition Error Event.

6.14.3.243 EventAcquisitionStart

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionStart
```

Returns the unique Identifier of the Acquisition Start type of Event.

6.14.3.244 EventAcquisitionStartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionStartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Acquisition Start Event.

6.14.3.245 EventAcquisitionStartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionStartTimestamp
```

Returns the Timestamp of the Acquisition Start Event.

6.14.3.246 EventAcquisitionTransferEnd

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionTransferEnd
```

Returns the unique Identifier of the Acquisition Transfer End type of Event.

6.14.3.247 EventAcquisitionTransferEndFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionTransferEndFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Acquisition Transfer End Event.

6.14.3.248 EventAcquisitionTransferEndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionTransferEndTimestamp
```

Returns the Timestamp of the Acquisition Transfer End Event.

6.14.3.249 EventAcquisitionTransferStart

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionTransferStart
```

Returns the unique Identifier of the Acquisition Transfer Start type of Event.

6.14.3.250 EventAcquisitionTransferStartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionTransferStartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Acquisition Transfer Start Event.

6.14.3.251 EventAcquisitionTransferStartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionTransferStartTimestamp
```

Returns the Timestamp of the Acquisition Transfer Start Event.

6.14.3.252 EventAcquisitionTrigger

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionTrigger
```

Returns the unique Identifier of the Acquisition Trigger type of Event.

6.14.3.253 EventAcquisitionTriggerFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionTriggerFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Acquisition Trigger Event.

6.14.3.254 EventAcquisitionTriggerMissed

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionTriggerMissed
```

Returns the unique Identifier of the Acquisition Trigger Missed type of Event.

6.14.3.255 EventAcquisitionTriggerMissedFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionTriggerMissedFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Acquisition Trigger Missed Event.

6.14.3.256 EventAcquisitionTriggerMissedTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionTriggerMissedTimestamp
```

Returns the Timestamp of the Acquisition Trigger Missed Event.

6.14.3.257 EventAcquisitionTriggerTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventAcquisitionTriggerTimestamp
```

Returns the Timestamp of the Acquisition Trigger Event.

6.14.3.258 EventActionLate

```
GenicamFeature<int64_t>* FliSfncCamera::EventActionLate
```

Returns the unique Identifier of the Action Late type of Event.

6.14.3.259 EventActionLateFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventActionLateFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Action Late Event.

6.14.3.260 EventActionLateTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventActionLateTimestamp
```

Returns the Timestamp of the Action Late Event.

6.14.3.261 EventCounter0End

```
GenicamFeature<int64_t>* FliSfncCamera::EventCounter0End
```

Returns the unique Identifier of the Counter 0 End type of Event.

6.14.3.262 EventCounter0EndFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventCounter0EndFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Counter 0 End Event.

6.14.3.263 EventCounter0EndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventCounter0EndTimestamp
```

Returns the Timestamp of the Counter 0 End Event.

6.14.3.264 EventCounter0Start

```
GenicamFeature<int64_t>* FliSfncCamera::EventCounter0Start
```

Returns the unique Identifier of the Counter 0 Start type of Event.

6.14.3.265 EventCounter0StartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventCounter0StartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Counter 0 Start Event.

6.14.3.266 EventCounter0StartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventCounter0StartTimestamp
```

Returns the Timestamp of the Counter 0 Start Event.

6.14.3.267 EventCounter1End

```
GenicamFeature<int64_t>* FliSfncCamera::EventCounter1End
```

Returns the unique Identifier of the Counter 1 End type of Event.

6.14.3.268 EventCounter1EndFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventCounter1EndFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Counter 1 End Event.

6.14.3.269 EventCounter1EndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventCounter1EndTimestamp
```

Returns the Timestamp of the Counter 1 End Event.

6.14.3.270 EventCounter1Start

```
GenicamFeature<int64_t>* FliSfncCamera::EventCounter1Start
```

Returns the unique Identifier of the Counter 1 Start type of Event.

6.14.3.271 EventCounter1StartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventCounter1StartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Counter 1 Start Event.

6.14.3.272 EventCounter1StartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventCounter1StartTimestamp
```

Returns the Timestamp of the Counter 1 Start Event.

6.14.3.273 EventEncoder0Restarted

```
GenicamFeature<int64_t>* FliSfncCamera::EventEncoder0Restarted
```

Returns the unique Identifier of the Encoder 0 Restarted type of Event.

6.14.3.274 EventEncoder0RestartedFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventEncoder0RestartedFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Encoder 0 Restarted Event.

6.14.3.275 EventEncoder0RestartedTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventEncoder0RestartedTimestamp
```

Returns the Timestamp of the Encoder 0 Restarted Event.

6.14.3.276 EventEncoder0Stopped

```
GenicamFeature<int64_t>* FliSfncCamera::EventEncoder0Stopped
```

Returns the unique Identifier of the Encoder 0 Stopped type of Event.

6.14.3.277 EventEncoder0StoppedFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventEncoder0StoppedFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Encoder 0 Stopped Event.

6.14.3.278 EventEncoder0StoppedTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventEncoder0StoppedTimestamp
```

Returns the Timestamp of the Encoder 0 Stopped Event.

6.14.3.279 EventEncoder1Restarted

```
GenicamFeature<int64_t>* FliSfncCamera::EventEncoder1Restarted
```

Returns the unique Identifier of the Encoder 1 Restarted type of Event.

6.14.3.280 EventEncoder1RestartedFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventEncoder1RestartedFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Encoder 1 Restarted Event.

6.14.3.281 EventEncoder1RestartedTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventEncoder1RestartedTimestamp
```

Returns the Timestamp of the Encoder 1 Restarted Event.

6.14.3.282 EventEncoder1Stopped

```
GenicamFeature<int64_t>* FliSfncCamera::EventEncoder1Stopped
```

Returns the unique Identifier of the Encoder 1 Stopped type of Event.

6.14.3.283 EventEncoder1StoppedFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventEncoder1StoppedFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Encoder 1 Stopped Event.

6.14.3.284 EventEncoder1StoppedTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventEncoder1StoppedTimestamp
```

Returns the Timestamp of the Encoder 1 Stopped Event.

6.14.3.285 EventError

```
GenicamFeature<int64_t>* FliSfncCamera::EventError
```

Returns the unique identifier of the Error type of Event. It can be used to register a callback function to be notified of the Error event occurrence. Its value uniquely identifies that the event received was an Error.

6.14.3.286 EventErrorCode

```
GenicamFeature<int64_t>* FliSfncCamera::EventErrorCode
```

Returns an error code for the error(s) that happened.

6.14.3.287 EventErrorFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventErrorFrameID
```

If applicable, returns the unique Identifier of the Frame (or image) that generated the Error Event.

6.14.3.288 EventErrorTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventErrorTimestamp
```

Returns the Timestamp of the Error Event. It can be used to determine when the event occurred.

6.14.3.289 EventExposureEnd

```
GenicamFeature<int64_t>* FliSfncCamera::EventExposureEnd
```

Returns the unique Identifier of the Exposure End type of Event.

6.14.3.290 EventExposureEndFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventExposureEndFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Exposure End Event.

6.14.3.291 EventExposureEndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventExposureEndTimestamp
```

Returns the Timestamp of the Exposure End Event.

6.14.3.292 EventExposureStart

```
GenicamFeature<int64_t>* FliSfncCamera::EventExposureStart
```

Returns the unique Identifier of the Exposure Start type of Event.

6.14.3.293 EventExposureStartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventExposureStartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Exposure Start Event.

6.14.3.294 EventExposureStartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventExposureStartTimestamp
```

Returns the Timestamp of the Exposure Start Event.

6.14.3.295 EventFrameBurstEnd

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameBurstEnd
```

Returns the unique Identifier of the Frame Burst End type of Event.

6.14.3.296 EventFrameBurstEndFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameBurstEndFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Frame Burst End Event.

6.14.3.297 EventFrameBurstEndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameBurstEndTimestamp
```

Returns the Timestamp of the Frame Burst End Event.

6.14.3.298 EventFrameBurstStart

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameBurstStart
```

Returns the unique Identifier of the Frame Burst Start type of Event.

6.14.3.299 EventFrameBurstStartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameBurstStartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Frame Burst Start Event.

6.14.3.300 EventFrameBurstStartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameBurstStartTimestamp
```

Returns the Timestamp of the Frame Burst Start Event.

6.14.3.301 EventFrameEnd

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameEnd
```

Returns the unique Identifier of the Frame End type of Event.

6.14.3.302 EventFrameEndFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameEndFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Frame End Event.

6.14.3.303 EventFrameEndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameEndTimestamp
```

Returns the Timestamp of the Frame End Event.

6.14.3.304 EventFrameStart

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameStart
```

Returns the unique Identifier of the Frame Start type of Event.

6.14.3.305 EventFrameStartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameStartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Frame Start Event.

6.14.3.306 EventFrameStartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameStartTimestamp
```

Returns the Timestamp of the Frame Start Event.

6.14.3.307 EventFrameTransferEnd

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameTransferEnd
```

Returns the unique Identifier of the Frame Transfer End type of Event.

6.14.3.308 EventFrameTransferEndFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameTransferEndFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Frame Transfer End Event.

6.14.3.309 EventFrameTransferEndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameTransferEndTimestamp
```

Returns the Timestamp of the Frame Transfer End Event.

6.14.3.310 EventFrameTransferStart

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameTransferStart
```

Returns the unique Identifier of the Frame Transfer Start type of Event.

6.14.3.311 EventFrameTransferStartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameTransferStartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Frame Transfer Start Event.

6.14.3.312 EventFrameTransferStartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameTransferStartTimestamp
```

Returns the Timestamp of the Frame Transfer Start Event.

6.14.3.313 EventFrameTrigger

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameTrigger
```

Returns the unique Identifier of the Frame Trigger type of Event.

6.14.3.314 EventFrameTriggerFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameTriggerFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Frame Trigger Event.

6.14.3.315 EventFrameTriggerMissed

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameTriggerMissed
```

Returns the unique Identifier of the Frame Trigger Missed type of Event.

6.14.3.316 EventFrameTriggerMissedFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameTriggerMissedFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Frame Trigger Missed Event.

6.14.3.317 EventFrameTriggerMissedTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameTriggerMissedTimestamp
```

Returns the Timestamp of the Frame Trigger Missed Event.

6.14.3.318 EventFrameTriggerTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventFrameTriggerTimestamp
```

Returns the Timestamp of the Frame Trigger Event.

6.14.3.319 EventLine0AnyEdge

```
GenicamFeature<int64_t>* FliSfncCamera::EventLine0AnyEdge
```

Returns the unique Identifier of the Line 0 Any Edge type of Event.

6.14.3.320 EventLine0AnyEdgeFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventLine0AnyEdgeFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Line 0 Any Edge Event.

6.14.3.321 EventLine0AnyEdgeTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventLine0AnyEdgeTimestamp
```

Returns the Timestamp of the Line 0 Any Edge Event.

6.14.3.322 EventLine0FallingEdge

```
GenicamFeature<int64_t>* FliSfncCamera::EventLine0FallingEdge
```

Returns the unique Identifier of the Line 0 Falling Edge type of Event.

6.14.3.323 EventLine0FallingEdgeFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventLine0FallingEdgeFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Line 0 Falling Edge Event.

6.14.3.324 EventLine0FallingEdgeTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventLine0FallingEdgeTimestamp
```

Returns the Timestamp of the Line 0 Falling Edge Event.

6.14.3.325 EventLine0RisingEdge

```
GenicamFeature<int64_t>* FliSfncCamera::EventLine0RisingEdge
```

Returns the unique Identifier of the Line 0 Rising Edge type of Event.

6.14.3.326 EventLine0RisingEdgeFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventLine0RisingEdgeFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Line 0 Rising Edge Event.

6.14.3.327 EventLine0RisingEdgeTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventLine0RisingEdgeTimestamp
```

Returns the Timestamp of the Line 0 Rising Edge Event.

6.14.3.328 EventLine1AnyEdge

```
GenicamFeature<int64_t>* FliSfncCamera::EventLine1AnyEdge
```

Returns the unique Identifier of the Line 1 Any Edge type of Event.

6.14.3.329 EventLine1AnyEdgeFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventLine1AnyEdgeFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Line 1 Any Edge Event.

6.14.3.330 EventLine1AnyEdgeTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventLine1AnyEdgeTimestamp
```

Returns the Timestamp of the Line 1 Any Edge Event.

6.14.3.331 EventLine1FallingEdge

```
GenicamFeature<int64_t>* FliSfncCamera::EventLine1FallingEdge
```

Returns the unique Identifier of the Line 1 Falling Edge type of Event.

6.14.3.332 EventLine1FallingEdgeFrameID

GenicamFeature<int64_t>* FliSfncCamera::EventLine1FallingEdgeFrameID

Returns the unique Identifier of the Frame (or image) that generated the Line 1 Falling Edge Event.

6.14.3.333 EventLine1FallingEdgeTimestamp

GenicamFeature<int64_t>* FliSfncCamera::EventLine1FallingEdgeTimestamp

Returns the Timestamp of the Line 1 Falling Edge Event.

6.14.3.334 EventLine1RisingEdge

GenicamFeature<int64_t>* FliSfncCamera::EventLine1RisingEdge

Returns the unique Identifier of the Line 1 Rising Edge type of Event.

6.14.3.335 EventLine1RisingEdgeFrameID

GenicamFeature<int64_t>* FliSfncCamera::EventLine1RisingEdgeFrameID

Returns the unique Identifier of the Frame (or image) that generated the Line 1 Rising Edge Event.

6.14.3.336 EventLine1RisingEdgeTimestamp

GenicamFeature<int64_t>* FliSfncCamera::EventLine1RisingEdgeTimestamp

Returns the Timestamp of the Line 1 Rising Edge Event.

6.14.3.337 EventLineEnd

GenicamFeature<int64_t>* FliSfncCamera::EventLineEnd

Returns the unique Identifier of the Line End type of Event.

6.14.3.338 EventLineEndFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventLineEndFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Line End Event.

6.14.3.339 EventLineEndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventLineEndTimestamp
```

Returns the Timestamp of the Line End Event.

6.14.3.340 EventLineStart

```
GenicamFeature<int64_t>* FliSfncCamera::EventLineStart
```

Returns the unique Identifier of the Line Start type of Event.

6.14.3.341 EventLineStartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventLineStartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Line Start Event.

6.14.3.342 EventLineStartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventLineStartTimestamp
```

Returns the Timestamp of the Line Start Event.

6.14.3.343 EventLineTrigger

```
GenicamFeature<int64_t>* FliSfncCamera::EventLineTrigger
```

Returns the unique Identifier of the Line Trigger type of Event.

6.14.3.344 EventLineTriggerFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventLineTriggerFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Line Trigger Event.

6.14.3.345 EventLineTriggerMissed

```
GenicamFeature<int64_t>* FliSfncCamera::EventLineTriggerMissed
```

Returns the unique Identifier of the Line Trigger Missed type of Event.

6.14.3.346 EventLineTriggerMissedFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventLineTriggerMissedFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Line Trigger Missed Event.

6.14.3.347 EventLineTriggerMissedTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventLineTriggerMissedTimestamp
```

Returns the Timestamp of the Line Trigger Missed Event.

6.14.3.348 EventLineTriggerTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventLineTriggerTimestamp
```

Returns the Timestamp of the Line Trigger Event.

6.14.3.349 EventLinkSpeedChange

```
GenicamFeature<int64_t>* FliSfncCamera::EventLinkSpeedChange
```

Returns the unique Identifier of the Link Speed Change type of Event.

6.14.3.350 EventLinkSpeedChangeFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventLinkSpeedChangeFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Link Speed Change Event.

6.14.3.351 EventLinkSpeedChangeTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventLinkSpeedChangeTimestamp
```

Returns the Timestamp of the Link Speed Change Event.

6.14.3.352 EventLinkTrigger0

```
GenicamFeature<int64_t>* FliSfncCamera::EventLinkTrigger0
```

Returns the unique Identifier of the Link Trigger 0 type of Event.

6.14.3.353 EventLinkTrigger0FrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventLinkTrigger0FrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Link Trigger 0 Event.

6.14.3.354 EventLinkTrigger0Timestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventLinkTrigger0Timestamp
```

Returns the Timestamp of the Link Trigger 0 Event.

6.14.3.355 EventLinkTrigger1

```
GenicamFeature<int64_t>* FliSfncCamera::EventLinkTrigger1
```

Returns the unique Identifier of the Link Trigger 1 type of Event.

6.14.3.356 EventLinkTrigger1FrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventLinkTrigger1FrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Link Trigger 1 Event.

6.14.3.357 EventLinkTrigger1Timestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventLinkTrigger1Timestamp
```

Returns the Timestamp of the Link Trigger 1 Event.

6.14.3.358 EventNotification

```
GenicamFeature<FliSfncCameraEnum::EventNotificationEnum>* FliSfncCamera::EventNotification
```

Activate or deactivate the notification to the host application of the occurrence of the selected Event.

6.14.3.359 EventPrimaryApplicationSwitch

```
GenicamFeature<int64_t>* FliSfncCamera::EventPrimaryApplicationSwitch
```

Returns the unique Identifier of the Primary Application Switch type of Event.

6.14.3.360 EventPrimaryApplicationSwitchFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventPrimaryApplicationSwitchFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Primary Application Switch Event.

6.14.3.361 EventPrimaryApplicationSwitchTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventPrimaryApplicationSwitchTimestamp
```

Returns the Timestamp of the Primary Application Switch Event.

6.14.3.362 EventSelector

GenicamFeature<FliSfncCameraEnum::EventSelectorEnum>* FliSfncCamera::EventSelector

Selects which Event to signal to the host application.

6.14.3.363 EventSequencerSetChange

GenicamFeature<int64_t>* FliSfncCamera::EventSequencerSetChange

Returns the unique Identifier of the Sequencer Set Change type of Event.

6.14.3.364 EventSequencerSetChangeFrameID

GenicamFeature<int64_t>* FliSfncCamera::EventSequencerSetChangeFrameID

Returns the unique Identifier of the Frame (or image) that generated the Sequencer Set Change Event.

6.14.3.365 EventSequencerSetChangeTimestamp

GenicamFeature<int64_t>* FliSfncCamera::EventSequencerSetChangeTimestamp

Returns the Timestamp of the Sequencer Set Change Event.

6.14.3.366 EventStream0TransferBlockEnd

GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBlockEnd

Returns the unique Identifier of the Stream 0 Transfer Block End type of Event.

6.14.3.367 EventStream0TransferBlockEndFrameID

GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBlockEndFrameID

Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Block End Event.

6.14.3.368 EventStream0TransferBlockEndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBlockEndTimestamp
```

Returns the Timestamp of the Stream 0 Transfer Block End Event.

6.14.3.369 EventStream0TransferBlockStart

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBlockStart
```

Returns the unique Identifier of the Stream 0 Transfer Block Start type of Event.

6.14.3.370 EventStream0TransferBlockStartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBlockStartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Block Start Event.

6.14.3.371 EventStream0TransferBlockStartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBlockStartTimestamp
```

Returns the Timestamp of the Stream 0 Transfer Block Start Event.

6.14.3.372 EventStream0TransferBlockTrigger

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBlockTrigger
```

Returns the unique Identifier of the Stream 0 Transfer Block Trigger type of Event.

6.14.3.373 EventStream0TransferBlockTriggerFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBlockTriggerFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Block Trigger Event.

6.14.3.374 EventStream0TransferBlockTriggerTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBlockTriggerTimestamp
```

Returns the Timestamp of the Stream 0 Transfer Block Trigger Event.

6.14.3.375 EventStream0TransferBurstEnd

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBurstEnd
```

Returns the unique Identifier of the Stream 0 Transfer Burst End type of Event.

6.14.3.376 EventStream0TransferBurstEndFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBurstEndFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Burst End Event.

6.14.3.377 EventStream0TransferBurstEndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBurstEndTimestamp
```

Returns the Timestamp of the Stream 0 Transfer Burst End Event.

6.14.3.378 EventStream0TransferBurstStart

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBurstStart
```

Returns the unique Identifier of the Stream 0 Transfer Burst Start type of Event.

6.14.3.379 EventStream0TransferBurstStartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBurstStartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Burst Start Event.

6.14.3.380 EventStream0TransferBurstStartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferBurstStartTimestamp
```

Returns the Timestamp of the Stream 0 Transfer Burst Start Event.

6.14.3.381 EventStream0TransferEnd

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferEnd
```

Returns the unique Identifier of the Stream 0 Transfer End type of Event.

6.14.3.382 EventStream0TransferEndFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferEndFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer End Event.

6.14.3.383 EventStream0TransferEndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferEndTimestamp
```

Returns the Timestamp of the Stream 0 Transfer End Event.

6.14.3.384 EventStream0TransferOverflow

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferOverflow
```

Returns the unique Identifier of the Stream 0 Transfer Overflow type of Event.

6.14.3.385 EventStream0TransferOverflowFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferOverflowFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Overflow Event.

6.14.3.386 EventStream0TransferOverflowTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferOverflowTimestamp
```

Returns the Timestamp of the Stream 0 Transfer Overflow Event.

6.14.3.387 EventStream0TransferPause

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferPause
```

Returns the unique Identifier of the Stream 0 Transfer Pause type of Event.

6.14.3.388 EventStream0TransferPauseFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferPauseFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Pause Event.

6.14.3.389 EventStream0TransferPauseTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferPauseTimestamp
```

Returns the Timestamp of the Stream 0 Transfer Pause Event.

6.14.3.390 EventStream0TransferResume

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferResume
```

Returns the unique Identifier of the Stream 0 Transfer Resume type of Event.

6.14.3.391 EventStream0TransferResumeFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferResumeFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Resume Event.

6.14.3.392 EventStream0TransferResumeTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferResumeTimestamp
```

Returns the Timestamp of the Stream 0 Transfer Resume Event.

6.14.3.393 EventStream0TransferStart

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferStart
```

Returns the unique Identifier of the Stream 0 Transfer Start type of Event.

6.14.3.394 EventStream0TransferStartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferStartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Stream 0 Transfer Start Event.

6.14.3.395 EventStream0TransferStartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventStream0TransferStartTimestamp
```

Returns the Timestamp of the Stream 0 Transfer Start Event.

6.14.3.396 EventTest

```
GenicamFeature<int64_t>* FliSfncCamera::EventTest
```

Returns the unique identifier of the Event Test type of event generated using the TestEventGenerate command. It can be used to register a callback function to be notified of the EventTest event occurrence. Its value uniquely identifies that the event received was an Event Test.

6.14.3.397 EventTestTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventTestTimestamp
```

Returns the Timestamp of the Event Test event. It can be used to determine when the event occurred.

6.14.3.398 EventTimer0End

```
GenicamFeature<int64_t>* FliSfncCamera::EventTimer0End
```

Returns the unique Identifier of the Timer 0 End type of Event.

6.14.3.399 EventTimer0EndFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventTimer0EndFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Timer 0 End Event.

6.14.3.400 EventTimer0EndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventTimer0EndTimestamp
```

Returns the Timestamp of the Timer 0 End Event.

6.14.3.401 EventTimer0Start

```
GenicamFeature<int64_t>* FliSfncCamera::EventTimer0Start
```

Returns the unique Identifier of the Timer 0 Start type of Event.

6.14.3.402 EventTimer0StartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventTimer0StartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Timer 0 Start Event.

6.14.3.403 EventTimer0StartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventTimer0StartTimestamp
```

Returns the Timestamp of the Timer 0 Start Event.

6.14.3.404 EventTimer1End

```
GenicamFeature<int64_t>* FliSfncCamera::EventTimer1End
```

Returns the unique Identifier of the Timer 1 End type of Event.

6.14.3.405 EventTimer1EndFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventTimer1EndFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Timer 1 End Event.

6.14.3.406 EventTimer1EndTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventTimer1EndTimestamp
```

Returns the Timestamp of the Timer 1 End Event.

6.14.3.407 EventTimer1Start

```
GenicamFeature<int64_t>* FliSfncCamera::EventTimer1Start
```

Returns the unique Identifier of the Timer 1 Start type of Event.

6.14.3.408 EventTimer1StartFrameID

```
GenicamFeature<int64_t>* FliSfncCamera::EventTimer1StartFrameID
```

Returns the unique Identifier of the Frame (or image) that generated the Timer 1 Start Event.

6.14.3.409 EventTimer1StartTimestamp

```
GenicamFeature<int64_t>* FliSfncCamera::EventTimer1StartTimestamp
```

Returns the Timestamp of the Timer 1 Start Event.

6.14.3.410 ExposureAuto

```
GenicamFeature<FliSfncCameraEnum::ExposureAutoEnum>* FliSfncCamera::ExposureAuto
```

Sets the automatic exposure mode when ExposureMode is Timed. The exact algorithm used to implement this control is device-specific.

6.14.3.411 ExposureMode

```
GenicamFeature<FliSfncCameraEnum::ExposureModeEnum>* FliSfncCamera::ExposureMode
```

Sets the operation mode of the Exposure.

6.14.3.412 ExposureTime

```
GenicamFeature<double>* FliSfncCamera::ExposureTime
```

Sets the Exposure time when ExposureMode is Timed and ExposureAuto is Off. This controls the duration where the photosensitive cells are exposed to light.

6.14.3.413 ExposureTimeMode

```
GenicamFeature<FliSfncCameraEnum::ExposureTimeModeEnum>* FliSfncCamera::ExposureTimeMode
```

Sets the configuration mode of the ExposureTime feature.

6.14.3.414 ExposureTimeSelector

```
GenicamFeature<FliSfncCameraEnum::ExposureTimeSelectorEnum>* FliSfncCamera::ExposureTime↔  
Selector
```

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

6.14.3.415 FileAccessLength

```
GenicamFeature<int64_t>* FliSfncCamera::FileAccessLength
```

Controls the Length of the mapping between the device file storage and the FileAccessBuffer.

6.14.3.416 FileAccessOffset

```
GenicamFeature<int64_t>* FliSfncCamera::FileAccessOffset
```

Controls the Offset of the mapping between the device file storage and the FileAccessBuffer.

6.14.3.417 FileOpenMode

```
GenicamFeature<FliSfncCameraEnum::FileOpenModeEnum>* FliSfncCamera::FileOpenMode
```

Selects the access mode in which a file is opened in the device.

6.14.3.418 FileOperationExecute

```
GenicamFeature* FliSfncCamera::FileOperationExecute
```

Executes the operation selected by FileOperationSelector on the selected file.

6.14.3.419 FileOperationResult

```
GenicamFeature<int64_t>* FliSfncCamera::FileOperationResult
```

Represents the file operation result. For Read or Write operations, the number of successfully read/written bytes is returned.

6.14.3.420 FileOperationSelector

```
GenicamFeature<FliSfncCameraEnum::FileOperationSelectorEnum>* FliSfncCamera::FileOperation↵  
Selector
```

Selects the target operation for the selected file in the device. This Operation is executed when the FileOperation↵
Execute feature is called.

6.14.3.421 FileOperationStatus

```
GenicamFeature<FliSfncCameraEnum::FileOperationStatusEnum>* FliSfncCamera::FileOperationStatus
```

Represents the file operation execution status.

6.14.3.422 FileSelector

```
GenicamFeature<FliSfncCameraEnum::FileSelectorEnum>* FliSfncCamera::FileSelector
```

Selects the target file in the device.

6.14.3.423 FileSize

```
GenicamFeature<int64_t>* FliSfncCamera::FileSize
```

Represents the size of the selected file in bytes.

6.14.3.424 Gain

```
GenicamFeature<double>* FliSfncCamera::Gain
```

Controls the selected gain as an absolute physical value. This is an amplification factor applied to the video signal.

6.14.3.425 GainAuto

```
GenicamFeature<FliSfncCameraEnum::GainAutoEnum>* FliSfncCamera::GainAuto
```

Sets the automatic gain control (AGC) mode. The exact algorithm used to implement AGC is device-specific.

6.14.3.426 GainAutoBalance

```
GenicamFeature<FliSfncCameraEnum::GainAutoBalanceEnum>* FliSfncCamera::GainAutoBalance
```

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.

6.14.3.427 GainSelector

```
GenicamFeature<FliSfncCameraEnum::GainSelectorEnum>* FliSfncCamera::GainSelector
```

Selects which Gain is controlled by the various Gain features.

6.14.3.428 Gamma

```
GenicamFeature<double>* FliSfncCamera::Gamma
```

Controls the gamma correction of pixel intensity. This is typically used to compensate for non-linearity of the display system (such as CRT).

6.14.3.429 GenDCStreamingMode

```
GenicamFeature<FliSfncCameraEnum::GenDCStreamingModeEnum>* FliSfncCamera::GenDCStreamingMode
```

Controls the device's streaming format.

6.14.3.430 GenDCStreamingStatus

```
GenicamFeature<FliSfncCameraEnum::GenDCStreamingStatusEnum>* FliSfncCamera::GenDCStreaming↔  
Status
```

Returns whether the current device data streaming format is GenDC. This value is conditioned by the GenDC↔
StreamingMode.

6.14.3.431 GevActiveLinkCount

```
GenicamFeature<int64_t>* FliSfncCamera::GevActiveLinkCount
```

Indicates the current number of active logical links.

6.14.3.432 GevCCP

```
GenicamFeature<FliSfncCameraEnum::GevCCPEnum>* FliSfncCamera::GevCCP
```

Controls the device access privilege of an application.

6.14.3.433 GevCurrentDefaultGateway

```
GenicamFeature<int64_t>* FliSfncCamera::GevCurrentDefaultGateway
```

Reports the default gateway IP address of the given logical link.

6.14.3.434 **GevCurrentIPAddress**

```
GenicamFeature<int64_t>* FliSfncCamera::GevCurrentIPAddress
```

Reports the IP address for the given logical link.

6.14.3.435 **GevCurrentIPConfigurationDHCP**

```
GenicamFeature<bool>* FliSfncCamera::GevCurrentIPConfigurationDHCP
```

Controls whether the DHCP IP configuration scheme is activated on the given logical link.

6.14.3.436 **GevCurrentIPConfigurationLLA**

```
GenicamFeature<bool>* FliSfncCamera::GevCurrentIPConfigurationLLA
```

Controls whether the Link Local Address IP configuration scheme is activated on the given logical link.

6.14.3.437 **GevCurrentIPConfigurationPersistentIP**

```
GenicamFeature<bool>* FliSfncCamera::GevCurrentIPConfigurationPersistentIP
```

Controls whether the PersistentIP configuration scheme is activated on the given logical link.

6.14.3.438 **GevCurrentPhysicalLinkConfiguration**

```
GenicamFeature<FliSfncCameraEnum::GevCurrentPhysicalLinkConfigurationEnum>* FliSfncCamera::↔  
GevCurrentPhysicalLinkConfiguration
```

Indicates the current physical link configuration of the device.

6.14.3.439 **GevCurrentSubnetMask**

```
GenicamFeature<int64_t>* FliSfncCamera::GevCurrentSubnetMask
```

Reports the subnet mask of the given logical link.

6.14.3.440 **GevDiscoveryAckDelay**

```
GenicamFeature<int64_t>* FliSfncCamera::GevDiscoveryAckDelay
```

Indicates the maximum randomized delay the device will wait to acknowledge a discovery command.

6.14.3.441 **GevFirstURL**

```
GenicamFeature<std::string>* FliSfncCamera::GevFirstURL
```

Indicates the first URL to the GenICam XML device description file. The First URL is used as the first choice by the application to retrieve the GenICam XML device description file.

6.14.3.442 **GevGVCPExtendedStatusCodes**

```
GenicamFeature<bool>* FliSfncCamera::GevGVCPExtendedStatusCodes
```

Enables the generation of extended status codes.

6.14.3.443 **GevGVCPExtendedStatusCodesSelector**

```
GenicamFeature<FliSfncCameraEnum::GevGVCPExtendedStatusCodesSelectorEnum>* FliSfncCamera::↔  
GevGVCPExtendedStatusCodesSelector
```

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

6.14.3.444 **GevGVCPPendingAck**

```
GenicamFeature<bool>* FliSfncCamera::GevGVCPPendingAck
```

Enables the generation of PENDING_ACK.

6.14.3.445 **GevGVSPExtendedIDMode**

```
GenicamFeature<FliSfncCameraEnum::GevGVSPExtendedIDModeEnum>* FliSfncCamera::GevGVSPExtended↔  
IDMode
```

Enables the extended IDs mode.

6.14.3.446 **GevInterfaceSelector**

```
GenicamFeature<int64_t>* FliSfncCamera::GevInterfaceSelector
```

Selects which logical link to control.

6.14.3.447 **GevIPConfigurationStatus**

```
GenicamFeature<FliSfncCameraEnum::GevIPConfigurationStatusEnum>* FliSfncCamera::GevIPConfiguration↔  
Status
```

Reports the current IP configuration status.

6.14.3.448 **GevMACAddress**

```
GenicamFeature<int64_t>* FliSfncCamera::GevMACAddress
```

MAC address of the logical link.

6.14.3.449 **GevMCDA**

```
GenicamFeature<int64_t>* FliSfncCamera::GevMCDA
```

Controls the destination IP address for the message channel.

6.14.3.450 **GevMCPHostPort**

```
GenicamFeature<int64_t>* FliSfncCamera::GevMCPHostPort
```

Controls the port to which the device must send messages. Setting this value to 0 closes the message channel.

6.14.3.451 **GevMCRC**

```
GenicamFeature<int64_t>* FliSfncCamera::GevMCRC
```

Controls the number of retransmissions allowed when a message channel message times out.

6.14.3.452 **GevMCSP**

```
GenicamFeature<int64_t>* FliSfncCamera::GevMCSP
```

This feature indicates the source port for the message channel.

6.14.3.453 **GevMCTT**

```
GenicamFeature<int64_t>* FliSfncCamera::GevMCTT
```

Provides the transmission timeout value in milliseconds.

6.14.3.454 **GevPAUSEFrameReception**

```
GenicamFeature<bool>* FliSfncCamera::GevPAUSEFrameReception
```

Controls whether incoming PAUSE Frames are handled on the given logical link.

6.14.3.455 **GevPAUSEFrameTransmission**

```
GenicamFeature<bool>* FliSfncCamera::GevPAUSEFrameTransmission
```

Controls whether PAUSE Frames can be generated on the given logical link.

6.14.3.456 **GevPersistentDefaultGateway**

```
GenicamFeature<int64_t>* FliSfncCamera::GevPersistentDefaultGateway
```

Controls the persistent default gateway for this logical link. It is only used when the device boots with the Persistent IP configuration scheme.

6.14.3.457 **GevPersistentIPAddress**

```
GenicamFeature<int64_t>* FliSfncCamera::GevPersistentIPAddress
```

Controls the Persistent IP address for this logical link. It is only used when the device boots with the Persistent IP configuration scheme.

6.14.3.458 **GevPersistentSubnetMask**

```
GenicamFeature<int64_t>* FliSfncCamera::GevPersistentSubnetMask
```

Controls the Persistent subnet mask associated with the Persistent IP address on this logical link. It is only used when the device boots with the Persistent IP configuration scheme.

6.14.3.459 **GevPhysicalLinkConfiguration**

```
GenicamFeature<FliSfncCameraEnum::GevPhysicalLinkConfigurationEnum>* FliSfncCamera::Gev↔  
PhysicalLinkConfiguration
```

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

6.14.3.460 **GevPrimaryApplicationIPAddress**

```
GenicamFeature<int64_t>* FliSfncCamera::GevPrimaryApplicationIPAddress
```

Returns the address of the primary application.

6.14.3.461 **GevPrimaryApplicationSocket**

```
GenicamFeature<int64_t>* FliSfncCamera::GevPrimaryApplicationSocket
```

Returns the UDP source port of the primary application.

6.14.3.462 **GevPrimaryApplicationSwitchoverKey**

```
GenicamFeature<int64_t>* FliSfncCamera::GevPrimaryApplicationSwitchoverKey
```

Controls the key to use to authenticate primary application switchover requests.

6.14.3.463 **GevSCCFGAllInTransmission**

```
GenicamFeature<bool>* FliSfncCamera::GevSCCFGAllInTransmission
```

Enables the selected GVSP transmitter to use the single packet per data block All-in Transmission mode.

6.14.3.464 **GevSCCFGExtendedChunkData**

```
GenicamFeature<bool>* FliSfncCamera::GevSCCFGExtendedChunkData
```

Enables cameras to use the extended chunk data payload type for this stream channel.

6.14.3.465 **GevSCCFGPacketResendDestination**

```
GenicamFeature<bool>* FliSfncCamera::GevSCCFGPacketResendDestination
```

Enables the alternate IP destination for stream packets resent due to a packet resend request. When True, the source IP address provided in the packet resend command packet is used. When False, the value set in the `GevSCDA[GevStreamChannelSelector]` feature is used.

6.14.3.466 **GevSCCFGUnconditionalStreaming**

```
GenicamFeature<bool>* FliSfncCamera::GevSCCFGUnconditionalStreaming
```

Enables the camera to continue to stream, for this stream channel, if its control channel is closed or regardless of the reception of any ICMP messages (such as destination unreachable messages).

6.14.3.467 **GevSCDA**

```
GenicamFeature<int64_t>* FliSfncCamera::GevSCDA
```

Controls the destination IP address of the selected stream channel to which a GVSP transmitter must send data stream or the destination IP address from which a GVSP receiver may receive data stream.

6.14.3.468 **GevSCPD**

```
GenicamFeature<int64_t>* FliSfncCamera::GevSCPD
```

Controls the delay (in GEV timestamp counter unit) to insert between each packet for this stream channel. This can be used as a crude flow-control mechanism if the application or the network infrastructure cannot keep up with the packets coming from the device.

6.14.3.469 **GevSCPHostPort**

```
GenicamFeature<int64_t>* FliSfncCamera::GevSCPHostPort
```

Controls the port of the selected channel to which a GVSP transmitter must send data stream or the port from which a GVSP receiver may receive data stream. Setting this value to 0 closes the stream channel.

6.14.3.470 **GevSCPIInterfaceIndex**

```
GenicamFeature<int64_t>* FliSfncCamera::GevSCPIInterfaceIndex
```

Index of the logical link to use.

6.14.3.471 **GevSCPSDoNotFragment**

```
GenicamFeature<bool>* FliSfncCamera::GevSCPSDoNotFragment
```

The state of this feature is copied into the "do not fragment" bit of IP header of each stream packet. It can be used by the application to prevent IP fragmentation of packets on the stream channel.

6.14.3.472 **GevSCPSFireTestPacket**

```
GenicamFeature<bool>* FliSfncCamera::GevSCPSFireTestPacket
```

Sends a test packet. When this feature is set, the device will fire one test packet.

6.14.3.473 **GevSCPSPacketSize**

```
GenicamFeature<int64_t>* FliSfncCamera::GevSCPSPacketSize
```

This GigE Vision specific feature corresponds to DeviceStreamChannelPacketSize and should be kept in sync with it. It specifies the stream packet size, in bytes, to send on the selected channel for a GVSP transmitter or specifies the maximum packet size supported by a GVSP receiver.

6.14.3.474 **GevSCSP**

```
GenicamFeature<int64_t>* FliSfncCamera::GevSCSP
```

Indicates the source port of the stream channel.

6.14.3.475 **GevSCZoneConfigurationLock**

```
GenicamFeature<bool>* FliSfncCamera::GevSCZoneConfigurationLock
```

Controls whether the selected stream channel multi-zone configuration is locked. When locked, the GVSP transmitter is not allowed to change the number of zones and their direction during block acquisition and transmission.

6.14.3.476 **GevSCZoneCount**

```
GenicamFeature<int64_t>* FliSfncCamera::GevSCZoneCount
```

Reports the number of zones per block transmitted on the selected stream channel.

6.14.3.477 **GevSCZoneDirectionAll**

```
GenicamFeature<int64_t>* FliSfncCamera::GevSCZoneDirectionAll
```

Reports the transmission direction of each zone transmitted on the selected stream channel.

6.14.3.478 **GevSecondURL**

```
GenicamFeature<std::string>* FliSfncCamera::GevSecondURL
```

Indicates the second URL to the GenICam XML device description file. This URL is an alternative if the application was unsuccessful to retrieve the device description file using the first URL.

6.14.3.479 **GevStreamChannelSelector**

```
GenicamFeature<int64_t>* FliSfncCamera::GevStreamChannelSelector
```

Selects the stream channel to control.

6.14.3.480 **GevSupportedOption**

```
GenicamFeature<bool>* FliSfncCamera::GevSupportedOption
```

Returns if the selected GEV option is supported.

6.14.3.481 `GevSupportedOptionSelector`

```
GenicamFeature<FliSfncCameraEnum::GevSupportedOptionSelectorEnum>* FliSfncCamera::GevSupportedOptionSelector
```

Selects the GEV option to interrogate for existing support.

6.14.3.482 `GroupIDValue`

```
GenicamFeature<int64_t>* FliSfncCamera::GroupIDValue
```

Returns a unique Identifier value corresponding to the selected Group of Components. If no grouping is required, this value should be set to 0.

6.14.3.483 `GroupSelector`

```
GenicamFeature<FliSfncCameraEnum::GroupSelectorEnum>* FliSfncCamera::GroupSelector
```

Selects a Group of component to control or inquire. The `GroupSelector` determines which components Group will be used for the selected features.

6.14.3.484 `Height`

```
GenicamFeature<int64_t>* FliSfncCamera::Height
```

Height of the image provided by the device (in pixels).

6.14.3.485 `HeightMax`

```
GenicamFeature<int64_t>* FliSfncCamera::HeightMax
```

Maximum height of the image (in pixels). This dimension is calculated after vertical binning, decimation or any other function changing the vertical dimension of the image.

6.14.3.486 `ImageCompressionBitrate`

```
GenicamFeature<double>* FliSfncCamera::ImageCompressionBitrate
```

Control the rate of the produced compressed stream.

6.14.3.487 ImageCompressionJPEGFormatOption

```
GenicamFeature<FliSfncCameraEnum::ImageCompressionJPEGFormatOptionEnum>* FliSfncCamera::ImageCompressionJPEGFormatOption
```

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

6.14.3.488 ImageCompressionMode

```
GenicamFeature<FliSfncCameraEnum::ImageCompressionModeEnum>* FliSfncCamera::ImageCompressionMode
```

Enable a specific image compression mode as the base mode for image transfer. Optionally, chunk data can be appended to the compressed image (See the REF [_Ref397502619](#) \h chapter).

6.14.3.489 ImageCompressionQuality

```
GenicamFeature<int64_t>* FliSfncCamera::ImageCompressionQuality
```

Control the quality of the produced compressed stream.

6.14.3.490 ImageCompressionRateOption

```
GenicamFeature<FliSfncCameraEnum::ImageCompressionRateOptionEnum>* FliSfncCamera::ImageCompressionRateOption
```

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

6.14.3.491 LightBrightness

```
GenicamFeature<double>* FliSfncCamera::LightBrightness
```

Set the brightness of the lighting output in percent. Can be greater than 100% for short overdrive period.

6.14.3.492 LightConnectionStatus

```
GenicamFeature<FliSfncCameraEnum::LightConnectionStatusEnum>* FliSfncCamera::LightConnection↔  
Status
```

Status of a light connected to the controller's output Line.

6.14.3.493 LightControllerSelector

```
GenicamFeature<FliSfncCameraEnum::LightControllerSelectorEnum>* FliSfncCamera::LightController↔  
Selector
```

Selects the Light Controller to configure.

6.14.3.494 LightControllerSource

```
GenicamFeature<FliSfncCameraEnum::LightControllerSourceEnum>* FliSfncCamera::LightController↔  
Source
```

Selects the input source signal of the Light Controller.

6.14.3.495 LightCurrentMeasured

```
GenicamFeature<double>* FliSfncCamera::LightCurrentMeasured
```

The measured current applied to the lighting.

6.14.3.496 LightCurrentRating

```
GenicamFeature<double>* FliSfncCamera::LightCurrentRating
```

Set the current rating of the lighting output.

6.14.3.497 LightVoltageMeasured

```
GenicamFeature<double>* FliSfncCamera::LightVoltageMeasured
```

The measured voltage applied to the lighting.

6.14.3.498 LightVoltageRating

```
GenicamFeature<double>* FliSfncCamera::LightVoltageRating
```

Set the voltage rating of the lighting output.

6.14.3.499 LineFormat

```
GenicamFeature<FliSfncCameraEnum::LineFormatEnum>* FliSfncCamera::LineFormat
```

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

6.14.3.500 LineInverter

```
GenicamFeature<bool>* FliSfncCamera::LineInverter
```

Controls the inversion of the signal of the selected input or output Line.

6.14.3.501 LineMode

```
GenicamFeature<FliSfncCameraEnum::LineModeEnum>* FliSfncCamera::LineMode
```

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

6.14.3.502 LinePitch

```
GenicamFeature<int64_t>* FliSfncCamera::LinePitch
```

Total number of bytes between the starts of 2 consecutive lines. This feature is used to facilitate alignment of image data.

6.14.3.503 LinePitchEnable

```
GenicamFeature<bool>* FliSfncCamera::LinePitchEnable
```

This feature controls whether the LinePitch feature is writable. Otherwise LinePitch is implicitly controlled by the combination of features like Width, PixelFormat, etc...

6.14.3.504 LineSelector

```
GenicamFeature<FliSfncCameraEnum::LineSelectorEnum>* FliSfncCamera::LineSelector
```

Selects the physical line (or pin) of the external device connector or the virtual line of the Transport Layer to configure.

6.14.3.505 LineSource

```
GenicamFeature<FliSfncCameraEnum::LineSourceEnum>* FliSfncCamera::LineSource
```

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

6.14.3.506 LineStatus

```
GenicamFeature<bool>* FliSfncCamera::LineStatus
```

Returns the current status of the selected input or output Line.

6.14.3.507 LineStatusAll

```
GenicamFeature<int64_t>* FliSfncCamera::LineStatusAll
```

Returns the current status of all available Line signals at time of polling in a single bitfield.

6.14.3.508 LogicBlockFunction

```
GenicamFeature<FliSfncCameraEnum::LogicBlockFunctionEnum>* FliSfncCamera::LogicBlockFunction
```

Selects the combinational logic Function of the Logic Block to configure.

6.14.3.509 LogicBlockInputInverter

```
GenicamFeature<bool>* FliSfncCamera::LogicBlockInputInverter
```

Selects if the selected Logic Block Input source signal is inverted. This feature is not available when the LogicBlockInputSource is set to True or False.

6.14.3.510 LogicBlockInputNumber

```
GenicamFeature<int64_t>* FliSfncCamera::LogicBlockInputNumber
```

Specifies the number of active signal inputs of the Logic Block.

6.14.3.511 LogicBlockInputSelector

```
GenicamFeature<int64_t>* FliSfncCamera::LogicBlockInputSelector
```

Selects the Logic Block's input to configure.

6.14.3.512 LogicBlockInputSource

```
GenicamFeature<FliSfncCameraEnum::LogicBlockInputSourceEnum>* FliSfncCamera::LogicBlockInput↔  
Source
```

Selects the source signal for the input into the Logic Block. True or False indicates the input is forced constant.

6.14.3.513 LogicBlockLUTIndex

```
GenicamFeature<int64_t>* FliSfncCamera::LogicBlockLUTIndex
```

Controls the index of the truth table to access in the selected LUT.

6.14.3.514 LogicBlockLUTSelector

```
GenicamFeature<FliSfncCameraEnum::LogicBlockLUTSelectorEnum>* FliSfncCamera::LogicBlockLUT↔  
Selector
```

Selects which of the two LUTs to configure when the selected Logic Block is a Latched dual LUTs (i.e: Logical↔ BlockFunction = LatchedLUT).

6.14.3.515 LogicBlockLUTValue

```
GenicamFeature<bool>* FliSfncCamera::LogicBlockLUTValue
```

Read or Write the Value associated with the entry at index LogicBlockLUTIndex of the selected LUT.

6.14.3.516 LogicBlockLUTValueAll

```
GenicamFeature<int64_t>* FliSfncCamera::LogicBlockLUTValueAll
```

Sets the values of all the output bits of the selected LUT in one access ignoring LogicBlockLUTIndex. LogicBlockLUTValueAll value can be any binary number and each bit correspond to the output value for the corresponding index (i.e. Bit 0 = LUT Index 0 output binary value).

6.14.3.517 LogicBlockSelector

```
GenicamFeature<FliSfncCameraEnum::LogicBlockSelectorEnum>* FliSfncCamera::LogicBlockSelector
```

Specifies the Logic Block to configure.

6.14.3.518 LUTEnable

```
GenicamFeature<bool>* FliSfncCamera::LUTEnable
```

Activates the selected LUT.

6.14.3.519 LUTIndex

```
GenicamFeature<int64_t>* FliSfncCamera::LUTIndex
```

Control the index (offset) of the coefficient to access in the selected LUT.

6.14.3.520 LUTSelector

```
GenicamFeature<FliSfncCameraEnum::LUTSelectorEnum>* FliSfncCamera::LUTSelector
```

Selects which LUT to control.

6.14.3.521 LUTValue

```
GenicamFeature<int64_t>* FliSfncCamera::LUTValue
```

Returns the Value at entry LUTIndex of the LUT selected by LUTSelector.

6.14.3.522 MultiSlopeExposureGradient

```
GenicamFeature<double>* FliSfncCamera::MultiSlopeExposureGradient
```

The gradient of the additional slope that is defined by this knee-point.

6.14.3.523 MultiSlopeExposureLimit

```
GenicamFeature<double>* FliSfncCamera::MultiSlopeExposureLimit
```

Percent of the ExposureTime at a certain knee-point of multi-slope exposure.

6.14.3.524 MultiSlopeIntensityLimit

```
GenicamFeature<double>* FliSfncCamera::MultiSlopeIntensityLimit
```

The relative intensity which divides intensities influenced by different exposure slopes.

6.14.3.525 MultiSlopeKneePointCount

```
GenicamFeature<int64_t>* FliSfncCamera::MultiSlopeKneePointCount
```

The number of knee-points as well as the number of additional exposure slopes used for multi-slope exposure.

6.14.3.526 MultiSlopeKneePointSelector

```
GenicamFeature<int64_t>* FliSfncCamera::MultiSlopeKneePointSelector
```

Selects the parameters for controlling an additional slope in multi-slope exposure.

6.14.3.527 MultiSlopeMode

```
GenicamFeature<FliSfncCameraEnum::MultiSlopeModeEnum>* FliSfncCamera::MultiSlopeMode
```

Controls multi-slope exposure state.

6.14.3.528 MultiSlopeSaturationThreshold

```
GenicamFeature<double>* FliSfncCamera::MultiSlopeSaturationThreshold
```

The percentage of the full saturation that is applied at a certain knee-point of a multi-slope exposure.

6.14.3.529 OffsetX

```
GenicamFeature<int64_t>* FliSfncCamera::OffsetX
```

Horizontal offset from the origin to the region of interest (in pixels).

6.14.3.530 OffsetY

```
GenicamFeature<int64_t>* FliSfncCamera::OffsetY
```

Vertical offset from the origin to the region of interest (in pixels).

6.14.3.531 PayloadSize

```
GenicamFeature<int64_t>* FliSfncCamera::PayloadSize
```

Provides the number of bytes transferred for each data buffer or chunk on the stream channel. This includes any end-of-line, end-of-frame statistics or other stamp data. This is the total size of data payload for a data block.

6.14.3.532 PixelColorFilter

```
GenicamFeature<FliSfncCameraEnum::PixelColorFilterEnum>* FliSfncCamera::PixelColorFilter
```

Type of color filter that is applied to the image.

6.14.3.533 PixelDynamicRangeMax

```
GenicamFeature<int64_t>* FliSfncCamera::PixelDynamicRangeMax
```

Maximum value that will be returned during the digitization process. This corresponds to the brightest value of the camera. For color camera, this returns the biggest value that each color component can take.

6.14.3.534 PixelDynamicRangeMin

```
GenicamFeature<int64_t>* FliSfncCamera::PixelDynamicRangeMin
```

Minimum value that can be returned during the digitization process. This corresponds to the darkest value of the camera. For color camera, this returns the smallest value that each color component can take.

6.14.3.535 PixelFormat

```
GenicamFeature<FliSfncCameraEnum::PixelFormatEnum>* FliSfncCamera::PixelFormat
```

Format of the pixels provided by the device. It represents all the information provided by PixelSize, PixelColorFilter combined in a single feature.

6.14.3.536 PixelFormatInfoID

```
GenicamFeature<int64_t>* FliSfncCamera::PixelFormatInfoID
```

Returns the value used by the streaming channels to identify the selected pixel format.

6.14.3.537 PixelFormatInfoSelector

```
GenicamFeature<FliSfncCameraEnum::PixelFormatInfoSelectorEnum>* FliSfncCamera::PixelFormatInfoSelector
```

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

6.14.3.538 PixelSize

```
GenicamFeature<FliSfncCameraEnum::PixelSizeEnum>* FliSfncCamera::PixelSize
```

Total size in bits of a pixel of the image.

6.14.3.539 PtpClockAccuracy

```
GenicamFeature<FliSfncCameraEnum::PtpClockAccuracyEnum>* FliSfncCamera::PtpClockAccuracy
```

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

6.14.3.540 PtpClockID

```
GenicamFeature<int64_t>* FliSfncCamera::PtpClockID
```

Returns the latched clock ID of the PTP device.

6.14.3.541 PtpDataSetLatch

```
GenicamFeature* FliSfncCamera::PtpDataSetLatch
```

Latches the current values from the device's PTP clock data set.

6.14.3.542 PtpEnable

```
GenicamFeature<bool>* FliSfncCamera::PtpEnable
```

Enables the Precision Time Protocol (PTP).

6.14.3.543 PtpGrandmasterClockID

```
GenicamFeature<int64_t>* FliSfncCamera::PtpGrandmasterClockID
```

Returns the latched grandmaster clock ID of the PTP device. The grandmaster clock ID is the clock ID of the current grandmaster clock.

6.14.3.544 PtpOffsetFromMaster

```
GenicamFeature<int64_t>* FliSfncCamera::PtpOffsetFromMaster
```

Returns the latched offset from the PTP master clock in nanoseconds.

6.14.3.545 PtpParentClockID

```
GenicamFeature<int64_t>* FliSfncCamera::PtpParentClockID
```

Returns the latched parent clock ID of the PTP device. The parent clock ID is the clock ID of the current master clock.

6.14.3.546 PtpServoStatus

```
GenicamFeature<FliSfncCameraEnum::PtpServoStatusEnum>* FliSfncCamera::PtpServoStatus
```

Returns the latched state of the clock servo. When the servo is in a locked state, the value returned is 'Locked'. When the servo is in a non-locked state, a device-specific value can be returned to give specific information. If no device-specific value is available to describe the current state of the clock servo, the value should be 'Unknown'.

6.14.3.547 PtpStatus

```
GenicamFeature<FliSfncCameraEnum::PtpStatusEnum>* FliSfncCamera::PtpStatus
```

Returns the latched state of the PTP clock.

6.14.3.548 RegionDestination

```
GenicamFeature<FliSfncCameraEnum::RegionDestinationEnum>* FliSfncCamera::RegionDestination
```

Control the destination of the selected region.

6.14.3.549 RegionIDValue

```
GenicamFeature<int64_t>* FliSfncCamera::RegionIDValue
```

Returns a unique Identifier value that corresponds to the selected Region.

6.14.3.550 RegionMode

```
GenicamFeature<FliSfncCameraEnum::RegionModeEnum>* FliSfncCamera::RegionMode
```

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

6.14.3.551 RegionSelector

```
GenicamFeature<FliSfncCameraEnum::RegionSelectorEnum>* FliSfncCamera::RegionSelector
```

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.

6.14.3.552 ReverseX

```
GenicamFeature<bool>* FliSfncCamera::ReverseX
```

Flip horizontally the image sent by the device. The Region of interest is applied after the flipping.

6.14.3.553 ReverseY

```
GenicamFeature<bool>* FliSfncCamera::ReverseY
```

Flip vertically the image sent by the device. The Region of interest is applied after the flipping.

6.14.3.554 Scan3dAxisMax

```
GenicamFeature<double>* FliSfncCamera::Scan3dAxisMax
```

Maximum valid transmitted coordinate value of the selected Axis.

6.14.3.555 Scan3dAxisMin

```
GenicamFeature<double>* FliSfncCamera::Scan3dAxisMin
```

Minimum valid transmitted coordinate value of the selected Axis.

6.14.3.556 Scan3dBaseline

```
GenicamFeature<double>* FliSfncCamera::Scan3dBaseline
```

Returns the baseline as the physical distance of two cameras in a stereo camera setup. The value of this feature can be used for 3D reconstruction from disparity images. In this case, the unit of the 3D coordinates corresponds to the unit of the baseline.

6.14.3.557 Scan3dCoordinateOffset

```
GenicamFeature<double>* FliSfncCamera::Scan3dCoordinateOffset
```

Offset when transforming a pixel from relative coordinates to world coordinates.

6.14.3.558 Scan3dCoordinateReferenceSelector

```
GenicamFeature<FliSfncCameraEnum::Scan3dCoordinateReferenceSelectorEnum>* FliSfncCamera::↔  
Scan3dCoordinateReferenceSelector
```

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.

6.14.3.559 Scan3dCoordinateReferenceValue

```
GenicamFeature<double>* FliSfncCamera::Scan3dCoordinateReferenceValue
```

Returns the reference value selected. Reads the value of a rotation or translation value for the current (Anchor or Transformed) coordinate system transformation to the Reference system.

6.14.3.560 Scan3dCoordinateScale

```
GenicamFeature<double>* FliSfncCamera::Scan3dCoordinateScale
```

Scale factor when transforming a pixel from relative coordinates to world coordinates.

6.14.3.561 Scan3dCoordinateSelector

```
GenicamFeature<FliSfncCameraEnum::Scan3dCoordinateSelectorEnum>* FliSfncCamera::Scan3dCoordinate↔  
Selector
```

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

6.14.3.562 Scan3dCoordinateSystem

```
GenicamFeature<FliSfncCameraEnum::Scan3dCoordinateSystemEnum>* FliSfncCamera::Scan3dCoordinate↔  
System
```

Specifies the Coordinate system to use for the device.

6.14.3.563 Scan3dCoordinateSystemReference

```
GenicamFeature<FliSfncCameraEnum::Scan3dCoordinateSystemReferenceEnum>* FliSfncCamera::↔  
Scan3dCoordinateSystemReference
```

Defines coordinate system reference location.

6.14.3.564 Scan3dCoordinateTransformSelector

```
GenicamFeature<FliSfncCameraEnum::Scan3dCoordinateTransformSelectorEnum>* FliSfncCamera::Scan3dCoordinateTransformSelector
```

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

6.14.3.565 Scan3dDistanceUnit

```
GenicamFeature<FliSfncCameraEnum::Scan3dDistanceUnitEnum>* FliSfncCamera::Scan3dDistanceUnit
```

Specifies the unit used when delivering (calibrated) distance data.

6.14.3.566 Scan3dExtractionMethod

```
GenicamFeature<FliSfncCameraEnum::Scan3dExtractionMethodEnum>* FliSfncCamera::Scan3dExtractionMethod
```

Selects the method for extracting 3D from the input sensor data.

6.14.3.567 Scan3dExtractionSelector

```
GenicamFeature<FliSfncCameraEnum::Scan3dExtractionSelectorEnum>* FliSfncCamera::Scan3dExtractionSelector
```

Selects the 3DExtraction processing module to control (if multiple ones are present).

6.14.3.568 Scan3dExtractionSource

```
GenicamFeature<FliSfncCameraEnum::Scan3dExtractionSourceEnum>* FliSfncCamera::Scan3dExtractionSource
```

Selects the sensor's data source region for 3D Extraction module.

6.14.3.569 Scan3dFocalLength

```
GenicamFeature<double>* FliSfncCamera::Scan3dFocalLength
```

Returns the focal length of the camera in pixel. The focal length depends on the selected region. The value of this feature takes into account horizontal binning, decimation, or any other function changing the image resolution.

6.14.3.570 Scan3dInvalidDataFlag

```
GenicamFeature<bool>* FliSfncCamera::Scan3dInvalidDataFlag
```

Enables the definition of a non-valid flag value in the data stream. Note that the confidence output is an alternate recommended way to identify non-valid pixels. Using a Scan3dInvalidDataValue may give processing penalties due to special handling.

6.14.3.571 Scan3dInvalidDataValue

```
GenicamFeature<double>* FliSfncCamera::Scan3dInvalidDataValue
```

Value which identifies a non-valid pixel if Scan3dInvalidDataFlag is enabled.

6.14.3.572 Scan3dOutputMode

```
GenicamFeature<FliSfncCameraEnum::Scan3dOutputModeEnum>* FliSfncCamera::Scan3dOutputMode
```

Controls the Calibration and data organization of the device and the coordinates transmitted.

6.14.3.573 Scan3dPrincipalPointU

```
GenicamFeature<double>* FliSfncCamera::Scan3dPrincipalPointU
```

Returns the value of the horizontal position of the principal point, relative to the region origin, i.e. OffsetX. The value of this feature takes into account horizontal binning, decimation, or any other function changing the image resolution.

6.14.3.574 Scan3dPrincipalPointV

```
GenicamFeature<double>* FliSfncCamera::Scan3dPrincipalPointV
```

Returns the value of the vertical position of the principal point, relative to the region origin, i.e. OffsetY. The value of this feature takes into account vertical binning, decimation, or any other function changing the image resolution.

6.14.3.575 Scan3dTransformValue

```
GenicamFeature<double>* FliSfncCamera::Scan3dTransformValue
```

Specifies the transform value selected. For translations (Scan3dCoordinateTransformSelector = TranslationX/Y/Z) it is expressed in the distance unit of the system, for rotations (Scan3dCoordinateTransformSelector =RotationX/Y/Z) in degrees.

6.14.3.576 SensorDigitizationTaps

```
GenicamFeature<FliSfncCameraEnum::SensorDigitizationTapsEnum>* FliSfncCamera::SensorDigitization↔  
Taps
```

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

6.14.3.577 SensorHeight

```
GenicamFeature<int64_t>* FliSfncCamera::SensorHeight
```

Effective height of the sensor in pixels.

6.14.3.578 SensorName

```
GenicamFeature<std::string>* FliSfncCamera::SensorName
```

Product name of the imaging Sensor.

6.14.3.579 SensorPixelHeight

```
GenicamFeature<double>* FliSfncCamera::SensorPixelHeight
```

Physical size (pitch) in the y direction of a photo sensitive pixel unit.

6.14.3.580 SensorPixelWidth

```
GenicamFeature<double>* FliSfncCamera::SensorPixelWidth
```

Physical size (pitch) in the x direction of a photo sensitive pixel unit.

6.14.3.581 SensorShutterMode

```
GenicamFeature<FliSfncCameraEnum::SensorShutterModeEnum>* FliSfncCamera::SensorShutterMode
```

Specifies the shutter mode of the device.

6.14.3.582 SensorTaps

```
GenicamFeature<FliSfncCameraEnum::SensorTapsEnum>* FliSfncCamera::SensorTaps
```

Number of taps of the camera sensor.

6.14.3.583 SensorWidth

```
GenicamFeature<int64_t>* FliSfncCamera::SensorWidth
```

Effective width of the sensor in pixels.

6.14.3.584 SequencerConfigurationMode

```
GenicamFeature<FliSfncCameraEnum::SequencerConfigurationModeEnum>* FliSfncCamera::Sequencer↔  
ConfigurationMode
```

Controls if the sequencer configuration mode is active.

6.14.3.585 SequencerFeatureEnable

```
GenicamFeature<bool>* FliSfncCamera::SequencerFeatureEnable
```

Enables the selected feature and make it active in all the sequencer sets.

6.14.3.586 SequencerFeatureSelector

```
GenicamFeature<FliSfncCameraEnum::SequencerFeatureSelectorEnum>* FliSfncCamera::Sequencer↔  
FeatureSelector
```

Selects which sequencer features to control.

6.14.3.587 SequencerMode

GenicamFeature<FliSfncCameraEnum::SequencerModeEnum>* FliSfncCamera::SequencerMode

Controls if the sequencer mechanism is active.

6.14.3.588 SequencerPathSelector

GenicamFeature<int64_t>* FliSfncCamera::SequencerPathSelector

Selects to which branching path further path settings applies.

6.14.3.589 SequencerSetActive

GenicamFeature<int64_t>* FliSfncCamera::SequencerSetActive

Contains the currently active sequencer set.

6.14.3.590 SequencerSetLoad

GenicamFeature* FliSfncCamera::SequencerSetLoad

Loads the sequencer set selected by SequencerSetSelector in the device. Even if SequencerMode is off, this will change the device state to the configuration of the selected set.

6.14.3.591 SequencerSetNext

GenicamFeature<int64_t>* FliSfncCamera::SequencerSetNext

Specifies the next sequencer set.

6.14.3.592 SequencerSetSave

GenicamFeature* FliSfncCamera::SequencerSetSave

Saves the current device state to the sequencer set selected by the SequencerSetSelector.

6.14.3.593 SequencerSetSelector

```
GenicamFeature<int64_t>* FliSfncCamera::SequencerSetSelector
```

Selects the sequencer set to which further feature settings applies.

6.14.3.594 SequencerSetStart

```
GenicamFeature<int64_t>* FliSfncCamera::SequencerSetStart
```

Sets the initial/start sequencer set, which is the first set used within a sequencer.

6.14.3.595 SequencerTriggerActivation

```
GenicamFeature<FliSfncCameraEnum::SequencerTriggerActivationEnum>* FliSfncCamera::Sequencer↔  
TriggerActivation
```

Specifies the activation mode of the sequencer trigger.

6.14.3.596 SequencerTriggerSource

```
GenicamFeature<FliSfncCameraEnum::SequencerTriggerSourceEnum>* FliSfncCamera::Sequencer↔  
TriggerSource
```

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

6.14.3.597 SoftwareSignalPulse

```
GenicamFeature* FliSfncCamera::SoftwareSignalPulse
```

Generates a pulse signal that can be used as a software trigger. This command can be used to trigger other modules that accept a SoftwareSignal as trigger source.

6.14.3.598 SoftwareSignalSelector

```
GenicamFeature<FliSfncCameraEnum::SoftwareSignalSelectorEnum>* FliSfncCamera::SoftwareSignal↔  
Selector
```

Selects which Software Signal features to control.

6.14.3.599 SourceCount

```
GenicamFeature<int64_t>* FliSfncCamera::SourceCount
```

Controls or returns the number of sources supported by the device.

6.14.3.600 SourceIDValue

```
GenicamFeature<int64_t>* FliSfncCamera::SourceIDValue
```

Returns a unique Identifier value that correspond to the selected Source.

6.14.3.601 SourceSelector

```
GenicamFeature<FliSfncCameraEnum::SourceSelectorEnum>* FliSfncCamera::SourceSelector
```

Selects the source to control.

6.14.3.602 TestEventGenerate

```
GenicamFeature* FliSfncCamera::TestEventGenerate
```

Generates a Test Event.

6.14.3.603 TestPattern

```
GenicamFeature<FliSfncCameraEnum::TestPatternEnum>* FliSfncCamera::TestPattern
```

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

6.14.3.604 TestPatternGeneratorSelector

```
GenicamFeature<FliSfncCameraEnum::TestPatternGeneratorSelectorEnum>* FliSfncCamera::Test↔  
PatternGeneratorSelector
```

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

6.14.3.605 TestPayloadFormatMode

```
GenicamFeature<FliSfncCameraEnum::TestPayloadFormatModeEnum>* FliSfncCamera::TestPayloadFormatMode
```

This feature allows setting a device in test mode and to output a specific payload format for validation of data streaming. This feature is intended solely for test purposes. The data can be real acquired data or any test pattern.

6.14.3.606 TestPendingAck

```
GenicamFeature<int64_t>* FliSfncCamera::TestPendingAck
```

Tests the device's pending acknowledge feature. When this feature is written, the device waits a time period corresponding to the value of TestPendingAck before acknowledging the write.

6.14.3.607 TimerDelay

```
GenicamFeature<double>* FliSfncCamera::TimerDelay
```

Sets the duration (in microseconds) of the delay to apply at the reception of a trigger before starting the Timer.

6.14.3.608 TimerDuration

```
GenicamFeature<double>* FliSfncCamera::TimerDuration
```

Sets the duration (in microseconds) of the Timer pulse.

6.14.3.609 TimerReset

```
GenicamFeature* FliSfncCamera::TimerReset
```

Does a software reset of the selected timer and starts it. The timer starts immediately after the reset unless a timer trigger is active.

6.14.3.610 TimerSelector

```
GenicamFeature<FliSfncCameraEnum::TimerSelectorEnum>* FliSfncCamera::TimerSelector
```

Selects which Timer to configure.

6.14.3.611 TimerStatus

```
GenicamFeature<FliSfncCameraEnum::TimerStatusEnum>* FliSfncCamera::TimerStatus
```

Returns the current status of the Timer.

6.14.3.612 TimerTriggerActivation

```
GenicamFeature<FliSfncCameraEnum::TimerTriggerActivationEnum>* FliSfncCamera::TimerTrigger←  
Activation
```

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

6.14.3.613 TimerTriggerArmDelay

```
GenicamFeature<double>* FliSfncCamera::TimerTriggerArmDelay
```

Sets the minimum period between two valid timer triggers.

6.14.3.614 TimerTriggerSource

```
GenicamFeature<FliSfncCameraEnum::TimerTriggerSourceEnum>* FliSfncCamera::TimerTriggerSource
```

Selects the source of the trigger to start the Timer.

6.14.3.615 TimerValue

```
GenicamFeature<double>* FliSfncCamera::TimerValue
```

Reads or writes the current value (in microseconds) of the selected Timer.

6.14.3.616 Timestamp

```
GenicamFeature<int64_t>* FliSfncCamera::Timestamp
```

Reports the current value of the device timestamp counter.

6.14.3.617 TimestampLatch

```
GenicamFeature* FliSfncCamera::TimestampLatch
```

Latches the current timestamp counter into TimestampLatchValue.

6.14.3.618 TimestampLatchValue

```
GenicamFeature<int64_t>* FliSfncCamera::TimestampLatchValue
```

Returns the latched value of the timestamp counter.

6.14.3.619 TimestampReset

```
GenicamFeature* FliSfncCamera::TimestampReset
```

Resets the current value of the device timestamp counter.

6.14.3.620 TLParamsLocked

```
GenicamFeature<int64_t>* FliSfncCamera::TLParamsLocked
```

Used by the Transport Layer to prevent critical features from changing during acquisition.

6.14.3.621 TLParamsLockedSelector

```
GenicamFeature<FliSfncCameraEnum::TLParamsLockedSelectorEnum>* FliSfncCamera::TLParamsLockedSelector
```

Selects the type of feature for which the locking behavior will be configured.

6.14.3.622 TLParamsLockedState

```
GenicamFeature<bool>* FliSfncCamera::TLParamsLockedState
```

Controls if the selected parameters are locked during acquisition.

6.14.3.623 TransferAbort

```
GenicamFeature* FliSfncCamera::TransferAbort
```

Aborts immediately the streaming of data block(s). Aborting the transfer will result in the lost of the data that is present or currently entering in the block queue. However, the next new block received will be stored in the queue and transferred to the host when the streaming is restarted. If implemented, this feature should be available when the TransferControlMode is set to "UserControlled".

6.14.3.624 TransferBlockCount

```
GenicamFeature<int64_t>* FliSfncCamera::TransferBlockCount
```

Specifies the number of data Blocks that the device should stream before stopping. This feature is only active if the TransferOperationMode is set to MultiBlock.

6.14.3.625 TransferBurstCount

```
GenicamFeature<int64_t>* FliSfncCamera::TransferBurstCount
```

Number of Block(s) to transfer for each TransferBurstStart trigger.

6.14.3.626 TransferComponentSelector

```
GenicamFeature<FliSfncCameraEnum::TransferComponentSelectorEnum>* FliSfncCamera::TransferComponentSelector
```

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

6.14.3.627 TransferControlMode

```
GenicamFeature<FliSfncCameraEnum::TransferControlModeEnum>* FliSfncCamera::TransferControlMode
```

Selects the control method for the transfers.

6.14.3.628 TransferOperationMode

```
GenicamFeature<FliSfncCameraEnum::TransferOperationModeEnum>* FliSfncCamera::TransferOperationMode
```

Selects the operation mode of the transfer.

6.14.3.629 TransferPause

```
GenicamFeature* FliSfncCamera::TransferPause
```

Pauses the streaming of data Block(s). Pausing the streaming will immediately suspend the ongoing data transfer even if a block is partially transferred. The device will resume its transmission at the reception of a TransferResume command.

6.14.3.630 TransferQueueCurrentBlockCount

```
GenicamFeature<int64_t>* FliSfncCamera::TransferQueueCurrentBlockCount
```

Returns the number of Block(s) currently in the transfer queue.

6.14.3.631 TransferQueueMaxBlockCount

```
GenicamFeature<int64_t>* FliSfncCamera::TransferQueueMaxBlockCount
```

Controls the maximum number of data blocks that can be stored in the block queue of the selected stream.

6.14.3.632 TransferQueueMode

```
GenicamFeature<FliSfncCameraEnum::TransferQueueModeEnum>* FliSfncCamera::TransferQueueMode
```

Specifies the operation mode of the transfer queue.

6.14.3.633 TransferResume

```
GenicamFeature* FliSfncCamera::TransferResume
```

Resumes a data Blocks streaming that was previously paused by a TransferPause command.

6.14.3.634 TransferSelector

```
GenicamFeature<FliSfncCameraEnum::TransferSelectorEnum>* FliSfncCamera::TransferSelector
```

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

6.14.3.635 TransferStart

```
GenicamFeature* FliSfncCamera::TransferStart
```

Starts the streaming of data blocks out of the device. This feature must be available when the TransferControlMode is set to "UserControlled". If the TransferStart feature is not writable (locked), the application should not start the transfer and should avoid using the feature until it becomes writable again.

6.14.3.636 TransferStatus

```
GenicamFeature<bool>* FliSfncCamera::TransferStatus
```

Reads the status of the Transfer module signal selected by TransferStatusSelector.

6.14.3.637 TransferStatusSelector

```
GenicamFeature<FliSfncCameraEnum::TransferStatusSelectorEnum>* FliSfncCamera::TransferStatus←  
Selector
```

Selects which status of the transfer module to read.

6.14.3.638 TransferStop

```
GenicamFeature* FliSfncCamera::TransferStop
```

Stops the streaming of data Block(s). The current block transmission will be completed. This feature must be available when the TransferControlMode is set to "UserControlled".

6.14.3.639 TransferStreamChannel

```
GenicamFeature<int64_t>* FliSfncCamera::TransferStreamChannel
```

Selects the streaming channel that will be used to transfer the selected stream of data. In general, this feature can be omitted and the default streaming channel will be used.

6.14.3.640 TransferTriggerActivation

```
GenicamFeature<FliSfncCameraEnum::TransferTriggerActivationEnum>* FliSfncCamera::Transfer←  
TriggerActivation
```

Specifies the activation mode of the transfer control trigger.

6.14.3.641 TransferTriggerMode

```
GenicamFeature<FliSfncCameraEnum::TransferTriggerModeEnum>* FliSfncCamera::TransferTriggerMode
```

Controls if the selected trigger is active.

6.14.3.642 TransferTriggerSelector

```
GenicamFeature<FliSfncCameraEnum::TransferTriggerSelectorEnum>* FliSfncCamera::Transfer↔  
TriggerSelector
```

Selects the type of transfer trigger to configure.

6.14.3.643 TransferTriggerSource

```
GenicamFeature<FliSfncCameraEnum::TransferTriggerSourceEnum>* FliSfncCamera::TransferTrigger↔  
Source
```

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

6.14.3.644 TriggerActivation

```
GenicamFeature<FliSfncCameraEnum::TriggerActivationEnum>* FliSfncCamera::TriggerActivation
```

Specifies the activation mode of the trigger.

6.14.3.645 TriggerDelay

```
GenicamFeature<double>* FliSfncCamera::TriggerDelay
```

Specifies the delay in microseconds (us) to apply after the trigger reception before activating it.

6.14.3.646 TriggerDivider

```
GenicamFeature<int64_t>* FliSfncCamera::TriggerDivider
```

Specifies a division factor for the incoming trigger pulses.

6.14.3.647 TriggerMode

```
GenicamFeature<FliSfncCameraEnum::TriggerModeEnum>* FliSfncCamera::TriggerMode
```

Controls if the selected trigger is active.

6.14.3.648 TriggerMultiplier

```
GenicamFeature<int64_t>* FliSfncCamera::TriggerMultiplier
```

Specifies a multiplication factor for the incoming trigger pulses. It is generally used in conjunction with TriggerDivider to control the ratio of triggers that are accepted.

6.14.3.649 TriggerOverlap

```
GenicamFeature<FliSfncCameraEnum::TriggerOverlapEnum>* FliSfncCamera::TriggerOverlap
```

Specifies the type trigger overlap permitted with the previous frame or line. This defines when a valid trigger will be accepted (or latched) for a new frame or a new line.

6.14.3.650 TriggerSelector

```
GenicamFeature<FliSfncCameraEnum::TriggerSelectorEnum>* FliSfncCamera::TriggerSelector
```

Selects the type of trigger to configure.

6.14.3.651 TriggerSoftware

```
GenicamFeature* FliSfncCamera::TriggerSoftware
```

Generates an internal trigger. TriggerSource must be set to Software.

6.14.3.652 TriggerSource

```
GenicamFeature<FliSfncCameraEnum::TriggerSourceEnum>* FliSfncCamera::TriggerSource
```

Specifies the internal signal or physical input Line to use as the trigger source. The selected trigger must have its TriggerMode set to On.

6.14.3.653 UserOutputSelector

```
GenicamFeature<FliSfncCameraEnum::UserOutputSelectorEnum>* FliSfncCamera::UserOutputSelector
```

Selects which bit of the User Output register will be set by UserOutputValue.

6.14.3.654 UserOutputValue

```
GenicamFeature<bool>* FliSfncCamera::UserOutputValue
```

Sets the value of the bit selected by UserOutputSelector.

6.14.3.655 UserOutputValueAll

```
GenicamFeature<int64_t>* FliSfncCamera::UserOutputValueAll
```

Sets the value of all the bits of the User Output register. It is subject to the UserOutputValueAllMask.

6.14.3.656 UserOutputValueAllMask

```
GenicamFeature<int64_t>* FliSfncCamera::UserOutputValueAllMask
```

Sets the write mask to apply to the value specified by UserOutputValueAll before writing it in the User Output register. If the UserOutputValueAllMask feature is present, setting the user Output register using UserOutputValueAll will only change the bits that have a corresponding bit in the mask set to one.

6.14.3.657 UserSetDefault

```
GenicamFeature<FliSfncCameraEnum::UserSetDefaultEnum>* FliSfncCamera::UserSetDefault
```

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

6.14.3.658 UserSetDescription

```
GenicamFeature<std::string>* FliSfncCamera::UserSetDescription
```

Description of the selected User Set content.

6.14.3.659 UserSetFeatureEnable

```
GenicamFeature<bool>* FliSfncCamera::UserSetFeatureEnable
```

Enables the selected feature and make it active in all the UserSets.

6.14.3.660 UserSetFeatureSelector

```
GenicamFeature<FliSfncCameraEnum::UserSetFeatureSelectorEnum>* FliSfncCamera::UserSetFeatureSelector
```

Selects which individual UserSet feature to control.

6.14.3.661 UserSetLoad

```
GenicamFeature* FliSfncCamera::UserSetLoad
```

Loads the User Set specified by UserSetSelector to the device and makes it active.

6.14.3.662 UserSetSave

```
GenicamFeature* FliSfncCamera::UserSetSave
```

Save the User Set specified by UserSetSelector to the non-volatile memory of the device.

6.14.3.663 UserSetSelector

```
GenicamFeature<FliSfncCameraEnum::UserSetSelectorEnum>* FliSfncCamera::UserSetSelector
```

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

6.14.3.664 WhiteClip

```
GenicamFeature<double>* FliSfncCamera::WhiteClip
```

Controls the maximal intensity taken by the video signal before being clipped as an absolute physical value. The video signal will never exceed the white clipping point: it will saturate at that level.

6.14.3.665 WhiteClipSelector

```
GenicamFeature<FliSfncCameraEnum::WhiteClipSelectorEnum>* FliSfncCamera::WhiteClipSelector
```

Selects which White Clip to control.

6.14.3.666 Width

```
GenicamFeature<int64_t>* FliSfncCamera::Width
```

Width of the image provided by the device (in pixels).

6.14.3.667 WidthMax

```
GenicamFeature<int64_t>* FliSfncCamera::WidthMax
```

Maximum width of the image (in pixels). The dimension is calculated after horizontal binning, decimation or any other function changing the horizontal dimension of the image.

6.15 IFliSdkObserver Class Reference

This interface defines an observer to observe some SDK states.

```
#include <IFliSdkObserver.h>
```

Public Member Functions

- virtual void [onStartedStateChanged](#) (bool started)
- virtual void [onGrabNStateChanged](#) (bool enabled, uint32_t nbFrames)
- virtual void [onResetBufferTriggered](#) ()
- virtual void [onCameraChanged](#) ()
- virtual void [onFowlerProcessingStateChanged](#) (bool enabled)

6.15.1 Detailed Description

This interface defines an observer to observe some SDK states.

6.15.2 Member Function Documentation

6.15.2.1 onCameraChanged()

```
virtual void IFliSdkObserver::onCameraChanged ( ) [inline], [virtual]
```

6.15.2.2 onFowlerProcessingStateChanged()

```
virtual void IFliSdkObserver::onFowlerProcessingStateChanged (
    bool enabled ) [inline], [virtual]
```

6.15.2.3 onGrabNStateChanged()

```
virtual void IFliSdkObserver::onGrabNStateChanged (
    bool enabled,
    uint32_t nbFrames ) [inline], [virtual]
```

6.15.2.4 onResetBufferTriggered()

```
virtual void IFliSdkObserver::onResetBufferTriggered ( ) [inline], [virtual]
```

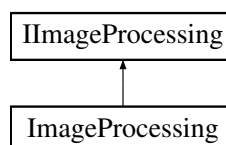
6.15.2.5 onStartedStateChanged()

```
virtual void IFliSdkObserver::onStartedStateChanged (
    bool started ) [inline], [virtual]
```

6.16 ImageProcessing Class Reference

```
#include <IImageProcessing.h>
```

Inheritance diagram for IImageProcessing:



Public Types

- enum [ClippingType](#) { [LINEAR_CLIPPING](#), [LOG_CLIPPING](#), [GAMMA_CLIPPING](#) }
- enum [ThermoUnit](#) { [CELSIUS](#), [KELVIN](#) }
- enum [BadPixelsAlgo](#) { [None](#), [Camera](#), [Soft](#) }

Public Member Functions

- virtual void [enableIndependentMode](#) (bool enable)=0
- virtual bool [isIndependent](#) ()=0
- virtual void [enable8bitsPixel](#) (bool enable)=0
- virtual unsigned char * [getProcessedImage](#) (const uint8_t *image)=0
- virtual const std::vector< std::string > [getColorMapList](#) () const =0
- virtual const std::vector< std::string > [getClippingTypeList](#) () const =0
- virtual void [setDimension](#) (unsigned int width, unsigned int height)=0
- virtual void [setClippingType](#) (std::string type)=0
- virtual void [setColorMapping](#) (std::string colorMap)=0
- virtual void [setClippingType](#) ([ClippingType](#) type)=0
- virtual [ClippingType](#) [getClippingType](#) ()=0
- virtual void [setGamma](#) (double gamma)=0
- virtual double [getGamma](#) ()=0
- virtual void [setRotationAngle](#) (unsigned int angle)=0
- virtual void [setRotationAngleText](#) (unsigned int angle)=0
- virtual void [enableDisplayInfos](#) (bool enable)=0
- virtual double [getMean16b](#) ()=0
- virtual double [getSpatialStdDev16b](#) ()=0
- virtual double [getMean8b](#) ()=0
- virtual double [getSpatialStdDev8b](#) ()=0
- virtual double [getClipBlack](#) ()=0
- virtual void [setClipBlack](#) (int32_t val)=0
- virtual double [getClipWhite](#) ()=0
- virtual void [setClipWhite](#) (int32_t val)=0
- virtual int16_t [getMinVal](#) ()=0
- virtual int32_t [getMaxVal](#) ()=0
- virtual double [getMean16bNoCompute](#) ()=0
- virtual double [getSpatialStdDev16bNoCompute](#) ()=0
- virtual const std::vector< uint64_t > & [getHistogram16bNoCompute](#) ()=0
- virtual const std::vector< uint64_t > & [getHistogram16bNegativeNoCompute](#) ()=0
- virtual void [forceCalcMeanStdDevAndHist16b](#) ()=0
- virtual const std::vector< uint64_t > & [getHistogram8b](#) ()=0
- virtual const std::vector< uint64_t > & [getHistogram16b](#) ()=0
- virtual const std::vector< uint64_t > & [getHistogram16bNegative](#) ()=0
- virtual void [clip](#) (int x, int y, int width, int height)=0
- virtual void [enableAutoClip](#) (bool enable)=0
- virtual bool [autoClipEnabled](#) ()=0
- virtual void [enableAutoExposure](#) (bool enable)=0
- virtual void [updateAutoExposureParam](#) ()=0
- virtual void [enableFilters](#) (bool enable)=0
- virtual double [getCoeffA](#) ()=0
- virtual double [getCoeffB](#) ()=0
- virtual void [setPixelSign](#) (bool unsignedPixel)=0
- virtual void [setPercentOfMean](#) (uint8_t percent)=0
- virtual uint8_t [getPercentOfMean](#) ()=0
- virtual void [setClipDepth](#) (uint8_t depth)=0

- virtual void [setClipLimit](#) (uint16_t limit)=0
- virtual void [setStdDevAndMeanSelection](#) (uint16_t x, uint16_t y, uint16_t width, uint16_t height)=0
- virtual void [getStdDevAndMeanSelection](#) (uint16_t &x, uint16_t &y, uint16_t &width, uint16_t &height)=0
- virtual void [enableDenoising](#) (bool enable)=0
- virtual void [setDenoisingH](#) (float val)=0
- virtual void [setDenoisingTemplateWindowSize](#) (int val)=0
- virtual void [setDenoisingSearchWindowSize](#) (int val)=0
- virtual void [enableSmoothImage](#) (bool enable)=0
- virtual void [enableManualClippingCoeff](#) (bool enable)=0
- virtual void [setClippingAlpha](#) (double alpha)=0
- virtual void [setClippingBeta](#) (double beta)=0
- virtual void [enableSharpen](#) (bool enable)=0
- virtual void [setSharpenKsize](#) (int width, int height)=0
- virtual void [setSharpenSigmaX](#) (double val)=0
- virtual void [setSharpenSigmaY](#) (double val)=0
- virtual void [setSharpenAlpha](#) (double val)=0
- virtual void [setSharpenBeta](#) (double val)=0
- virtual void [setSharpenGamma](#) (double val)=0
- virtual void [enableClahe](#) (bool enable)=0
- virtual void [setClaheCliplimit](#) (double limit)=0
- virtual void [setClaheTileGridSize](#) (int width, int height)=0
- virtual void [enableImagesAccumulation](#) (bool enable)=0
- virtual void [setnblImagesAccumulation](#) (uint8_t val)=0
- virtual void [flipVertically](#) ()=0
- virtual void [flipHorizontally](#) ()=0
- virtual bool [isFlippedHorizontally](#) ()=0
- virtual bool [isFlippedVertically](#) ()=0
- virtual void [setToneMappingNormal](#) ()=0
- virtual void [setToneMappingDrago](#) ()=0
- virtual void [setToneMappingReinhard](#) ()=0
- virtual void [setToneMappingMantiuk](#) ()=0
- virtual void [setDragoGamma](#) (float gamma)=0
- virtual void [setDragoSaturation](#) (float saturation)=0
- virtual void [setDragoBias](#) (float bias)=0
- virtual void [setDragoMultiplier](#) (uint8_t multiplier)=0
- virtual void [setReinhardGamma](#) (float gamma)=0
- virtual void [setReinhardIntensity](#) (float intensity)=0
- virtual void [setReinhardLightAdapt](#) (float light)=0
- virtual void [setReinhardColorAdapt](#) (float color)=0
- virtual void [setMantiukGamma](#) (float gamma)=0
- virtual void [setMantiukScale](#) (float scale)=0
- virtual void [setMantiukSaturation](#) (float saturation)=0
- virtual void [setMantiukMultiplier](#) (uint8_t multiplier)=0
- virtual void [setBadPixelsCarto](#) (std::vector< bool > carto)=0
- virtual void [enableBadPixelsCarto](#) ([BadPixelsAlgo](#) algo)=0
- virtual bool [badPixelsCartoLoaded](#) ()=0
- virtual uint16_t * [getRawThermoImage](#) (int64_t index=-1)=0
- virtual uint16_t * [getRawThermoImage](#) (const uint8_t *buffer)=0
- virtual void [enableThermo](#) (bool enable)=0
- virtual bool [isThermoEnabled](#) ()=0
- virtual void [setThermoCalibrationData](#) (Fli::ThermoCalibrationData &data)=0
- virtual Fli::ThermoCalibrationData & [getThermoCalibrationData](#) ()=0
- virtual void [setThermoUnit](#) ([ThermoUnit](#) unit)=0
- virtual [ThermoUnit](#) [getThermoUnit](#) ()=0
- virtual uint8_t * [getColorMapImage](#) (uint16_t width, uint16_t height)=0

- virtual uint8_t [getNumThreads](#) ()=0
- virtual uint8_t [getNumThreadsMax](#) ()=0
- virtual void [setNumThreads](#) (uint8_t num)=0
- virtual void [setIsThermoThrRaw](#) (bool isThermoThrRaw)=0
- virtual unsigned int [getSize](#) ()=0

Public Attributes

- std::mutex [processMutex](#)

6.16.1 Member Enumeration Documentation

6.16.1.1 BadPixelsAlgo

enum [IImageProcessing::BadPixelsAlgo](#)

Enumerator

None	
Camera	
Soft	

6.16.1.2 ClippingType

enum [IImageProcessing::ClippingType](#)

Enumerator

LINEAR_CLIPPING	
LOG_CLIPPING	
GAMMA_CLIPPING	

6.16.1.3 ThermoUnit

enum [IImageProcessing::ThermoUnit](#)

Enumerator

CELSIUS	
KELVIN	

6.16.2 Member Function Documentation

6.16.2.1 autoClipEnabled()

```
virtual bool IImageProcessing::autoClipEnabled ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.2 badPixelsCartoLoaded()

```
virtual bool IImageProcessing::badPixelsCartoLoaded ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.3 clip()

```
virtual void IImageProcessing::clip (
    int x,
    int y,
    int width,
    int height ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.4 enable8bitsPixel()

```
virtual void IImageProcessing::enable8bitsPixel (
    bool enable ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.5 enableAutoClip()

```
virtual void IImageProcessing::enableAutoClip (
    bool enable ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.6 enableAutoExposure()

```
virtual void IImageProcessing::enableAutoExposure (
    bool enable ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.7 enableBadPixelsCarto()

```
virtual void IImageProcessing::enableBadPixelsCarto (
    BadPixelsAlgo algo ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.8 enableClahe()

```
virtual void IImageProcessing::enableClahe (
    bool enable ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.9 enableDenoising()

```
virtual void IImageProcessing::enableDenoising (
    bool enable ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.10 enableDisplayInfos()

```
virtual void IImageProcessing::enableDisplayInfos (
    bool enable ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.11 enableFilters()

```
virtual void IImageProcessing::enableFilters (
    bool enable ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.12 enableImagesAccumulation()

```
virtual void IImageProcessing::enableImagesAccumulation (
    bool enable ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.13 enableIndependentMode()

```
virtual void IImageProcessing::enableIndependentMode (
    bool enable ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.14 enableManualClippingCoeff()

```
virtual void IImageProcessing::enableManualClippingCoeff (
    bool enable ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.15 enableSharpen()

```
virtual void IImageProcessing::enableSharpen (
    bool enable ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.16 enableSmoothImage()

```
virtual void IImageProcessing::enableSmoothImage (
    bool enable ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.17 enableThermo()

```
virtual void IImageProcessing::enableThermo (
    bool enable ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.18 flipHorizontally()

```
virtual void IImageProcessing::flipHorizontally ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.19 flipVertically()

```
virtual void IImageProcessing::flipVertically ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.20 forceCalcMeanStdDevAndHist16b()

```
virtual void IImageProcessing::forceCalcMeanStdDevAndHist16b ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.21 getClipBlack()

```
virtual double IImageProcessing::getClipBlack ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.22 getClippingType()

```
virtual ClippingType IImageProcessing::getClippingType ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.23 getClippingTypeList()

```
virtual const std::vector<std::string> IImageProcessing::getClippingTypeList ( ) const [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.24 getClipWhite()

```
virtual double IImageProcessing::getClipWhite ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.25 getCoeffA()

```
virtual double IImageProcessing::getCoeffA ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.26 getCoeffB()

```
virtual double IImageProcessing::getCoeffB ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.27 getColorMapImage()

```
virtual uint8_t* IImageProcessing::getColorMapImage (
    uint16_t width,
    uint16_t height ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.28 getColorMapList()

```
virtual const std::vector<std::string> IImageProcessing::getColorMapList ( ) const [pure
virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.29 getGamma()

```
virtual double IImageProcessing::getGamma ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.30 `getHistogram16b()`

```
virtual const std::vector<uint64_t>& IImageProcessing::getHistogram16b ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.31 `getHistogram16bNegative()`

```
virtual const std::vector<uint64_t>& IImageProcessing::getHistogram16bNegative ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.32 `getHistogram16bNegativeNoCompute()`

```
virtual const std::vector<uint64_t>& IImageProcessing::getHistogram16bNegativeNoCompute ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.33 `getHistogram16bNoCompute()`

```
virtual const std::vector<uint64_t>& IImageProcessing::getHistogram16bNoCompute ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.34 `getHistogram8b()`

```
virtual const std::vector<uint64_t>& IImageProcessing::getHistogram8b ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.35 `getMaxVal()`

```
virtual int32_t IImageProcessing::getMaxVal ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.36 getMean16b()

```
virtual double IImageProcessing::getMean16b ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.37 getMean16bNoCompute()

```
virtual double IImageProcessing::getMean16bNoCompute ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.38 getMean8b()

```
virtual double IImageProcessing::getMean8b ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.39 getMinVal()

```
virtual int16_t IImageProcessing::getMinVal ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.40 getNumThreads()

```
virtual uint8_t IImageProcessing::getNumThreads ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.41 getNumThreadsMax()

```
virtual uint8_t IImageProcessing::getNumThreadsMax ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.42 getPercentOfMean()

```
virtual uint8_t IImageProcessing::getPercentOfMean ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.43 getProcessedImage()

```
virtual unsigned char* IImageProcessing::getProcessedImage (
    const uint8_t * image ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.44 getRawThermoImage() [1/2]

```
virtual uint16_t* IImageProcessing::getRawThermoImage (
    const uint8_t * buffer ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.45 getRawThermoImage() [2/2]

```
virtual uint16_t* IImageProcessing::getRawThermoImage (
    int64_t index = -1 ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.46 getSize()

```
virtual unsigned int IImageProcessing::getSize ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.47 getSpatialStdDev16b()

```
virtual double IImageProcessing::getSpatialStdDev16b ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.48 getSpatialStdDev16bNoCompute()

```
virtual double IImageProcessing::getSpatialStdDev16bNoCompute ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.49 getSpatialStdDev8b()

```
virtual double IImageProcessing::getSpatialStdDev8b ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.50 getStdDevAndMeanSelection()

```
virtual void IImageProcessing::getStdDevAndMeanSelection (
    uint16_t & x,
    uint16_t & y,
    uint16_t & width,
    uint16_t & height ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.51 getThermoCalibrationData()

```
virtual Fli::ThermoCalibrationData& IImageProcessing::getThermoCalibrationData ( ) [pure
virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.52 getThermoUnit()

```
virtual ThermoUnit IImageProcessing::getThermoUnit ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.53 isFlippedHorizontally()

```
virtual bool IImageProcessing::isFlippedHorizontally ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.54 isFlippedVertically()

```
virtual bool IImageProcessing::isFlippedVertically ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.55 isIndependent()

```
virtual bool IImageProcessing::isIndependent ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.56 isThermoEnabled()

```
virtual bool IImageProcessing::isThermoEnabled ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.57 setBadPixelsCarto()

```
virtual void IImageProcessing::setBadPixelsCarto (
    std::vector< bool > carto ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.58 setClaheCliplimit()

```
virtual void IImageProcessing::setClaheCliplimit (
    double limit ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.59 setClaheTileGridSize()

```
virtual void IImageProcessing::setClaheTileGridSize (
    int width,
    int height ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.60 setClipBlack()

```
virtual void IImageProcessing::setClipBlack (
    int32_t val ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.61 setClipDepth()

```
virtual void IImageProcessing::setClipDepth (
    uint8_t depth ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.62 setClipLimit()

```
virtual void IImageProcessing::setClipLimit (
    uint16_t limit ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.63 setClippingAlpha()

```
virtual void IImageProcessing::setClippingAlpha (
    double alpha ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.64 setClippingBeta()

```
virtual void IImageProcessing::setClippingBeta (
    double beta ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.65 setClippingType() [1/2]

```
virtual void IImageProcessing::setClippingType (
    ClippingType type ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.66 setClippingType() [2/2]

```
virtual void IImageProcessing::setClippingType (
    std::string type ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.67 setClipWhite()

```
virtual void IImageProcessing::setClipWhite (
    int32_t val ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.68 setColorMapping()

```
virtual void IImageProcessing::setColorMapping (
    std::string colorMap ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.69 setDenoisingH()

```
virtual void IImageProcessing::setDenoisingH (
    float val ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.70 setDenoisingSearchWindowSize()

```
virtual void IImageProcessing::setDenoisingSearchWindowSize (
    int val ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.71 setDenoisingTemplateWindowSize()

```
virtual void IImageProcessing::setDenoisingTemplateWindowSize (
    int val ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.72 setDimension()

```
virtual void IImageProcessing::setDimension (
    unsigned int width,
    unsigned int height ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.73 setDragoBias()

```
virtual void IImageProcessing::setDragoBias (
    float bias ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.74 setDragoGamma()

```
virtual void IImageProcessing::setDragoGamma (
    float gamma ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.75 setDragoMultiplier()

```
virtual void IImageProcessing::setDragoMultiplier (
    uint8_t multiplier ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.76 setDragoSaturation()

```
virtual void IImageProcessing::setDragoSaturation (
    float saturation ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.77 setGamma()

```
virtual void IImageProcessing::setGamma (
    double gamma ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.78 setIsThermoThrRaw()

```
virtual void IImageProcessing::setIsThermoThrRaw (
    bool isThermoThrRaw ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.79 setMantiukGamma()

```
virtual void IImageProcessing::setMantiukGamma (
    float gamma ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.80 setMantiukMultiplier()

```
virtual void IImageProcessing::setMantiukMultiplier (
    uint8_t multiplier ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.81 setMantiukSaturation()

```
virtual void IImageProcessing::setMantiukSaturation (
    float saturation ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.82 setMantiukScale()

```
virtual void IImageProcessing::setMantiukScale (
    float scale ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.83 setnbImagesAccumulation()

```
virtual void IImageProcessing::setnbImagesAccumulation (
    uint8_t val ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.84 setNumThreads()

```
virtual void IImageProcessing::setNumThreads (
    uint8_t num ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.85 setPercentOfMean()

```
virtual void IImageProcessing::setPercentOfMean (
    uint8_t percent ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.86 setPixelSign()

```
virtual void IImageProcessing::setPixelSign (
    bool unsignedPixel ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.87 setReinhardColorAdapt()

```
virtual void IImageProcessing::setReinhardColorAdapt (
    float color ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.88 setReinhardGamma()

```
virtual void IImageProcessing::setReinhardGamma (
    float gamma ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.89 setReinhardIntensity()

```
virtual void IImageProcessing::setReinhardIntensity (
    float intensity ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.90 setReinhardLightAdapt()

```
virtual void IImageProcessing::setReinhardLightAdapt (
    float light ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.91 setRotationAngle()

```
virtual void IImageProcessing::setRotationAngle (
    unsigned int angle ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.92 setRotationAngleText()

```
virtual void IImageProcessing::setRotationAngleText (
    unsigned int angle ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.93 setSharpenAlpha()

```
virtual void IImageProcessing::setSharpenAlpha (
    double val ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.94 setSharpenBeta()

```
virtual void IImageProcessing::setSharpenBeta (
    double val ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.95 setSharpenGamma()

```
virtual void IImageProcessing::setSharpenGamma (
    double val ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.96 setSharpenKsize()

```
virtual void IImageProcessing::setSharpenKsize (
    int width,
    int height ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.97 setSharpenSigmaX()

```
virtual void IImageProcessing::setSharpenSigmaX (
    double val ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.98 setSharpenSigmaY()

```
virtual void IImageProcessing::setSharpenSigmaY (
    double val ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.99 setStdDevAndMeanSelection()

```
virtual void IImageProcessing::setStdDevAndMeanSelection (
    uint16_t x,
    uint16_t y,
    uint16_t width,
    uint16_t height ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.100 setThermoCalibrationData()

```
virtual void IImageProcessing::setThermoCalibrationData (
    Fli::ThermoCalibrationData & data ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.101 setThermoUnit()

```
virtual void IImageProcessing::setThermoUnit (
    ThermoUnit unit ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.102 setToneMappingDrago()

```
virtual void IImageProcessing::setToneMappingDrago ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.103 setToneMappingMantiuk()

```
virtual void IImageProcessing::setToneMappingMantiuk ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.104 setToneMappingNormal()

```
virtual void IImageProcessing::setToneMappingNormal ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.105 setToneMappingReinhard()

```
virtual void IImageProcessing::setToneMappingReinhard ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.2.106 updateAutoExposureParam()

```
virtual void IImageProcessing::updateAutoExposureParam ( ) [pure virtual]
```

Implemented in [ImageProcessing](#).

6.16.3 Member Data Documentation

6.16.3.1 processMutex

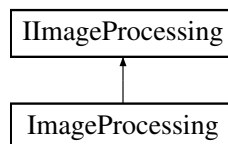
```
std::mutex IImageProcessing::processMutex
```

6.17 ImageProcessing Class Reference

This class manages all the processing of the images such as : statistics (mean stddev), clipping, flipping, sharpen etc...

```
#include <ImageProcessing.h>
```

Inheritance diagram for ImageProcessing:



Public Member Functions

- [ImageProcessing](#) ([ImageRingBuffer](#) *ringBuffer, [FliCred](#) *camera, unsigned int width, unsigned int height)
- [ImageProcessing](#) ([ImageRingBuffer](#) *ringBuffer, [FliSfncCamera](#) *camera, unsigned int width, unsigned int height)
- [ImageProcessing](#) ([ImageRingBuffer](#) *ringBuffer)
- virtual [~ImageProcessing](#) ()
- void [setCamera](#) ([FliCred](#) *camera)
- void [setCamera](#) ([FliSfncCamera](#) *camera)
- void [setRingBuffer](#) ([ImageRingBuffer](#) *ringBuffer)
- unsigned char * [getProcessedImage](#) (int64_t index=-1, int16_t burstFilter=-1, uint16_t nbReadWoReset=-1)
- unsigned char * [getProcessedImage16b](#) (int64_t index=-1, int16_t burstFilter=-1, uint16_t nbReadWoReset=-1)
- virtual void [enable8bitsPixel](#) (bool enable) override
- template<typename T >
void [aduToDegrees](#) (void *image, int precision=1)
- template<typename T >
void [aduToDegrees](#) (void *image, double *dest, int precision=1)
- virtual void [setPixelSign](#) (bool unsignedPixel) override

- virtual void [enableIndependentMode](#) (bool enable) override
- virtual bool [isIndependent](#) () override
- virtual unsigned char * [getProcessedImage](#) (const uint8_t *image) override
- virtual const std::vector< std::string > [getColorMapList](#) () const override
- virtual const std::vector< std::string > [getClippingTypeList](#) () const override
- virtual void [setDimension](#) (unsigned int width, unsigned int height) override
- virtual void [setClippingType](#) (std::string type) override
- virtual void [setColorMapping](#) (std::string colorMap) override
- virtual void [setClippingType](#) ([ClippingType](#) type) override
- virtual [ClippingType](#) [getClippingType](#) () override
- virtual void [setGamma](#) (double gamma) override
- virtual double [getGamma](#) () override
- virtual void [setRotationAngle](#) (unsigned int angle) override
- virtual void [setRotationAngleText](#) (unsigned int angle) override
- virtual void [enableDisplayInfos](#) (bool enable) override
- virtual double [getMean16b](#) () override
- virtual double [getSpatialStdDev16b](#) () override
- virtual double [getMean8b](#) () override
- virtual double [getSpatialStdDev8b](#) () override
- virtual double [getClipBlack](#) () override
- virtual void [setClipBlack](#) (int32_t val) override
- virtual double [getClipWhite](#) () override
- virtual void [setClipWhite](#) (int32_t val) override
- virtual int16_t [getMinVal](#) () override
- virtual int32_t [getMaxVal](#) () override
- virtual double [getMean16bNoCompute](#) () override
- virtual double [getSpatialStdDev16bNoCompute](#) () override
- virtual const std::vector< uint64_t > & [getHistogram16bNoCompute](#) () override
- virtual const std::vector< uint64_t > & [getHistogram16bNegativeNoCompute](#) () override
- virtual void [forceCalcMeanStdDevAndHist16b](#) () override
- virtual const std::vector< uint64_t > & [getHistogram8b](#) () override
- virtual const std::vector< uint64_t > & [getHistogram16b](#) () override
- virtual const std::vector< uint64_t > & [getHistogram16bNegative](#) () override
- virtual void [clip](#) (int x, int y, int width, int height) override
- virtual void [enableAutoClip](#) (bool enable) override
- virtual bool [autoClipEnabled](#) () override
- virtual void [enableAutoExposure](#) (bool enable) override
- virtual void [updateAutoExposureParam](#) () override
- virtual void [enableFilters](#) (bool enable) override
- virtual double [getCoeffA](#) () override
- virtual double [getCoeffB](#) () override
- virtual void [setPercentOfMean](#) (uint8_t percent) override
- virtual uint8_t [getPercentOfMean](#) () override
- virtual void [setClipDepth](#) (uint8_t depth) override
- virtual void [setClipLimit](#) (uint16_t limit) override
- virtual void [setStdDevAndMeanSelection](#) (uint16_t x, uint16_t y, uint16_t width, uint16_t height) override
- virtual void [getStdDevAndMeanSelection](#) (uint16_t &x, uint16_t &y, uint16_t &width, uint16_t &height) override
- virtual void [enableDenoising](#) (bool enable) override
- virtual void [setDenoisingH](#) (float val) override
- virtual void [setDenoisingTemplateWindowSize](#) (int val) override
- virtual void [setDenoisingSearchWindowSize](#) (int val) override
- virtual void [enableSmoothImage](#) (bool enable) override
- virtual void [enableManualClippingCoeff](#) (bool enable) override
- virtual void [setClippingAlpha](#) (double alpha) override

- virtual void [setClippingBeta](#) (double beta) override
- virtual void [enableSharpen](#) (bool enable) override
- virtual void [setSharpenKsize](#) (int width, int height) override
- virtual void [setSharpenSigmaX](#) (double val) override
- virtual void [setSharpenSigmaY](#) (double val) override
- virtual void [setSharpenAlpha](#) (double val) override
- virtual void [setSharpenBeta](#) (double val) override
- virtual void [setSharpenGamma](#) (double val) override
- virtual void [enableClahe](#) (bool enable) override
- virtual void [setClaheCliplimit](#) (double limit) override
- virtual void [setClaheTileGridSize](#) (int width, int height) override
- virtual void [enableImagesAccumulation](#) (bool enable) override
- virtual void [setnblImagesAccumulation](#) (uint8_t val) override
- virtual void [flipVertically](#) () override
- virtual void [flipHorizontally](#) () override
- virtual bool [isFlippedHorizontally](#) () override
- virtual bool [isFlippedVertically](#) () override
- virtual void [setToneMappingNormal](#) () override
- virtual void [setToneMappingDrago](#) () override
- virtual void [setToneMappingReinhard](#) () override
- virtual void [setToneMappingMantiuk](#) () override
- virtual void [setDragoGamma](#) (float gamma) override
- virtual void [setDragoSaturation](#) (float saturation) override
- virtual void [setDragoBias](#) (float bias) override
- virtual void [setDragoMultiplier](#) (uint8_t multiplier) override
- virtual void [setReinhardGamma](#) (float gamma) override
- virtual void [setReinhardIntensity](#) (float intensity) override
- virtual void [setReinhardLightAdapt](#) (float light) override
- virtual void [setReinhardColorAdapt](#) (float color) override
- virtual void [setMantiukGamma](#) (float gamma) override
- virtual void [setMantiukScale](#) (float scale) override
- virtual void [setMantiukSaturation](#) (float saturation) override
- virtual void [setMantiukMultiplier](#) (uint8_t multiplier) override
- virtual uint16_t * [getRawThermoImage](#) (int64_t index=-1) override
- virtual uint16_t * [getRawThermoImage](#) (const uint8_t *buffer) override
- virtual void [enableThermo](#) (bool enable) override
- virtual bool [isThermoEnabled](#) () override
- virtual void [setThermoCalibrationData](#) (Fli::ThermoCalibrationData &data) override
- virtual Fli::ThermoCalibrationData & [getThermoCalibrationData](#) () override
- virtual void [setThermoUnit](#) (ThermoUnit unit) override
- virtual ThermoUnit [getThermoUnit](#) () override
- virtual uint8_t * [getColorMapImage](#) (uint16_t width, uint16_t height) override
- virtual uint8_t [getNumThreads](#) () override
- virtual uint8_t [getNumThreadsMax](#) () override
- virtual void [setNumThreads](#) (uint8_t num) override
- virtual void [setBadPixelsCarto](#) (std::vector< bool > carto) override
- virtual void [enableBadPixelsCarto](#) (ImageProcessing::BadPixelsAlgo algo) override
- virtual bool [badPixelsCartoLoaded](#) () override
- virtual void [setIsThermoThrRaw](#) (bool isThermoThrRaw) override
- virtual unsigned int [getSize](#) () override
- [ImageRingBuffer](#) * [getRingBuffer](#) () const

Additional Inherited Members

6.17.1 Detailed Description

This class manages all the processing of the images such as : statistics (mean stddev), clipping, flipping, sharpen etc...

6.17.2 Constructor & Destructor Documentation

6.17.2.1 ImageProcessing() [1/3]

```
ImageProcessing::ImageProcessing (
    ImageRingBuffer * ringBuffer,
    FliCred * camera,
    unsigned int width,
    unsigned int height )
```

6.17.2.2 ImageProcessing() [2/3]

```
ImageProcessing::ImageProcessing (
    ImageRingBuffer * ringBuffer,
    FliSfncCamera * camera,
    unsigned int width,
    unsigned int height )
```

6.17.2.3 ImageProcessing() [3/3]

```
ImageProcessing::ImageProcessing (
    ImageRingBuffer * ringBuffer ) [explicit]
```

6.17.2.4 ~ImageProcessing()

```
virtual ImageProcessing::~ImageProcessing ( ) [virtual]
```

6.17.3 Member Function Documentation

6.17.3.1 `aduToDegrees()` [1/2]

```
template<typename T >
void ImageProcessing::aduToDegrees (
    void * image,
    double * dest,
    int precision = 1 )
```

6.17.3.2 `aduToDegrees()` [2/2]

```
template<typename T >
void ImageProcessing::aduToDegrees (
    void * image,
    int precision = 1 )
```

6.17.3.3 `autoClipEnabled()`

```
virtual bool ImageProcessing::autoClipEnabled ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.4 `badPixelsCartoLoaded()`

```
virtual bool ImageProcessing::badPixelsCartoLoaded ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.5 `clip()`

```
virtual void ImageProcessing::clip (
    int x,
    int y,
    int width,
    int height ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.6 enable8bitsPixel()

```
virtual void ImageProcessing::enable8bitsPixel (
    bool enable ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.7 enableAutoClip()

```
virtual void ImageProcessing::enableAutoClip (
    bool enable ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.8 enableAutoExposure()

```
virtual void ImageProcessing::enableAutoExposure (
    bool enable ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.9 enableBadPixelsCarto()

```
virtual void ImageProcessing::enableBadPixelsCarto (
    IImageProcessing::BadPixelsAlgo algo ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.10 enableClahe()

```
virtual void ImageProcessing::enableClahe (
    bool enable ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.11 enableDenoising()

```
virtual void ImageProcessing::enableDenoising (
    bool enable ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.12 enableDisplayInfos()

```
virtual void ImageProcessing::enableDisplayInfos (
    bool enable ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.13 enableFilters()

```
virtual void ImageProcessing::enableFilters (
    bool enable ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.14 enableImagesAccumulation()

```
virtual void ImageProcessing::enableImagesAccumulation (
    bool enable ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.15 enableIndependentMode()

```
virtual void ImageProcessing::enableIndependentMode (
    bool enable ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.16 enableManualClippingCoeff()

```
virtual void ImageProcessing::enableManualClippingCoeff (
    bool enable ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.17 enableSharpen()

```
virtual void ImageProcessing::enableSharpen (
    bool enable ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.18 enableSmoothImage()

```
virtual void ImageProcessing::enableSmoothImage (
    bool enable ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.19 enableThermo()

```
virtual void ImageProcessing::enableThermo (
    bool enable ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.20 flipHorizontally()

```
virtual void ImageProcessing::flipHorizontally ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.21 flipVertically()

```
virtual void ImageProcessing::flipVertically ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.22 forceCalcMeanStdDevAndHist16b()

```
virtual void ImageProcessing::forceCalcMeanStdDevAndHist16b ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.23 getClipBlack()

```
virtual double ImageProcessing::getClipBlack ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.24 getClippingType()

```
virtual ClippingType ImageProcessing::getClippingType ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.25 getClippingTypeList()

```
virtual const std::vector<std::string> ImageProcessing::getClippingTypeList ( ) const [override],  
[virtual]
```

Implements [IImageProcessing](#).

6.17.3.26 getClipWhite()

```
virtual double ImageProcessing::getClipWhite ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.27 getCoeffA()

```
virtual double ImageProcessing::getCoeffA ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.28 getCoeffB()

```
virtual double ImageProcessing::getCoeffB ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.29 getColorMapImage()

```
virtual uint8_t* ImageProcessing::getColorMapImage (   
    uint16_t width,  
    uint16_t height ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.30 getColorMapList()

```
virtual const std::vector<std::string> ImageProcessing::getColorMapList ( ) const [override],  
[virtual]
```

Implements [IImageProcessing](#).

6.17.3.31 getGamma()

```
virtual double ImageProcessing::getGamma ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.32 getHistogram16b()

```
virtual const std::vector<uint64_t>& ImageProcessing::getHistogram16b ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.33 getHistogram16bNegative()

```
virtual const std::vector<uint64_t>& ImageProcessing::getHistogram16bNegative ( ) [override],  
[virtual]
```

Implements [IImageProcessing](#).

6.17.3.34 getHistogram16bNegativeNoCompute()

```
virtual const std::vector<uint64_t>& ImageProcessing::getHistogram16bNegativeNoCompute ( )  
[override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.35 getHistogram16bNoCompute()

```
virtual const std::vector<uint64_t>& ImageProcessing::getHistogram16bNoCompute ( ) [override],  
[virtual]
```

Implements [IImageProcessing](#).

6.17.3.36 getHistogram8b()

```
virtual const std::vector<uint64_t>& ImageProcessing::getHistogram8b ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.37 getMaxVal()

```
virtual int32_t ImageProcessing::getMaxVal ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.38 getMean16b()

```
virtual double ImageProcessing::getMean16b ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.39 getMean16bNoCompute()

```
virtual double ImageProcessing::getMean16bNoCompute ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.40 getMean8b()

```
virtual double ImageProcessing::getMean8b ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.41 getMinVal()

```
virtual int16_t ImageProcessing::getMinVal ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.42 `getNumThreads()`

```
virtual uint8_t ImageProcessing::getNumThreads ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.43 `getNumThreadsMax()`

```
virtual uint8_t ImageProcessing::getNumThreadsMax ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.44 `getPercentOfMean()`

```
virtual uint8_t ImageProcessing::getPercentOfMean ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.45 `getProcessedImage()` [1/2]

```
virtual unsigned char* ImageProcessing::getProcessedImage (
    const uint8_t * image ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.46 `getProcessedImage()` [2/2]

```
unsigned char* ImageProcessing::getProcessedImage (
    int64_t index = -1,
    int16_t burstFilter = -1,
    uint16_t nbReadWoReset = -1 )
```

6.17.3.47 `getProcessedImage16b()`

```
unsigned char* ImageProcessing::getProcessedImage16b (
    int64_t index = -1,
    int16_t burstFilter = -1,
    uint16_t nbReadWoReset = -1 )
```

6.17.3.48 getRawThermoImage() [1/2]

```
virtual uint16_t* ImageProcessing::getRawThermoImage (
    const uint8_t * buffer ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.49 getRawThermoImage() [2/2]

```
virtual uint16_t* ImageProcessing::getRawThermoImage (
    int64_t index = -1 ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.50 getRingBuffer()

```
ImageRingBuffer* ImageProcessing::getRingBuffer ( ) const
```

6.17.3.51 getSize()

```
virtual unsigned int ImageProcessing::getSize ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.52 getSpatialStdDev16b()

```
virtual double ImageProcessing::getSpatialStdDev16b ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.53 getSpatialStdDev16bNoCompute()

```
virtual double ImageProcessing::getSpatialStdDev16bNoCompute ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.54 `getSpatialStdDev8b()`

```
virtual double ImageProcessing::getSpatialStdDev8b ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.55 `getStdDevAndMeanSelection()`

```
virtual void ImageProcessing::getStdDevAndMeanSelection (
    uint16_t & x,
    uint16_t & y,
    uint16_t & width,
    uint16_t & height ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.56 `getThermoCalibrationData()`

```
virtual Fli::ThermoCalibrationData& ImageProcessing::getThermoCalibrationData ( ) [override],
[virtual]
```

Implements [IImageProcessing](#).

6.17.3.57 `getThermoUnit()`

```
virtual ThermoUnit ImageProcessing::getThermoUnit ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.58 `isFlippedHorizontally()`

```
virtual bool ImageProcessing::isFlippedHorizontally ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.59 `isFlippedVertically()`

```
virtual bool ImageProcessing::isFlippedVertically ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.60 isIndependent()

```
virtual bool ImageProcessing::isIndependent ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.61 isThermoEnabled()

```
virtual bool ImageProcessing::isThermoEnabled ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.62 setBadPixelsCarto()

```
virtual void ImageProcessing::setBadPixelsCarto (
    std::vector< bool > carto ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.63 setCamera() [1/2]

```
void ImageProcessing::setCamera (
    FliCred * camera )
```

6.17.3.64 setCamera() [2/2]

```
void ImageProcessing::setCamera (
    FliSfncCamera * camera )
```

6.17.3.65 setClaheCliplimit()

```
virtual void ImageProcessing::setClaheCliplimit (
    double limit ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.66 setClaheTileGridSize()

```
virtual void ImageProcessing::setClaheTileGridSize (
    int width,
    int height ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.67 setClipBlack()

```
virtual void ImageProcessing::setClipBlack (
    int32_t val ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.68 setClipDepth()

```
virtual void ImageProcessing::setClipDepth (
    uint8_t depth ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.69 setClipLimit()

```
virtual void ImageProcessing::setClipLimit (
    uint16_t limit ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.70 setClippingAlpha()

```
virtual void ImageProcessing::setClippingAlpha (
    double alpha ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.71 setClippingBeta()

```
virtual void ImageProcessing::setClippingBeta (
    double beta ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.72 setClippingType() [1/2]

```
virtual void ImageProcessing::setClippingType (
    ClippingType type ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.73 setClippingType() [2/2]

```
virtual void ImageProcessing::setClippingType (
    std::string type ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.74 setClipWhite()

```
virtual void ImageProcessing::setClipWhite (
    int32_t val ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.75 setColorMapping()

```
virtual void ImageProcessing::setColorMapping (
    std::string colorMap ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.76 setDenoisingH()

```
virtual void ImageProcessing::setDenoisingH (
    float val ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.77 setDenoisingSearchWindowSize()

```
virtual void ImageProcessing::setDenoisingSearchWindowSize (
    int val ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.78 setDenoisingTemplateWindowSize()

```
virtual void ImageProcessing::setDenoisingTemplateWindowSize (
    int val ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.79 setDimension()

```
virtual void ImageProcessing::setDimension (
    unsigned int width,
    unsigned int height ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.80 setDragoBias()

```
virtual void ImageProcessing::setDragoBias (
    float bias ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.81 setDragoGamma()

```
virtual void ImageProcessing::setDragoGamma (
    float gamma ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.82 setDragoMultiplier()

```
virtual void ImageProcessing::setDragoMultiplier (
    uint8_t multiplier ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.83 setDragoSaturation()

```
virtual void ImageProcessing::setDragoSaturation (
    float saturation ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.84 setGamma()

```
virtual void ImageProcessing::setGamma (
    double gamma ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.85 setIsThermoThrRaw()

```
virtual void ImageProcessing::setIsThermoThrRaw (
    bool isThermoThrRaw ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.86 setMantiukGamma()

```
virtual void ImageProcessing::setMantiukGamma (
    float gamma ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.87 setMantiukMultiplier()

```
virtual void ImageProcessing::setMantiukMultiplier (
    uint8_t multiplier ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.88 setMantiukSaturation()

```
virtual void ImageProcessing::setMantiukSaturation (
    float saturation ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.89 setMantiukScale()

```
virtual void ImageProcessing::setMantiukScale (
    float scale ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.90 setnbImagesAccumulation()

```
virtual void ImageProcessing::setnbImagesAccumulation (
    uint8_t val ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.91 setNumThreads()

```
virtual void ImageProcessing::setNumThreads (
    uint8_t num ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.92 setPercentOfMean()

```
virtual void ImageProcessing::setPercentOfMean (
    uint8_t percent ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.93 setPixelSign()

```
virtual void ImageProcessing::setPixelSign (
    bool unsignedPixel ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.94 setReinhardColorAdapt()

```
virtual void ImageProcessing::setReinhardColorAdapt (
    float color ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.95 setReinhardGamma()

```
virtual void ImageProcessing::setReinhardGamma (
    float gamma ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.96 setReinhardIntensity()

```
virtual void ImageProcessing::setReinhardIntensity (
    float intensity ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.97 setReinhardLightAdapt()

```
virtual void ImageProcessing::setReinhardLightAdapt (
    float light ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.98 setRingBuffer()

```
void ImageProcessing::setRingBuffer (
    ImageRingBuffer * ringBuffer )
```

6.17.3.99 setRotationAngle()

```
virtual void ImageProcessing::setRotationAngle (
    unsigned int angle ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.100 setRotationAngleText()

```
virtual void ImageProcessing::setRotationAngleText (
    unsigned int angle ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.101 setSharpenAlpha()

```
virtual void ImageProcessing::setSharpenAlpha (
    double val ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.102 setSharpenBeta()

```
virtual void ImageProcessing::setSharpenBeta (
    double val ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.103 setSharpenGamma()

```
virtual void ImageProcessing::setSharpenGamma (
    double val ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.104 setSharpenKsize()

```
virtual void ImageProcessing::setSharpenKsize (
    int width,
    int height ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.105 setSharpenSigmaX()

```
virtual void ImageProcessing::setSharpenSigmaX (
    double val ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.106 setSharpenSigmaY()

```
virtual void ImageProcessing::setSharpenSigmaY (
    double val ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.107 setStdDevAndMeanSelection()

```
virtual void ImageProcessing::setStdDevAndMeanSelection (
    uint16_t x,
    uint16_t y,
    uint16_t width,
    uint16_t height ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.108 setThermoCalibrationData()

```
virtual void ImageProcessing::setThermoCalibrationData (
    Fli::ThermoCalibrationData & data ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.109 setThermoUnit()

```
virtual void ImageProcessing::setThermoUnit (
    ThermoUnit unit ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.110 setToneMappingDrago()

```
virtual void ImageProcessing::setToneMappingDrago ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.111 setToneMappingMantiuk()

```
virtual void ImageProcessing::setToneMappingMantiuk ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.112 setToneMappingNormal()

```
virtual void ImageProcessing::setToneMappingNormal ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.113 setToneMappingReinhard()

```
virtual void ImageProcessing::setToneMappingReinhard ( ) [override], [virtual]
```

Implements [IImageProcessing](#).

6.17.3.114 updateAutoExposureParam()

```
virtual void ImageProcessing::updateAutoExposureParam ( ) [override], [virtual]
```

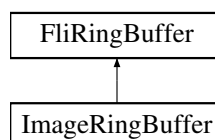
Implements [IImageProcessing](#).

6.18 ImageRingBuffer Class Reference

This class derive from pure virtual [FliRingBuffer](#) and manages the images inside a ring buffer with some put methods to add one or several images or nro or iota buffers into the ring buffer.

```
#include <ImageRingBuffer.h>
```

Inheritance diagram for ImageRingBuffer:



Public Member Functions

- [ImageRingBuffer](#) ()
- [~ImageRingBuffer](#) ()
- virtual uint32_t [getFilling](#) () const override
Get buffer filling.
- virtual uint16_t [getSizeInMo](#) () override
Get current buffer size.
- virtual uint32_t [getSizeInFrames](#) () override
Give the images capacity of the buffer.
- virtual uint32_t [getNumberOfWrap](#) () override
Get the number of times that the buffer had been full since reset.
- virtual uint64_t [getNbCountError](#) () override
Get the number of frame count error.
- virtual int64_t [getLastImageIndex](#) () const override
Get the last image acquired index.
- double [getFps](#) () const
- void [getImageDimension](#) (uint16_t &width, uint16_t &height)
- unsigned int [getOcamFrameNumber](#) (int64_t index=-1)
- const uint8_t * [getImage](#) (int64_t index=-1)
- virtual uint16_t [nbFramesInAccumulation](#) () override
- virtual void [setSizeInMo](#) (uint16_t sizeMo) override
Change the buffer capacity in Mo.
- virtual void [setSizeInFrames](#) (uint32_t nbFrames) override
Change the buffer capacity in number of images.
- virtual void [setSizeInFramesThermo](#) (uint32_t nbFrames) override
Change the buffer capacity in number of images for a thermographic analysis (.thr.raw files)*
- virtual void [setFowlerOffset](#) (uint16_t offset) override
- void [setImageDimension](#) (uint16_t width, uint16_t height)
setImageDimension define the image size and capacity from the width and height
- void [setImageDimensionThermo](#) (uint16_t width, uint16_t height)
setImageDimension define the thermographic image size and capacity from the width and height
- void [setImageTagState](#) (bool enabled)
- void [resetCountError](#) ()
- void [resetGrabN](#) ()
- void [setNbReadImro](#) (uint32_t nbRead)
- void [resetNbSecondsFps](#) ()
- void [setCameraModel](#) (Fli::CameraModel model)
- void [setOcamFrameNumberOffset](#) (uint8_t offset)
- void [setDefaultCapacity](#) ()
- void [setObserverList](#) (std::list< IFliSdkObserver * > *obs)
- virtual void [reset](#) () override
Reset the buffer.
- virtual void [resetAccumulation](#) () override
- virtual bool [isEnabled](#) () override
Return true if the buffer is enabled else false.
- virtual void [enableGrabN](#) (uint32_t nbFrames) override
Enable grab N mode, the buffer will stop copy images when nbFrames images is hit.
- virtual void [disableGrabN](#) () override
Disable grab N mode.
- virtual bool [isGrabNFinished](#) () const override
State of the grab N.

- virtual bool `isGrabNEnabled ()` const override
State of the grab N mode.
- virtual void `enableSubstractMode (bool enable)` override
Enable/disable the mode substract that will substract the image N by the image N-1 and save it in the buffer.
- virtual void `enableAccumulationMode (bool enable)` override
- virtual void `enable (bool enable)` override
Enable or disable internal ring buffer of the SDK.
- void `enableModelmro (bool enable)`
- void `enableFollowUpTheRamp (bool enable)`
- void `enableObserversNotif (bool enable)`
- void `enable8BitsPixel (bool enable)`
- void `enable8BitsPixelThermo ()`
- void `put (const uint8_t *image, bool bypassGrabN=false)`
put an image into the main buffer
- void `put (const uint8_t *data, uint64_t size)`
put one or several images to the main buffer
- void `putFollowUpTheRamp (const uint16_t *image)`
putFollowUpTheRamp add an image to the pixels sum of the follow up
- void `setNbRead (uint16_t nbRead)`
- void `setNbLoop (uint16_t nbLoop)`
- void `setNbSampPix (uint16_t nbSamp)`
- void `putNro (const uint16_t *images)`
putNro (only for C-RED 1) put an image from a nro buffer of images into the main buffer
- void `putlota (const uint16_t *images)`
putlota (only for C-RED 1) put an image from a iota buffer of images into the main buffer
- void `putFowler (const uint16_t *image)`
putFowler (only for C-RED 1) add if image number < number of image to read or else substract an image in a fowler image

6.18.1 Detailed Description

This class derive from pure virtual [FliRingBuffer](#) and manages the images inside a ring buffer with some put methods to add one or several images or nro or iota buffers into the ring buffer.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 ImageRingBuffer()

```
ImageRingBuffer::ImageRingBuffer ( ) [explicit]
```

6.18.2.2 ~ImageRingBuffer()

```
ImageRingBuffer::~ImageRingBuffer ( )
```

6.18.3 Member Function Documentation

6.18.3.1 disableGrabN()

```
virtual void ImageRingBuffer::disableGrabN ( ) [override], [virtual]
```

Disable grab N mode.

Implements [FliRingBuffer](#).

6.18.3.2 enable()

```
virtual void ImageRingBuffer::enable (
    bool enable ) [override], [virtual]
```

Enable or disable internal ring buffer of the SDK.

Attention

If ring buffer is disabled loadBuffer, saveBuffer, getRawImage, getImage and getImage16b will be disabled. grabN and error count will be disable too.

Implements [FliRingBuffer](#).

6.18.3.3 enable8BitsPixel()

```
void ImageRingBuffer::enable8BitsPixel (
    bool enable )
```

6.18.3.4 enable8BitsPixelThermo()

```
void ImageRingBuffer::enable8BitsPixelThermo ( )
```

6.18.3.5 enableAccumulationMode()

```
virtual void ImageRingBuffer::enableAccumulationMode (
    bool enable ) [override], [virtual]
```

Implements [FliRingBuffer](#).

6.18.3.6 enableFollowUpTheRamp()

```
void ImageRingBuffer::enableFollowUpTheRamp (
    bool enable )
```

6.18.3.7 enableGrabN()

```
virtual void ImageRingBuffer::enableGrabN (
    uint32_t nbFrames ) [override], [virtual]
```

Enable grab N mode, the buffer will stop copy images when nbFrames images is hit.

Parameters

<i>nbFrames</i>	: number of frames to grab.
-----------------	-----------------------------

Implements [FliRingBuffer](#).

6.18.3.8 enableModelmro()

```
void ImageRingBuffer::enableModeImro (
    bool enable )
```

6.18.3.9 enableObserversNotif()

```
void ImageRingBuffer::enableObserversNotif (
    bool enable )
```

6.18.3.10 enableSubstractMode()

```
virtual void ImageRingBuffer::enableSubstractMode (
    bool enable ) [override], [virtual]
```

Enable/disable the mode substract that will substract the image N by the image N-1 and save it in the buffer.

Parameters

<i>enable</i>	enable/disable the mode
---------------	-------------------------

Implements [FliRingBuffer](#).

6.18.3.11 getFilling()

```
virtual uint32_t ImageRingBuffer::getFilling ( ) const [override], [virtual]
```

Get buffer filling.

Returns

a number representing the filling

Implements [FliRingBuffer](#).

6.18.3.12 getFps()

```
double ImageRingBuffer::getFps ( ) const
```

6.18.3.13 getImage()

```
const uint8_t* ImageRingBuffer::getImage (
    int64_t index = -1 )
```

6.18.3.14 getImageDimension()

```
void ImageRingBuffer::getImageDimension (
    uint16_t & width,
    uint16_t & height )
```

6.18.3.15 getLastImageIndex()

```
virtual int64_t ImageRingBuffer::getLastImageIndex ( ) const [override], [virtual]
```

Get the last image acquired index.

Returns

the index or -1 if no image in buffer

Implements [FliRingBuffer](#).

6.18.3.16 getNbCountError()

```
virtual uint64_t ImageRingBuffer::getNbCountError ( ) [override], [virtual]
```

Get the number of frame count error.

Returns

the number of count error

Implements [FliRingBuffer](#).

6.18.3.17 getNumberOfWrap()

```
virtual uint32_t ImageRingBuffer::getNumberOfWrap ( ) [override], [virtual]
```

Get the number of times that the buffer had been full since reset.

Returns

the number of wrap

Implements [FliRingBuffer](#).

6.18.3.18 getOcamFrameNumber()

```
unsigned int ImageRingBuffer::getOcamFrameNumber (
    int64_t index = -1 )
```

6.18.3.19 getSizeInFrames()

```
virtual uint32_t ImageRingBuffer::getSizeInFrames ( ) [override], [virtual]
```

Give the images capacity of the buffer.

Returns

FliSdkError

Implements [FliRingBuffer](#).

6.18.3.20 getSizeInMo()

```
virtual uint16_t ImageRingBuffer::getSizeInMo ( ) [override], [virtual]
```

Get current buffer size.

Returns

the buffer size in Mo

Implements [FliRingBuffer](#).

6.18.3.21 isEnabled()

```
virtual bool ImageRingBuffer::isEnabled ( ) [override], [virtual]
```

Return true if the buffer is enabled else false.

Implements [FliRingBuffer](#).

6.18.3.22 isGrabNEnabled()

```
virtual bool ImageRingBuffer::isGrabNEnabled ( ) const [override], [virtual]
```

State of the grab N mode.

Returns

true if grab N mode activated else false

Implements [FliRingBuffer](#).

6.18.3.23 isGrabNFinished()

```
virtual bool ImageRingBuffer::isGrabNFinished ( ) const [override], [virtual]
```

State of the grab N.

Returns

true if the grab is over else false

Implements [FliRingBuffer](#).

6.18.3.24 nbFramesInAccumulation()

```
virtual uint16_t ImageRingBuffer::nbFramesInAccumulation ( ) [override], [virtual]
```

Implements [FliRingBuffer](#).

6.18.3.25 put() [1/2]

```
void ImageRingBuffer::put (
    const uint8_t * data,
    uint64_t size )
```

put one or several images to the main buffer

Parameters

<i>data</i>	a pointer to a buffer containing one or several image
<i>size</i>	the size of the buffer : number of images x one image size

6.18.3.26 put() [2/2]

```
void ImageRingBuffer::put (
    const uint8_t * image,
    bool bypassGrabN = false )
```

put an image into the main buffer

Parameters

<i>image</i>	the image to put
<i>bypassGrabN</i>	true will bypass the grabN mechanism

6.18.3.27 putFollowUpTheRamp()

```
void ImageRingBuffer::putFollowUpTheRamp (
    const uint16_t * image )
```

putFollowUpTheRamp add an image to the pixels sum of the follow up

Parameters

<i>image</i>	the image to add
--------------	------------------

6.18.3.28 putFowler()

```
void ImageRingBuffer::putFowler (
    const uint16_t * image )
```

putFowler (only for C-RED 1) add if image number < number of image to read or else subtract an image in a fowler image

Parameters

<i>image</i>	the image to add or subtract
--------------	------------------------------

6.18.3.29 putIota()

```
void ImageRingBuffer::putIota (
    const uint16_t * images )
```

putIota (only for C-RED 1) put an image from a iota buffer of images into the main buffer

Parameters

<i>images</i>	a iota buffer of images
---------------	-------------------------

6.18.3.30 putNro()

```
void ImageRingBuffer::putNro (
    const uint16_t * images )
```

putNro (only for C-RED 1) put an image from a nro buffer of images into the main buffer

Parameters

<i>images</i>	a nro buffer of images
---------------	------------------------

6.18.3.31 reset()

```
virtual void ImageRingBuffer::reset ( ) [override], [virtual]
```

Reset the buffer.

Implements [FliRingBuffer](#).

6.18.3.32 resetAccumulation()

```
virtual void ImageRingBuffer::resetAccumulation ( ) [override], [virtual]
```

Implements [FliRingBuffer](#).

6.18.3.33 resetCountError()

```
void ImageRingBuffer::resetCountError ( )
```

6.18.3.34 resetGrabN()

```
void ImageRingBuffer::resetGrabN ( )
```

6.18.3.35 resetNbSecondsFps()

```
void ImageRingBuffer::resetNbSecondsFps ( )
```

6.18.3.36 setCameraModel()

```
void ImageRingBuffer::setCameraModel (
    Fli::CameraModel model )
```

6.18.3.37 setDefaultCapacity()

```
void ImageRingBuffer::setDefaultCapacity ( )
```

6.18.3.38 setFowlerOffset()

```
virtual void ImageRingBuffer::setFowlerOffset (
    uint16_t offset ) [override], [virtual]
```

Implements [FliRingBuffer](#).

6.18.3.39 setImageDimension()

```
void ImageRingBuffer::setImageDimension (
    uint16_t width,
    uint16_t height )
```

setImageDimension define the image size and capacity from the width and height

Parameters

<i>width</i>	of the image in pixel
<i>height</i>	of the image in pixel

6.18.3.40 setImageDimensionThermo()

```
void ImageRingBuffer::setImageDimensionThermo (
    uint16_t width,
    uint16_t height )
```

setImageDimension define the thermographic image size and capacity from the width and height

Parameters

<i>width</i>	of the image in pixel
<i>height</i>	of the image in pixel

6.18.3.41 setImageTagState()

```
void ImageRingBuffer::setImageTagState (
    bool enabled )
```

6.18.3.42 setNbLoop()

```
void ImageRingBuffer::setNbLoop (
    uint16_t nbLoop )
```

6.18.3.43 setNbRead()

```
void ImageRingBuffer::setNbRead (
    uint16_t nbRead )
```

6.18.3.44 setNbReadImro()

```
void ImageRingBuffer::setNbReadImro (
    uint32_t nbRead )
```

6.18.3.45 setNbSampPix()

```
void ImageRingBuffer::setNbSampPix (
    uint16_t nbSamp )
```

6.18.3.46 setObserverList()

```
void ImageRingBuffer::setObserverList (
    std::list< IFliSdkObserver * > * obs )
```

6.18.3.47 setOcamFrameNumberOffset()

```
void ImageRingBuffer::setOcamFrameNumberOffset (
    uint8_t offset )
```

6.18.3.48 setSizeInFrames()

```
virtual void ImageRingBuffer::setSizeInFrames (
    uint32_t nbFrames ) [override], [virtual]
```

Change the buffer capacity in number of images.

Parameters

<i>nbFrames</i>	: capacity of the ring buffer in nb images
-----------------	--

Implements [FliRingBuffer](#).

6.18.3.49 setSizeInFramesThermo()

```
virtual void ImageRingBuffer::setSizeInFramesThermo (
    uint32_t nbFrames ) [override], [virtual]
```

Change the buffer capacity in number of images for a thermographic analysis (*.thr.raw files)

Parameters

<i>nbFrames</i>	: capacity of the ring buffer in nb images
-----------------	--

Implements [FliRingBuffer](#).

6.18.3.50 setSizeInMo()

```
virtual void ImageRingBuffer::setSizeInMo (
    uint16_t sizeMo ) [override], [virtual]
```

Change the buffer capacity in Mo.

Parameters

<i>sizeMo</i>	: capacity of the ring buffer in Mo
---------------	-------------------------------------

Implements [FliRingBuffer](#).

6.19 IRawImageReceivedObserver Class Reference

This can be herited to be an observer of the reception of a raw image.

```
#include <FliSdk.h>
```

Public Member Functions

- virtual void [imageReceived](#) (const uint8_t *image)

This function is called when a new image is received (this function is deprecated, please use [imageReceivedBeforeBuffer](#) or [imageReceivedAfterBuffer](#)) Use [addRawImageReceivedObserver](#) with `beforeCopy` to true to be notified before the copy in the ringBuffer, so the image come directly from the grabber Use [addRawImageReceivedObserver](#) with `beforeCopy` to false to be notified after the copy in the ringBuffer, so the image comes from the ringBuffer.
- virtual void [imageReceivedBeforeBuffer](#) (const uint8_t *image, uint16_t nblImages, bool ©InBuffer)

This function is called when a new image is received and the observer is register before the ring buffer Use [addRawImageReceivedObserver](#) with `beforeCopy` to true to be notified before the copy in the ringBuffer, so the image come directly from the grabber The function "useDeprecatedFunction" must return false to use this function.
- virtual void [imageReceivedAfterBuffer](#) (const uint8_t *image, uint16_t nblImages)

This function is called when a new image is received and the observer is register after the ring buffer Use [addRawImageReceivedObserver](#) with `beforeCopy` to false to be notified after the copy in the ringBuffer, so the image comes from the ringBuffer The function "useDeprecatedFunction" must return false to use this function.
- virtual uint16_t [fpsTrigger](#) ()

This function is called to know at which fps does the observer want to receive images.
- virtual bool [useDeprecatedFunction](#) ()

If return true then "imageReceived" is called, else "imageReceivedBeforeBuffer" or "imageReceivedAfterBuffer" is called.

6.19.1 Detailed Description

This can be herited to be an observer of the reception of a raw image.

6.19.2 Member Function Documentation

6.19.2.1 fpsTrigger()

```
virtual uint16_t IRawImageReceivedObserver::fpsTrigger ( ) [inline], [virtual]
```

This function is called to know at which fps does the observer want to receive images.

Returns

fps value, if 0 then observer is notified at full speed.

6.19.2.2 imageReceived()

```
virtual void IRawImageReceivedObserver::imageReceived (
    const uint8_t * image ) [inline], [virtual]
```

This function is called when a new image is received (this function is deprecated, please use `imageReceivedBeforeBuffer` or `imageReceivedAfterBuffer`) Use `addRawImageReceivedObserver` with `beforeCopy` to true to be notified before the copy in the ringBuffer, so the image come directly from the grabber Use `addRawImageReceivedObserver` with `beforeCopy` to false to be notified after the copy in the ringBuffer, so the image comes from the ringBuffer.

Parameters

<i>image</i>	pointer to the buffer of image
--------------	--------------------------------

Attention

This function is called by a thread, you need to resynchronize it and not call a sdk function because it's not thread safe.

6.19.2.3 imageReceivedAfterBuffer()

```
virtual void IRawImageReceivedObserver::imageReceivedAfterBuffer (
    const uint8_t * image,
    uint16_t nbImages ) [inline], [virtual]
```

This function is called when a new image is received and the observer is register after the ring buffer Use `addRawImageReceivedObserver` with `beforeCopy` to false to be notified after the copy in the ringBuffer, so the image comes from the ringBuffer The function "useDeprecatedFunction" must return false to use this function.

Parameters

<i>image</i>	pointer to the buffer of image
<i>nbImages</i>	number of images in the buffer

Attention

This function is called by a thread, you need to resynchronize it and not call a sdk function because it's not thread safe.

6.19.2.4 imageReceivedBeforeBuffer()

```
virtual void IRawImageReceivedObserver::imageReceivedBeforeBuffer (
    const uint8_t * image,
    uint16_t nbImages,
    bool & copyInBuffer ) [inline], [virtual]
```

This function is called when a new image is received and the observer is register before the ring buffer Use add↔RawImageReceivedObserver with beforeCopy to true to be notified before the copy in the ringBuffer, so the image come directly from the grabber The function "useDeprecatedFunction" must return false to use this function.

Parameters

<i>image</i>	pointer to the buffer of image(s)
<i>nbImages</i>	number of images in the buffer
<i>copyInBuffer</i>	if true the images are copied in the ring buffer of the SDK (if the ring buffer is enabled)

Attention

This function is called by a thread, you need to resynchronize it and not call a sdk function because it's not thread safe.

6.19.2.5 useDeprecatedFunction()

```
virtual bool IRawImageReceivedObserver::useDeprecatedFunction ( ) [inline], [virtual]
```

If return true then "imageReceived" is called, else "imageReceivedBeforeBuffer" or "imageReceivedAfterBuffer" is called.

6.20 Ocam2Conf Struct Reference

```
#include <FliOcam2K.h>
```


Public Attributes

- [Ocam2Mode wmode](#)
- `std::string configFile`
- `uint16_t width`
- `uint16_t height`
- `uint16_t nbPixels`
- `uint16_t rawWidth`
- `uint16_t rawHeight`
- `uint16_t rawNbPixels`
- `uint16_t binningOffset`
- `uint16_t nbIdenticalPixels`
- `uint16_t fpsMax`

6.20.1 Member Data Documentation

6.20.1.1 binningOffset

`uint16_t Ocam2Conf::binningOffset`

6.20.1.2 configFile

`std::string Ocam2Conf::configFile`

6.20.1.3 fpsMax

`uint16_t Ocam2Conf::fpsMax`

6.20.1.4 height

`uint16_t Ocam2Conf::height`

6.20.1.5 nbIdenticalPixels

`uint16_t Ocam2Conf::nbIdenticalPixels`

6.20.1.6 nbPixels

`uint16_t Ocam2Conf::nbPixels`

6.20.1.7 rawHeight

`uint16_t Ocam2Conf::rawHeight`

6.20.1.8 rawNbPixels

`uint16_t Ocam2Conf::rawNbPixels`

6.20.1.9 rawWidth

`uint16_t Ocam2Conf::rawWidth`

6.20.1.10 width

`uint16_t Ocam2Conf::width`

6.20.1.11 wmode

[Ocam2Mode](#) `Ocam2Conf::wmode`

Chapter 7

File Documentation

7.1 FliCblueOne.h File Reference

Classes

- class [FliCblueOne](#)

7.2 FliCblueOneEnum.h File Reference

Namespaces

- [FliCblueOneEnum](#)

Enumerations

- enum [FliCblueOneEnum::DeviceTemperatureSelectorEnum](#) : int64_t {
 [FliCblueOneEnum::DeviceTemperatureSelectorEnum::Sensor](#) = 0, [FliCblueOneEnum::DeviceTemperatureSelectorEnum::CPU](#)
 = 1, [FliCblueOneEnum::DeviceTemperatureSelectorEnum::Power](#) = 2, [FliCblueOneEnum::DeviceTemperatureSelectorEnum::Fan](#)
 = 3,
 [FliCblueOneEnum::DeviceTemperatureSelectorEnum::Heatsink](#) = 4, [FliCblueOneEnum::DeviceTemperatureSelectorEnum::Case](#)
 = 5 }
- enum [FliCblueOneEnum::DeviceTecSelectorEnum](#) : int64_t { [FliCblueOneEnum::DeviceTecSelectorEnum::TEC1](#)
 = 0 }
- enum [FliCblueOneEnum::DeviceFanModeEnum](#) : int64_t { [FliCblueOneEnum::DeviceFanModeEnum::Automatic](#)
 = 0, [FliCblueOneEnum::DeviceFanModeEnum::Manual](#) = 1 }
- enum [FliCblueOneEnum::FirmwareUpdateStatusEnum](#) : int64_t { [FliCblueOneEnum::FirmwareUpdateStatusEnum::Idle](#)
 = 0, [FliCblueOneEnum::FirmwareUpdateStatusEnum::InProgress](#) = 1, [FliCblueOneEnum::FirmwareUpdateStatusEnum::Done](#)
 = 2, [FliCblueOneEnum::FirmwareUpdateStatusEnum::Failed](#) = 3 }
- enum [FliCblueOneEnum::LogCollectStatusEnum](#) : int64_t { [FliCblueOneEnum::LogCollectStatusEnum::Idle](#)
 = 0, [FliCblueOneEnum::LogCollectStatusEnum::InProgress](#) = 1, [FliCblueOneEnum::LogCollectStatusEnum::Done](#)
 = 2, [FliCblueOneEnum::LogCollectStatusEnum::Failed](#) = 3 }
- enum [FliCblueOneEnum::IPModeEnum](#) : int64_t { [FliCblueOneEnum::IPModeEnum::Automatic](#) = 0,
 [FliCblueOneEnum::IPModeEnum::Manual](#) = 1 }

- enum `FliCblueOneEnum::SparseSelectorEnum` : `int64_t` {
`FliCblueOneEnum::SparseSelectorEnum::Region0` = 0, `FliCblueOneEnum::SparseSelectorEnum::Region1` = 1, `FliCblueOneEnum::SparseSelectorEnum::Region2` = 2, `FliCblueOneEnum::SparseSelectorEnum::Region3` = 3,
`FliCblueOneEnum::SparseSelectorEnum::Region4` = 4, `FliCblueOneEnum::SparseSelectorEnum::Region5` = 5, `FliCblueOneEnum::SparseSelectorEnum::Region6` = 6, `FliCblueOneEnum::SparseSelectorEnum::Region7` = 7 }
- enum `FliCblueOneEnum::SparseModeEnum` : `int64_t` { `FliCblueOneEnum::SparseModeEnum::Off` = 0, `FliCblueOneEnum::SparseModeEnum::On` = 1 }
- enum `FliCblueOneEnum::TestPatternGeneratorSelectorEnum` : `int64_t` { `FliCblueOneEnum::TestPatternGeneratorSelectorEnum` = 0, `FliCblueOneEnum::TestPatternGeneratorSelectorEnum::Simulator` = 1 }
- enum `FliCblueOneEnum::TestPatternEnum` : `int64_t` {
`FliCblueOneEnum::TestPatternEnum::Off` = 0, `FliCblueOneEnum::TestPatternEnum::Black` = 1, `FliCblueOneEnum::TestPatternEnum` = 2, `FliCblueOneEnum::TestPatternEnum::GreyHorizontalRamp` = 3,
`FliCblueOneEnum::TestPatternEnum::SimulatorGreyHorizontalRamp` = 10, `FliCblueOneEnum::TestPatternEnum::SimulatorGrey` = 11 }
- enum `FliCblueOneEnum::GlowReductionEnum` : `int64_t` { `FliCblueOneEnum::GlowReductionEnum::Off` = 0, `FliCblueOneEnum::GlowReductionEnum::On` = 1 }
- enum `FliCblueOneEnum::ConversionEfficiencyEnum` : `int64_t` { `FliCblueOneEnum::ConversionEfficiencyEnum::Low` = 0, `FliCblueOneEnum::ConversionEfficiencyEnum::High` = 1 }
- enum `FliCblueOneEnum::UserSetSelectorEnum` : `int64_t` {
`FliCblueOneEnum::UserSetSelectorEnum::Default8bits` = 30, `FliCblueOneEnum::UserSetSelectorEnum::Default12bits` = 32, `FliCblueOneEnum::UserSetSelectorEnum::HighSensitivity8bits` = 40, `FliCblueOneEnum::UserSetSelectorEnum::HighSensitivity12bits` = 42,
`FliCblueOneEnum::UserSetSelectorEnum::UserSet0` = 0, `FliCblueOneEnum::UserSetSelectorEnum::UserSet1` = 1, `FliCblueOneEnum::UserSetSelectorEnum::UserSet2` = 2, `FliCblueOneEnum::UserSetSelectorEnum::UserSet3` = 3,
`FliCblueOneEnum::UserSetSelectorEnum::UserSet4` = 4, `FliCblueOneEnum::UserSetSelectorEnum::UserSet5` = 5, `FliCblueOneEnum::UserSetSelectorEnum::UserSet6` = 6, `FliCblueOneEnum::UserSetSelectorEnum::UserSet7` = 7,
`FliCblueOneEnum::UserSetSelectorEnum::UserSet8` = 8, `FliCblueOneEnum::UserSetSelectorEnum::UserSet9` = 9 }
- enum `FliCblueOneEnum::UserSetDefaultEnum` : `int64_t` {
`FliCblueOneEnum::UserSetDefaultEnum::Default8bits` = 30, `FliCblueOneEnum::UserSetDefaultEnum::Default12bits` = 32, `FliCblueOneEnum::UserSetDefaultEnum::HighSensitivity8bits` = 40, `FliCblueOneEnum::UserSetDefaultEnum::HighSensitivity12bits` = 42,
`FliCblueOneEnum::UserSetDefaultEnum::UserSet0` = 0, `FliCblueOneEnum::UserSetDefaultEnum::UserSet1` = 1, `FliCblueOneEnum::UserSetDefaultEnum::UserSet2` = 2, `FliCblueOneEnum::UserSetDefaultEnum::UserSet3` = 3,
`FliCblueOneEnum::UserSetDefaultEnum::UserSet4` = 4, `FliCblueOneEnum::UserSetDefaultEnum::UserSet5` = 5, `FliCblueOneEnum::UserSetDefaultEnum::UserSet6` = 6, `FliCblueOneEnum::UserSetDefaultEnum::UserSet7` = 7,
`FliCblueOneEnum::UserSetDefaultEnum::UserSet8` = 8, `FliCblueOneEnum::UserSetDefaultEnum::UserSet9` = 9 }

Variables

- `const std::map< std::string, int64_t > FliCblueOneEnum::DeviceTemperatureSelectorStringToValue`
- `const std::map< std::string, int64_t > FliCblueOneEnum::DeviceTecSelectorStringToValue`
- `const std::map< std::string, int64_t > FliCblueOneEnum::DeviceFanModeStringToValue`
- `const std::map< std::string, int64_t > FliCblueOneEnum::FirmwareUpdateStatusStringToValue`
- `const std::map< std::string, int64_t > FliCblueOneEnum::LogCollectStatusStringToValue`
- `const std::map< std::string, int64_t > FliCblueOneEnum::IPModeStringToValue`
- `const std::map< std::string, int64_t > FliCblueOneEnum::SparseSelectorStringToValue`
- `const std::map< std::string, int64_t > FliCblueOneEnum::SparseModeStringToValue`
- `const std::map< std::string, int64_t > FliCblueOneEnum::TestPatternGeneratorSelectorStringToValue`

- `const std::map< std::string, int64_t > FliCblueOneEnum::TestPatternStringToValue`
- `const std::map< std::string, int64_t > FliCblueOneEnum::GlowReductionStringToValue`
- `const std::map< std::string, int64_t > FliCblueOneEnum::ConversionEfficiencyStringToValue`
- `const std::map< std::string, int64_t > FliCblueOneEnum::UserSetSelectorStringToValue`
- `const std::map< std::string, int64_t > FliCblueOneEnum::UserSetDefaultStringToValue`

7.3 FliCblueSfncEnum.h File Reference

Namespaces

- [FliCblueSfncEnum](#)

Enumerations

- enum `FliCblueSfncEnum::DeviceScanTypeEnum` : `int64_t` { `FliCblueSfncEnum::DeviceScanTypeEnum::Areascan` = 0 }
- enum `FliCblueSfncEnum::DeviceIndicatorModeEnum` : `int64_t` { `FliCblueSfncEnum::DeviceIndicatorModeEnum::Inactive` = 0, `FliCblueSfncEnum::DeviceIndicatorModeEnum::Active` = 1, `FliCblueSfncEnum::DeviceIndicatorModeEnum::ErrorStatus` = 2 }
- enum `FliCblueSfncEnum::SensorShutterModeEnum` : `int64_t` { `FliCblueSfncEnum::SensorShutterModeEnum::Global` = 0, `FliCblueSfncEnum::SensorShutterModeEnum::Rolling` = 1, `FliCblueSfncEnum::SensorShutterModeEnum::GlobalReset` = 2 }
- enum `FliCblueSfncEnum::RegionSelectorEnum` : `int64_t` { `FliCblueSfncEnum::RegionSelectorEnum::Region0` = 0 }
- enum `FliCblueSfncEnum::RegionModeEnum` : `int64_t` { `FliCblueSfncEnum::RegionModeEnum::Off` = 0, `FliCblueSfncEnum::RegionModeEnum::On` = 1 }
- enum `FliCblueSfncEnum::RegionDestinationEnum` : `int64_t` { `FliCblueSfncEnum::RegionDestinationEnum::Stream0` = 0 }
- enum `FliCblueSfncEnum::PixelFormatEnum` : `int64_t` { `FliCblueSfncEnum::PixelFormatEnum::Mono8` = 0, `FliCblueSfncEnum::PixelFormatEnum::Mono10` = 1, `FliCblueSfncEnum::PixelFormatEnum::Mono12` = 2 }
- enum `FliCblueSfncEnum::AcquisitionModeEnum` : `int64_t` { `FliCblueSfncEnum::AcquisitionModeEnum::Continuous` = 0 }
- enum `FliCblueSfncEnum::ExposureModeEnum` : `int64_t` { `FliCblueSfncEnum::ExposureModeEnum::Timed` = 0 }
- enum `FliCblueSfncEnum::GainSelectorEnum` : `int64_t` { `FliCblueSfncEnum::GainSelectorEnum::AnalogAll` = 0, `FliCblueSfncEnum::GainSelectorEnum::DigitalAll` = 1 }
- enum `FliCblueSfncEnum::BlackLevelSelectorEnum` : `int64_t` { `FliCblueSfncEnum::BlackLevelSelectorEnum::All` = 0 }
- enum `FliCblueSfncEnum::BlackLevelAutoEnum` : `int64_t` { `FliCblueSfncEnum::BlackLevelAutoEnum::Off` = 0, `FliCblueSfncEnum::BlackLevelAutoEnum::Continuous` = 1 }
- enum `FliCblueSfncEnum::CxpLinkConfigurationStatusEnum` : `int64_t` { `FliCblueSfncEnum::CxpLinkConfigurationStatusEnum::CXP1_X1` = 0, `FliCblueSfncEnum::CxpLinkConfigurationStatusEnum::CXP1_X2` = 1, `FliCblueSfncEnum::CxpLinkConfigurationStatusEnum::CXP10_X2` = 2, `FliCblueSfncEnum::CxpLinkConfigurationStatusEnum::CXP12_X2` = 3, `FliCblueSfncEnum::CxpLinkConfigurationStatusEnum::CXP10_X2` = 4, `FliCblueSfncEnum::CxpLinkConfigurationStatusEnum::CXP12_X2` = 5 }
- enum `FliCblueSfncEnum::CxpLinkConfigurationPreferredEnum` : `int64_t` { `FliCblueSfncEnum::CxpLinkConfigurationPreferredEnum::CXP6_X2` = 0, `FliCblueSfncEnum::CxpLinkConfigurationPreferredEnum::CXP6_X2` = 1, `FliCblueSfncEnum::CxpLinkConfigurationPreferredEnum::CXP12_X2` = 2, `FliCblueSfncEnum::CxpLinkConfigurationPreferredEnum::CXP12_X2` = 3 }
- enum `FliCblueSfncEnum::CxpLinkConfigurationEnum` : `int64_t` { `FliCblueSfncEnum::CxpLinkConfigurationEnum::CXP10_X2` = 0 }
- enum `FliCblueSfncEnum::CxpConnectionTestModeEnum` : `int64_t` { `FliCblueSfncEnum::CxpConnectionTestModeEnum::Off` = 0, `FliCblueSfncEnum::CxpConnectionTestModeEnum::Mode1` = 1 }

- enum [FliCblueSfncEnum::CxpSendReceiveSelectorEnum](#) : int64_t { [FliCblueSfncEnum::CxpSendReceiveSelectorEnum::Send](#) = 0, [FliCblueSfncEnum::CxpSendReceiveSelectorEnum::Receive](#) = 1 }
- enum [FliCblueSfncEnum::CxpErrorCounterSelectorEnum](#) : int64_t { [FliCblueSfncEnum::CxpErrorCounterSelectorEnum::ConnectionLockLoss](#) = 0, [FliCblueSfncEnum::CxpErrorCounterSelectorEnum::StreamDataPacketCrc](#) = 1, [FliCblueSfncEnum::CxpErrorCounterSelectorEnum::EventPacketCrc](#) = 2, [FliCblueSfncEnum::CxpErrorCounterSelectorEnum::DuplicatedCharactersUncorrected](#) = 3, [FliCblueSfncEnum::CxpErrorCounterSelectorEnum::CounterOverflow](#) = 4, [FliCblueSfncEnum::CxpErrorCounterSelectorEnum::CounterOverflow](#) = 5, [FliCblueSfncEnum::CxpErrorCounterSelectorEnum::CounterOverflow](#) = 6 }
- enum [FliCblueSfncEnum::CxpErrorCounterStatusEnum](#) : int64_t { [FliCblueSfncEnum::CxpErrorCounterStatusEnum::CounterOverflow](#) = 0, [FliCblueSfncEnum::CxpErrorCounterStatusEnum::CounterOverflow](#) = 1 }

Variables

- const std::vector< std::string > [FliCblueSfncEnum::DeviceScanTypeString](#)
- const std::vector< std::string > [FliCblueSfncEnum::DeviceIndicatorModeString](#)
- const std::vector< std::string > [FliCblueSfncEnum::SensorShutterModeString](#)
- const std::vector< std::string > [FliCblueSfncEnum::RegionSelectorString](#)
- const std::vector< std::string > [FliCblueSfncEnum::RegionModeString](#)
- const std::vector< std::string > [FliCblueSfncEnum::RegionDestinationString](#)
- const std::vector< std::string > [FliCblueSfncEnum::PixelFormatString](#)
- const std::vector< std::string > [FliCblueSfncEnum::AcquisitionModeString](#)
- const std::vector< std::string > [FliCblueSfncEnum::ExposureModeString](#)
- const std::vector< std::string > [FliCblueSfncEnum::GainSelectorString](#)
- const std::vector< std::string > [FliCblueSfncEnum::BlackLevelSelectorString](#)
- const std::vector< std::string > [FliCblueSfncEnum::BlackLevelAutoString](#)
- const std::vector< std::string > [FliCblueSfncEnum::CxpLinkConfigurationStatusString](#)
- const std::vector< std::string > [FliCblueSfncEnum::CxpLinkConfigurationPreferredString](#)
- const std::vector< std::string > [FliCblueSfncEnum::CxpLinkConfigurationString](#)
- const std::vector< std::string > [FliCblueSfncEnum::CxpConnectionTestModeString](#)
- const std::vector< std::string > [FliCblueSfncEnum::CxpSendReceiveSelectorString](#)
- const std::vector< std::string > [FliCblueSfncEnum::CxpErrorCounterSelectorString](#)
- const std::vector< std::string > [FliCblueSfncEnum::CxpErrorCounterStatusString](#)
- const std::vector< std::string > [FliCblueSfncEnum::featuresListString](#)

7.4 FliCblueTwo.h File Reference

Classes

- class [FliCblueTwo](#)

7.5 FliCblueTwoEnum.h File Reference

Namespaces

- [FliCblueTwoEnum](#)

Enumerations

- enum [FliCblueTwoEnum::BinningSelectorEnum](#) : int64_t { [FliCblueTwoEnum::BinningSelectorEnum::Sensor](#) = 0 }
Selects which binning engine is controlled by the [BinningHorizontal](#) and [BinningVertical](#) features.
- enum [FliCblueTwoEnum::BinningHorizontalModeEnum](#) : int64_t { [FliCblueTwoEnum::BinningHorizontalModeEnum::Sum](#) = 0, [FliCblueTwoEnum::BinningHorizontalModeEnum::Average](#) = 1 }
Sets the mode to use to combine horizontal photo-sensitive cells together when [BinningHorizontal](#) is used.
- enum [FliCblueTwoEnum::BinningVerticalModeEnum](#) : int64_t { [FliCblueTwoEnum::BinningVerticalModeEnum::Sum](#) = 0, [FliCblueTwoEnum::BinningVerticalModeEnum::Average](#) = 1 }
Sets the mode to use to combine vertical photo-sensitive cells together when [BinningVertical](#) is used.
- enum [FliCblueTwoEnum::FirmwareUpdateStatusEnum](#) : int64_t { [FliCblueTwoEnum::FirmwareUpdateStatusEnum::Idle](#) = 0, [FliCblueTwoEnum::FirmwareUpdateStatusEnum::InProgress](#) = 1, [FliCblueTwoEnum::FirmwareUpdateStatusEnum::Done](#) = 2, [FliCblueTwoEnum::FirmwareUpdateStatusEnum::Failed](#) = 3 }

Variables

- const std::map< std::string, int64_t > [FliCblueTwoEnum::BinningSelectorStringToValue](#)
- const std::map< std::string, int64_t > [FliCblueTwoEnum::BinningHorizontalModeStringToValue](#)
- const std::map< std::string, int64_t > [FliCblueTwoEnum::BinningVerticalModeStringToValue](#)
- const std::map< std::string, int64_t > [FliCblueTwoEnum::FirmwareUpdateStatusStringToValue](#)

7.6 FliCred.h File Reference

Classes

- class [FliCred](#)
This class manages the methods common to all the C-RED camera (C-RED One, C-RED 2, C-RED 2 Lite, C-RED 2 ER, C-RED 3)

Typedefs

- typedef [FliCred](#) [FliCamera](#)

7.6.1 Typedef Documentation

7.6.1.1 FliCamera

```
typedef FliCred FliCamera
```

7.7 FliCredOne.h File Reference

Classes

- class [FliCredOne](#)
This class manages the methods specific to the C-RED One camera.

7.8 FliCredThree.h File Reference

Classes

- class [FliCredThree](#)

This class manages the methods specific to the C-RED 3 camera.

7.9 FliCredTwo.h File Reference

Classes

- class [FliCredTwo](#)

This class manages the methods specific to the C-RED 2 and C-RED 2 ER cameras.

7.10 FliCredTwoLite.h File Reference

Classes

- class [FliCredTwoLite](#)

This class manages the methods specific to the C-RED 2 Lite camera.

7.11 FliGenicamCamera.h File Reference

Classes

- class [FliGenicamCamera](#)

This is the base class of all genicam camera (C-BLUE)

7.12 FliOcam2K.h File Reference

Classes

- struct [Ocam2Conf](#)
- class [FliOcam2K](#)

This class manages the methods specific to the OCAM2K camera.

Enumerations

- enum [Ocam2Mode](#) {
[Ocam2Mode::OCAM2_UNKNOWN](#) = 0, [Ocam2Mode::OCAM2_NORMAL](#) = 1, [Ocam2Mode::OCAM2_CROPPING240x120](#) = 2, [Ocam2Mode::OCAM2_BINNING2x2](#) = 3,
[Ocam2Mode::OCAM2_BINNING3x3](#) = 4, [Ocam2Mode::OCAM2_BINNING4x4](#) = 5, [Ocam2Mode::OCAM2_BINNING](#) = [OCAM2_BINNING2x2](#), [Ocam2Mode::OCAM2_CROPPING240x128](#) = 6,
[Ocam2Mode::OCAM2_2_TRACK](#) = 7, [Ocam2Mode::OCAM2_4_TRACK](#) = 8, [Ocam2Mode::OCAM2_BINNING1x3](#) = 9, [Ocam2Mode::OCAM2_BINNING1x4](#) = 10 }
- enum [Ocam2CoolingState](#) { [Ocam2CoolingState::on](#), [Ocam2CoolingState::off](#), [Ocam2CoolingState::alarm](#) }

7.12.1 Enumeration Type Documentation

7.12.1.1 Ocam2CoolingState

```
enum Ocam2CoolingState [strong]
```

Enumerator

on	
off	
alarm	

7.12.1.2 Ocam2Mode

```
enum Ocam2Mode [strong]
```

Enumerator

OCAM2_UNKNOWN	Invalid.
OCAM2_NORMAL	Default mode.
OCAM2_CROPPING240x120	Cropping 120.
OCAM2_BINNING2x2	Binning 2x2.
OCAM2_BINNING3x3	Binning 3x3.
OCAM2_BINNING4x4	Binning 4x4.
OCAM2_BINNING	For compatibility(= Binning 2x2)
OCAM2_CROPPING240x128	Cropping 128.
OCAM2_2_TRACK	Binning 2 lignes.
OCAM2_4_TRACK	Binning 4 lignes.
OCAM2_BINNING1x3	Binning 1x3.
OCAM2_BINNING1x4	Binning 1x4.

7.13 FliOcam2S.h File Reference

Classes

- class [FliOcam2S](#)

This class manages the methods specific to the OCAM2S camera.

7.14 FliRingBuffer.h File Reference

Classes

- class [FliRingBuffer](#)

7.15 FliSdk.h File Reference

Classes

- class [IRawImageReceivedObserver](#)
This can be herited to be an observer of the reception of a raw image.
- class [FliSdk](#)
This class manages the interface with the camera and the grabber.

7.16 FliSdk_C_V2.h File Reference

Typedefs

- typedef void(* [newImageAvailableCallback](#)) (const uint8_t *image, void *ctx)
- typedef bool(* [saveBufferProgressionCallback](#)) (int progress)
- typedef void * [callbackHandler](#)

Functions

- EXPORTED [FliContext](#) [FliSdk_init_V2](#) ()
Create a SDK context.
- EXPORTED void [FliSdk_exit_V2](#) (FliContext context)
Delete the SDK context.
- EXPORTED void [FliSdk_detectGrabbers_V2](#) (FliContext context, char *listOfGrabbers, size_t size)
Start the grabbers detection and return a list with the names of detected grabbers separated by ";".
- EXPORTED void [FliSdk_getDetectedGrabbers_V2](#) (FliContext context, char *listOfGrabbers, size_t size)
Get the list with the names of detected grabbers separated by ";".
- EXPORTED void [FliSdk_detectCameras_V2](#) (FliContext context, char *listOfCameras, size_t size)
Start the cameras detection and return a list with the names of detected cameras separated by ";".
- EXPORTED void [FliSdk_getDetectedCameras_V2](#) (FliContext context, char *listOfCameras, size_t size)
Get the list with the names of detected cameras separated by ";".
- EXPORTED bool [FliSdk_setGrabber_V2](#) (FliContext context, const char *grabberName)
Set the grabber to be used.
- EXPORTED bool [FliSdk_setCamera_V2](#) (FliContext context, const char *cameraName)
Set the camera to be used.
- EXPORTED bool [FliSdk_getCurrentCameraName_V2](#) (FliContext context, char *cameraName, size_t size)
Get the name of the current camera used by the SDK.
- EXPORTED void [FliSdk_setMode_V2](#) (FliContext context, Mode mode)
Set the mode of use of the SDK.
- EXPORTED void [FliSdk_setImageDimension_V2](#) (FliContext context, uint16_t width, uint16_t height)
Force the image dimension apply to the grabber.
- EXPORTED bool [FliSdk_update_V2](#) (FliContext context)
Update the changes, must be call after setCamera, setGrabber or setMode to take effects.
- EXPORTED bool [FliSdk_start_V2](#) (FliContext context)
Start the grabber (must be initialized before)
- EXPORTED bool [FliSdk_stop_V2](#) (FliContext context)
Stop the grabber.
- EXPORTED bool [FliSdk_isStarted_V2](#) (FliContext context)

- Get the state of the grabber (started or stopped)*

 - EXPORTED CameraModel_C [FliSdk_getCurrentCameraModel_V2](#) (FliContext context)

returns the current camera model
- EXPORTED void [FliSdk_getCameraModelAsString_V2](#) (FliContext context, char *model, size_t size)

returns the current camera model as a string
- EXPORTED bool [FliSdk_enableGrabN_V2](#) (FliContext context, uint32_t nbFrames)

Enable grab N mode.
- EXPORTED bool [FliSdk_disableGrabN_V2](#) (FliContext context)

Disable grab N mode.
- EXPORTED bool [FliSdk_isGrabNEnabled_V2](#) (FliContext context)

State of the grab N.
- EXPORTED bool [FliSdk_isGrabNFinished_V2](#) (FliContext context)

State of the grab N.
- const EXPORTED unsigned char * [FliSdk_getRawImage_V2](#) (FliContext context, int64_t index)

Get the image at index or the last image if index is -1, without processing.
- EXPORTED void [FliSdk_display8bImage_V2](#) (FliContext context, uint8_t *image, const char *windowName)

Open an Opencv window to display image.
- EXPORTED void [FliSdk_display16bImage_V2](#) (FliContext context, uint8_t *image, const char *windowName, bool unsignedPixels)

Open an Opencv window to display image.
- EXPORTED void [FliSdk_initLog_V2](#) (FliContext context, const char *appName)

init SDK logging
- EXPORTED uint32_t [FliSdk_getFps_V2](#) (FliContext context)

Get the buffer acquisition rate.
- EXPORTED uint32_t [FliSdk_getBufferFilling_V2](#) (FliContext context)

Get buffer filling.
- EXPORTED void [FliSdk_setBufferSizeInImages_V2](#) (FliContext context, uint64_t nblImages)

Change the buffer capacity in number of images.
- EXPORTED void [FliSdk_setBufferSize_V2](#) (FliContext context, uint16_t sizeMo)

Change the buffer capacity in Mo.
- EXPORTED uint16_t [FliSdk_getBufferSize_V2](#) (FliContext context)

Get current buffer size.
- EXPORTED bool [FliSdk_loadBufferFromFile_V2](#) (FliContext context, const char *path, CroppingData_C *bufferCrop)

Load a buffer from a file, in the ringBuffer of the SDK.
- EXPORTED bool [FliSdk_loadBufferRaw_V2](#) (FliContext context, const uint8_t *buffer, uint32_t nblImages)

Load a buffer in the ringBuffer of the SDK.
- EXPORTED bool [FliSdk_getBufferWithInfo_V2](#) (FliContext context, const char *path, LoadBufferInfo_C *info)

Load a buffer from a file, allocate memory, and return that memory to the user.
- EXPORTED void [FliSdk_resetBuffer_V2](#) (FliContext context)

Reset the buffer.
- EXPORTED bool [FliSdk_saveBuffer_V2](#) (FliContext context, const char *path, uint32_t startIndex, uint32_t endIndex)

Save the buffer at path.
- EXPORTED bool [FliSdk_saveBufferWithOptions_V2](#) (FliContext context, const char *path, uint32_t startIndex, uint32_t endIndex, [saveBufferProgressionCallback](#) func, bool withMetadata, uint16_t offset, bool forceUnsigned, uint16_t decimation)

Save the buffer at path.
- EXPORTED uint32_t [FliSdk_getImagesCapacity_V2](#) (FliContext context)

Give the images capacity of the buffer.
- EXPORTED uint8_t * [FliSdk_getProcessedImage_V2](#) (FliContext context, int64_t index)

Get the RGB processed image at the given index, if -1 then the last image is processed The buffer is overwritten only when the function is recalled.

- EXPORTED uint8_t * [FliSdk_getProcessedImage16b_V2](#) (FliContext context, int64_t index)

Get the 16bits grayscale processed image at the given index, if -1 then the last image is processed The buffer is overwritten only when the function is recalled.
- EXPORTED bool [FliSdk_isCroppingDataValid_V2](#) (FliContext context, CroppingData_C croppingData)

Check if the cropping data is valid for Cred2 & Cred3.
- EXPORTED bool [FliSdk_getCroppingState_V2](#) (FliContext context, bool *isEnabled, CroppingData_C *croppingData)

Get the cropping data from the camera.
- EXPORTED bool [FliSdk_setCroppingState_V2](#) (FliContext context, bool enable, CroppingData_C croppingData)

Set the cropping data.
- EXPORTED void [FliSdk_getCurrentImageDimension_V2](#) (FliContext context, uint16_t *width, uint16_t *height)

Get the current image dimension considering cropping.
- EXPORTED [callbackHandler](#) [FliSdk_addCallbackNewImage_V2](#) (FliContext context, [newImageAvailableCallBack](#) func, uint16_t fpsTrigger, bool beforeCopy, void *ctx)

Add a callback in order to receive frames.
- EXPORTED void [FliSdk_removeCallbackNewImage_V2](#) (FliContext context, [callbackHandler](#) ctx)

Remove callback.
- EXPORTED void [FliSdk_setFpsTrigger_V2](#) (FliContext context, [callbackHandler](#) ctx, uint16_t fps)

Add a callback in order to receive frames.
- EXPORTED bool [FliSdk_isCredOne_V2](#) (FliContext context)
- EXPORTED bool [FliSdk_isCredTwo_V2](#) (FliContext context)
- EXPORTED bool [FliSdk_isCredTwoLite_V2](#) (FliContext context)
- EXPORTED bool [FliSdk_isCredThree_V2](#) (FliContext context)
- EXPORTED bool [FliSdk_isCblueOne_V2](#) (FliContext context)
- EXPORTED bool [FliSdk_isCblueTwo_V2](#) (FliContext context)
- EXPORTED bool [FliSdk_isCred_V2](#) (FliContext context)
- EXPORTED bool [FliSdk_isCblueSfnc_V2](#) (FliContext context)
- EXPORTED bool [FliSdk_isOcam2k_V2](#) (FliContext context)
- EXPORTED bool [FliSdk_isOcam2s_V2](#) (FliContext context)
- EXPORTED bool [FliSdk_isSerialCamera_V2](#) (FliContext context)
- EXPORTED void [FliSdk_forceCurrentCameraModel_V2](#) (FliContext context, CameraModel_C model)

use this function when a camera is undefined
- EXPORTED void [FliSdk_setNbImagesPerBuffer_V2](#) (FliContext context, uint8_t nbImages)

Set set number of images the grabber should acquire before trigger, use this function for high FPS.
- EXPORTED bool [FliSdk_isUnsignedPixel_V2](#) (FliContext context)

Return the pixel sign (int16 or uint16)
- EXPORTED bool [FliSdk_isMono8Pixel_V2](#) (FliContext context)

Return the pixel size (1 byte if true, 2 bytes if false)
- EXPORTED void [FliSdk_enableUnsignedPixel_V2](#) (FliContext context, bool enable)

Change the pixel sign (int16 or uint16)
- EXPORTED void [FliSdk_enableRingBuffer_V2](#) (FliContext context, bool enable)

Enable or disable internal ring buffer of the SDK.
- EXPORTED void [FliSdk_getAvailableSaveFormats_V2](#) (FliContext context, char *fullName, size_t fullNameSize, char *extension, size_t extensionSize)

Return a list with the formats full name and a list with the formats extension, each item is separated by a " , ".
- EXPORTED uint64_t [FliSdk_getNbCountError_V2](#) (FliContext context)

Get the number of frame count error.
- EXPORTED unsigned int [FliSdk_getOcamFrameNumber_V2](#) (FliContext context, int64_t index)

- Get the frame number of ocam image at index.*

 - EXPORTED void [FliSdk_setOcamFrameNumberOffset_V2](#) (FliContext context, uint8_t offset)

Set the offset for the frame number.
- EXPORTED void [FliSdk_setBurstFilter_V2](#) (FliContext context, int16_t id)

Set the burst filter for id.
- EXPORTED int16_t [FliSdk_getBurstFilter_V2](#) (FliContext context)

Get the current burst filter id applied.
- EXPORTED void [FliSdk_enableSubstractMode_V2](#) (FliContext context, bool enable)

Enable/disable the mode substract that will substract the image N by the image N-1 and save it in the buffer.
- EXPORTED void [FliSdk_enableFowlerProcessing_V2](#) (FliContext context, bool enable)

Enable/disable the Fowler processing for C-RED 1 cameras.
- EXPORTED void [FliSdk_setFowlerOffset_V2](#) (FliContext context, int16_t offset)

Set the value of the Fowler offset to apply on sum of the images.
- EXPORTED void [FliSdk_enableFollowUpTheRamp_V2](#) (FliContext context, bool enable)

Enable/disable the initialisation of the pixel sum for the follow up.
- EXPORTED bool [FliSdk_addEthernetCamera_V2](#) (FliContext context, const char *ip, const char *userName, const char *sshPassword, char *cameraName, size_t size)

Try to detect an ethernet camera and add it in the list.
- EXPORTED ProcessingId [FliSdk_addImageProcessing_V2](#) (FliContext context)

add an image processing and return an id
- EXPORTED void [FliSdk_removeImageProcessing_V2](#) (FliContext context, ProcessingId id)

remove an image processing
- EXPORTED void [FliSdk_getProcessedImage_lv_V2](#) (FliContext context, int64_t index, uint32_t *image)

Get the RGB processed image at the given index, if -1 then the last image is processed The buffer is overwritten only when the function is recalled.
- EXPORTED void [FliSdk_getProcessedImage16b_lv_V2](#) (FliContext context, int64_t index, uint16_t *image)

Get the RGB processed image at the given index, if -1 then the last image is processed The buffer is overwritten only when the function is recalled.
- EXPORTED void [FliSdk_getRawImage_lv_V2](#) (FliContext context, int64_t index, uint16_t *image)

Get the image at index or the last image if index is -1, without processing.

7.16.1 Typedef Documentation

7.16.1.1 callbackHandler

```
typedef void* callbackHandler
```

7.16.1.2 newImageAvailableCallBack

```
typedef void(* newImageAvailableCallBack) (const uint8_t *image, void *ctx)
```

7.16.1.3 saveBufferProgressionCallback

```
typedef bool(* saveBufferProgressionCallback) (int progress)
```

7.16.2 Function Documentation

7.16.2.1 FliSdk_addCallbackNewImage_V2()

```
EXPORTED callbackHandler FliSdk_addCallbackNewImage_V2 (
    FliContext context,
    newImageAvailableCallBack func,
    uint16_t fpsTrigger,
    bool beforeCopy,
    void * ctx )
```

Add a callback in order to receive frames.

Parameters

<i>context</i>	SDK context
<i>func</i>	callback function
<i>fpsTrigger</i>	at which fps does the callback want to receive images, if 0 then full speed
<i>beforeCopy</i>	if true then the observer will be notified before the copy in the ringbuffer (image from grabber), else after the copy in the ringBuffer (image from ringBuffer). if beforeCopy is set to true, user will have only the time of the buffer overflow of the grabber but less time between the grabber and the notification. if beforeCopy is set to false, user will have more time because the ringBuffer can be bigger than the grabber buffer but it will have a copy between the grabber and the notification. If you want to switch from before to after or after to before then call removeRawImageReceivedObserver before addRawImageReceivedObserver
<i>ctx</i>	user context return in the callback function

Returns

a callbackHandler

7.16.2.2 FliSdk_addEthernetCamera_V2()

```
EXPORTED bool FliSdk_addEthernetCamera_V2 (
    FliContext context,
    const char * ip,
    const char * userName,
    const char * sshPassword,
    char * cameraName,
    size_t size )
```

Try to detect an ethernet camera and add it in the list.

Parameters

<i>context</i>	SDK context
<i>ip</i>	: ip of the camera or a range of ip to auto detect (ex: 192.168.0.1-60)
<i>userName</i>	: the ssh user name of the camera
<i>sshPassword</i>	: the ssh password of the camera
<i>cameraName</i>	: return the detected camera name
<i>size</i>	size of cameraName array

7.16.2.3 FliSdk_addImageProcessing_V2()

```
EXPORTED ProcessingId FliSdk_addImageProcessing_V2 (
    FliContext context )
```

add an image processing and return an id

Parameters

<i>context</i>	SDK context
----------------	-------------

7.16.2.4 FliSdk_detectCameras_V2()

```
EXPORTED void FliSdk_detectCameras_V2 (
    FliContext context,
    char * listOfCameras,
    size_t size )
```

Start the cameras detection and return a list with the names of detected cameras separated by ";".

Parameters

<i>context</i>	SDK context
<i>listOfCameras</i>	char array allocated by user
<i>size</i>	size of listOfCameras

7.16.2.5 FliSdk_detectGrabbers_V2()

```
EXPORTED void FliSdk_detectGrabbers_V2 (
    FliContext context,
    char * listOfGrabbers,
    size_t size )
```

Start the grabbers detection and return a list with the names of detected grabbers separated by ";".

Parameters

<i>context</i>	SDK context
<i>listOfGrabbers</i>	char array allocated by user
<i>size</i>	size of listOfGrabbers

Attention

This function must be called before [FliSdk_detectCameras_V2\(\)](#)

7.16.2.6 FliSdk_disableGrabN_V2()

```
EXPORTED bool FliSdk_disableGrabN_V2 (
    FliContext context )
```

Disable grab N mode.

Parameters

<i>context</i>	SDK context
----------------	-------------

7.16.2.7 FliSdk_display16bImage_V2()

```
EXPORTED void FliSdk_display16bImage_V2 (
    FliContext context,
    uint8_t * image,
    const char * windowName,
    bool unsignedPixels )
```

Open an Opencv window to display image.

Parameters

<i>context</i>	SDK context
<i>image</i>	pointer to the image buffer
<i>windowName</i>	name of the window
<i>unsignedPixels</i>	indicate if pixel are signed/unsigned

7.16.2.8 FliSdk_display8bImage_V2()

```
EXPORTED void FliSdk_display8bImage_V2 (
    FliContext context,
```



```
uint8_t * image,
const char * windowName )
```

Open an OpenCv window to display image.

Parameters

<i>context</i>	SDK context
<i>image</i>	pointer to the image buffer
<i>windowName</i>	name of the window

7.16.2.9 FliSdk_enableFollowUpTheRamp_V2()

```
EXPORTED void FliSdk_enableFollowUpTheRamp_V2 (
    FliContext context,
    bool enable )
```

Enable/disable the initialisation of the pixel sum for the follow up.

Parameters

<i>context</i>	SDK context
<i>enable</i>	enable/disable the follow up ramp

7.16.2.10 FliSdk_enableFowlerProcessing_V2()

```
EXPORTED void FliSdk_enableFowlerProcessing_V2 (
    FliContext context,
    bool enable )
```

Enable/disable the Fowler processing for C-RED 1 cameras.

Parameters

<i>context</i>	SDK context
<i>enable</i>	enable/disable the Fowler processing

7.16.2.11 FliSdk_enableGrabN_V2()

```
EXPORTED bool FliSdk_enableGrabN_V2 (
    FliContext context,
    uint32_t nbFrames )
```

Enable grab N mode.

Parameters

<i>context</i>	SDK context
<i>nbFrames</i>	: number of frames to grab.

7.16.2.12 FliSdk_enableRingBuffer_V2()

```
EXPORTED void FliSdk_enableRingBuffer_V2 (
    FliContext context,
    bool enable )
```

Enable or disable internal ring buffer of the SDK.

Parameters

<i>context</i>	SDK context
<i>enable</i>	will enable if true, disable if false

Attention

If ring buffer is disabled loadBuffer, saveBuffer, getRawImage, getImage and getImage16b will be disabled. grabN and error count will be disable too.

7.16.2.13 FliSdk_enableSubstractMode_V2()

```
EXPORTED void FliSdk_enableSubstractMode_V2 (
    FliContext context,
    bool enable )
```

Enable/disable the mode substract that will substract the image N by the image N-1 and save it in the buffer.

Parameters

<i>context</i>	SDK context
<i>enable</i>	enable/disable the mode

7.16.2.14 FliSdk_enableUnsignedPixel_V2()

```
EXPORTED void FliSdk_enableUnsignedPixel_V2 (
    FliContext context,
    bool enable )
```

Change the pixel sign (int16 or uint16)

Parameters

<i>context</i>	SDK context
<i>enable</i>	is unsigned if true, signed if false

7.16.2.15 FliSdk_exit_V2()

```
EXPORTED void FliSdk_exit_V2 (
    FliContext context )
```

Delete the SDK context.

Parameters

<i>context</i>	SDK context
----------------	-------------

7.16.2.16 FliSdk_forceCurrentCameraModel_V2()

```
EXPORTED void FliSdk_forceCurrentCameraModel_V2 (
    FliContext context,
    CameraModel_C model )
```

use this function when a camera is undefined

Parameters

<i>context</i>	SDK context
<i>model</i>	: model to set to the camera

7.16.2.17 FliSdk_getAvailableSaveFormats_V2()

```
EXPORTED void FliSdk_getAvailableSaveFormats_V2 (
    FliContext context,
    char * fullName,
    size_t fullNameSize,
    char * extension,
    size_t extensionSize )
```

Return a list with the formats full name and a list with the formats extension, each item is separated by a ";".

Parameters

<i>context</i>	SDK context
----------------	-------------

Parameters

<i>fullName</i>	an array of char allocated by user for the list of full name formats
<i>fullNameSize</i>	size of the previous array
<i>extension</i>	an array of char allocated by user for the list of the extensions formats
<i>extensionSize</i>	size of the previous array

7.16.2.18 FliSdk_getBufferFilling_V2()

```
EXPORTED uint32_t FliSdk_getBufferFilling_V2 (
    FliContext context )
```

Get buffer filling.

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

a number representing the filling

7.16.2.19 FliSdk_getBufferSize_V2()

```
EXPORTED uint16_t FliSdk_getBufferSize_V2 (
    FliContext context )
```

Get current buffer size.

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

the buffer size in Mo

7.16.2.20 FliSdk_getBufferWithInfo_V2()

```
EXPORTED bool FliSdk_getBufferWithInfo_V2 (
    FliContext context,
```

```

    const char * path,
    LoadBufferInfo_C * info )

```

Load a buffer from a file, allocate memory, and return that memory to the user.

User must delete this memory if not used. For a raw file, user have to set width, height and isMono8 info in the LoadBufferInfo.

Parameters

<i>context</i>	SDK context
<i>path</i>	path to the file
<i>info</i>	struct with images and info

7.16.2.21 FliSdk_getBurstFilter_V2()

```

EXPORTED int16_t FliSdk_getBurstFilter_V2 (
    FliContext context )

```

Get the current burst filter id applied.

Parameters

<i>context</i>	SDK context
----------------	-------------

7.16.2.22 FliSdk_getCameraModelAsString_V2()

```

EXPORTED void FliSdk_getCameraModelAsString_V2 (
    FliContext context,
    char * model,
    size_t size )

```

returns the current camera model as a string

Parameters

<i>context</i>	SDK context
<i>model</i>	char array allocated by user
<i>size</i>	size of model

7.16.2.23 FliSdk_getCroppingState_V2()

```

EXPORTED bool FliSdk_getCroppingState_V2 (
    FliContext context,

```

```
bool * isEnabled,
CroppingData_C * croppingData )
```

Get the cropping data from the camera.

Parameters

<i>context</i>	SDK context
<i>isEnabled</i>	a reference to a bool from the user
<i>croppingData</i>	a reference of the cropping data of user

7.16.2.24 FliSdk_getCurrentCameraModel_V2()

```
EXPORTED CameraModel_C FliSdk_getCurrentCameraModel_V2 (
    FliContext context )
```

returns the current camera model

Parameters

<i>context</i>	SDK context
----------------	-------------

7.16.2.25 FliSdk_getCurrentCameraName_V2()

```
EXPORTED bool FliSdk_getCurrentCameraName_V2 (
    FliContext context,
    char * cameraName,
    size_t size )
```

Get the name of the current camera used by the SDK.

Parameters

<i>context</i>	SDK context
<i>cameraName</i>	char array allocated by user
<i>size</i>	size of cameraName

7.16.2.26 FliSdk_getCurrentImageDimension_V2()

```
EXPORTED void FliSdk_getCurrentImageDimension_V2 (
    FliContext context,
```

```
uint16_t * width,  
uint16_t * height )
```

Get the current image dimension considering cropping.

Parameters

<i>context</i>	SDK context
<i>width</i>	reference to user variable width
<i>height</i>	reference to user variable height

7.16.2.27 FliSdk_getDetectedCameras_V2()

```
EXPORTED void FliSdk_getDetectedCameras_V2 (
    FliContext context,
    char * listOfCameras,
    size_t size )
```

Get the list with the names of detected cameras separated by ";".

Parameters

<i>context</i>	SDK context
<i>listOfCameras</i>	char array allocated by user
<i>size</i>	size of listOfCameras

7.16.2.28 FliSdk_getDetectedGrabbers_V2()

```
EXPORTED void FliSdk_getDetectedGrabbers_V2 (
    FliContext context,
    char * listOfGrabbers,
    size_t size )
```

Get the list with the names of detected grabbers separated by ";".

Parameters

<i>context</i>	SDK context
<i>listOfGrabbers</i>	char array allocated by user
<i>size</i>	size of listOfGrabbers

7.16.2.29 FliSdk_getFps_V2()

```
EXPORTED uint32_t FliSdk_getFps_V2 (
    FliContext context )
```

Get the buffer acquisition rate.

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

a number representing acquisition FPS

7.16.2.30 FliSdk_getImagesCapacity_V2()

```
EXPORTED uint32_t FliSdk_getImagesCapacity_V2 (  
    FliContext context )
```

Give the images capacity of the buffer.

Parameters

<i>context</i>	SDK context
----------------	-------------

7.16.2.31 FliSdk_getNbCountError_V2()

```
EXPORTED uint64_t FliSdk_getNbCountError_V2 (  
    FliContext context )
```

Get the number of frame count error.

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

the number of count error

7.16.2.32 FliSdk_getOcamFrameNumber_V2()

```
EXPORTED unsigned int FliSdk_getOcamFrameNumber_V2 (  
    FliContext context,  
    int64_t index )
```

Get the frame number of ocam image at index.

Parameters

<i>context</i>	SDK context
<i>index</i>	index of the image in the buffer

Returns

the frame number of the image at index

7.16.2.33 FliSdk_getProcessedImage16b_lv_V2()

```
EXPORTED void FliSdk_getProcessedImage16b_lv_V2 (
    FliContext context,
    int64_t index,
    uint16_t * image )
```

Get the RGB processed image at the given index, if -1 then the last image is processed The buffer is overwritten only when the function is recalled.

Parameters

<i>context</i>	SDK context
<i>index</i>	: index of the image in the buffer
<i>image</i>	a pointer to the processed image

7.16.2.34 FliSdk_getProcessedImage16b_V2()

```
EXPORTED uint8_t* FliSdk_getProcessedImage16b_V2 (
    FliContext context,
    int64_t index )
```

Get the 16bits grayscale processed image at the given index, if -1 then the last image is processed The buffer is overwritten only when the function is recalled.

Parameters

<i>context</i>	SDK context
<i>index</i>	: index of the image in the buffer

Returns

a pointer to the processed image

7.16.2.35 FliSdk_getProcessedImage_lv_V2()

```
EXPORTED void FliSdk_getProcessedImage_lv_V2 (
    FliContext context,
    int64_t index,
    uint32_t * image )
```

Get the RGB processed image at the given index, if -1 then the last image is processed The buffer is overwritten only when the function is recalled.

Parameters

<i>context</i>	SDK context
<i>index</i>	: index of the image in the buffer
<i>image</i>	a pointer to the processed image

7.16.2.36 FliSdk_getProcessedImage_V2()

```
EXPORTED uint8_t* FliSdk_getProcessedImage_V2 (
    FliContext context,
    int64_t index )
```

Get the RGB processed image at the given index, if -1 then the last image is processed The buffer is overwritten only when the function is recalled.

Parameters

<i>context</i>	SDK context
<i>index</i>	: index of the image in the buffer

Returns

a pointer to the processed image

7.16.2.37 FliSdk_getRawImage_lv_V2()

```
EXPORTED void FliSdk_getRawImage_lv_V2 (
    FliContext context,
    int64_t index,
    uint16_t * image )
```

Get the image at index or the last image if index is -1, without processing.

Parameters

<i>context</i>	SDK context
<i>index</i>	: index of the image in the buffer.
<i>image</i>	a pointer to the image array if index is valid else nullptr

7.16.2.38 FliSdk_getRawImage_V2()

```
const EXPORTED unsigned char* FliSdk_getRawImage_V2 (
    FliContext context,
    int64_t index )
```

Get the image at index or the last image if index is -1, without processing.

Parameters

<i>context</i>	SDK context
<i>index</i>	: index of the image in the buffer.

Returns

pointer to the image array if index is valid else nullptr

7.16.2.39 FliSdk_init_V2()

```
EXPORTED FliContext FliSdk_init_V2 ( )
```

Create a SDK context.

Returns

FliContext: SDK context

7.16.2.40 FliSdk_initLog_V2()

```
EXPORTED void FliSdk_initLog_V2 (
    FliContext context,
    const char * appName )
```

init SDK logging

Parameters

<i>context</i>	SDK context
<i>appName</i>	: appName will be used for the name of the file

7.16.2.41 FliSdk_isCblueOne_V2()

```
EXPORTED bool FliSdk_isCblueOne_V2 (  
    FliContext context )
```

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if the current camera is a C-BLUE One else false

7.16.2.42 FliSdk_isCblueSfnc_V2()

```
EXPORTED bool FliSdk_isCblueSfnc_V2 (  
    FliContext context )
```

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if the current camera is a C-BLUE camera else false

7.16.2.43 FliSdk_isCblueTwo_V2()

```
EXPORTED bool FliSdk_isCblueTwo_V2 (  
    FliContext context )
```

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if the current camera is a C-BLUE One else false

7.16.2.44 FliSdk_isCred_V2()

```
EXPORTED bool FliSdk_isCred_V2 (  
    FliContext context )
```

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if the current camera is a C-RED camera else false

7.16.2.45 FliSdk_isCredOne_V2()

```
EXPORTED bool FliSdk_isCredOne_V2 (  
    FliContext context )
```

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if the current camera is a C-RED One else false

7.16.2.46 FliSdk_isCredThree_V2()

```
EXPORTED bool FliSdk_isCredThree_V2 (  
    FliContext context )
```

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if the current camera is a C-RED 3 else false

7.16.2.47 FliSdk_isCredTwo_V2()

```
EXPORTED bool FliSdk_isCredTwo_V2 (  
    FliContext context )
```

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if the current camera is a C-RED 2 else false

7.16.2.48 FliSdk_isCredTwoLite_V2()

```
EXPORTED bool FliSdk_isCredTwoLite_V2 (
    FliContext context )
```

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if the current camera is a C-RED 2 else false

7.16.2.49 FliSdk_isCroppingDataValid_V2()

```
EXPORTED bool FliSdk_isCroppingDataValid_V2 (
    FliContext context,
    CroppingData_C croppingData )
```

Check if the cropping data is valid for Cred2 & Cred3.

Parameters

<i>context</i>	SDK context
<i>croppingData</i>	the cropping data of user

Returns

true if valid else false

7.16.2.50 FliSdk_isGrabNEnabled_V2()

```
EXPORTED bool FliSdk_isGrabNEnabled_V2 (
    FliContext context )
```


State of the grab N.

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if grab N mode activated else false

7.16.2.51 FliSdk_isGrabNFinished_V2()

```
EXPORTED bool FliSdk_isGrabNFinished_V2 (  
    FliContext context )
```

State of the grab N.

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if the grab is over else false

7.16.2.52 FliSdk_isMono8Pixel_V2()

```
EXPORTED bool FliSdk_isMono8Pixel_V2 (  
    FliContext context )
```

Return the pixel size (1 byte if true, 2 bytes if false)

Parameters

<i>context</i>	SDK context
----------------	-------------

7.16.2.53 FliSdk_isOcam2k_V2()

```
EXPORTED bool FliSdk_isOcam2k_V2 (  
    FliContext context )
```

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if the current camera is a Ocam2K else false

7.16.2.54 FliSdk_isOcam2s_V2()

```
EXPORTED bool FliSdk_isOcam2s_V2 (  
    FliContext context )
```

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if the current camera is a Ocam2S else false

7.16.2.55 FliSdk_isSerialCamera_V2()

```
EXPORTED bool FliSdk_isSerialCamera_V2 (  
    FliContext context )
```

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if the current camera is a serial camera (C-RED or Ocam2) else false

7.16.2.56 FliSdk_isStarted_V2()

```
EXPORTED bool FliSdk_isStarted_V2 (  
    FliContext context )
```

Get the state of the grabber (started or stopped)

Parameters

<i>context</i>	SDK context
----------------	-------------

Returns

true if grabber is started else false

7.16.2.57 FliSdk_isUnsignedPixel_V2()

```
EXPORTED bool FliSdk_isUnsignedPixel_V2 (
    FliContext context )
```

Return the pixel sign (int16 or uint16)

Parameters

<i>context</i>	SDK context
----------------	-------------

7.16.2.58 FliSdk_loadBufferFromFile_V2()

```
EXPORTED bool FliSdk_loadBufferFromFile_V2 (
    FliContext context,
    const char * path,
    CroppingData_C * bufferCrop )
```

Load a buffer from a file, in the ringBuffer of the SDK.

Parameters

<i>context</i>	SDK context
<i>path</i>	path to the file
<i>bufferCrop</i>	a ref to CroppingData to get current cropping

7.16.2.59 FliSdk_loadBufferRaw_V2()

```
EXPORTED bool FliSdk_loadBufferRaw_V2 (
    FliContext context,
    const uint8_t * buffer,
    uint32_t nbImages )
```

Load a buffer in the ringBuffer of the SDK.

Parameters

<i>context</i>	SDK context
<i>buffer</i>	data buffer
<i>nblimages</i>	nblimages in the buffer

7.16.2.60 FliSdk_removeCallbackNewImage_V2()

```
EXPORTED void FliSdk_removeCallbackNewImage_V2 (
    FliContext context,
    callbackHandler ctx )
```

Remove callback.

Parameters

<i>context</i>	SDK context
<i>ctx</i>	callback handler

7.16.2.61 FliSdk_removeImageProcessing_V2()

```
EXPORTED void FliSdk_removeImageProcessing_V2 (
    FliContext context,
    ProcessingId id )
```

remove an image processing

Parameters

<i>context</i>	SDK context
<i>id</i>	id of the imageprocessing to remove

7.16.2.62 FliSdk_resetBuffer_V2()

```
EXPORTED void FliSdk_resetBuffer_V2 (
    FliContext context )
```

Reset the buffer.

Parameters

<i>context</i>	SDK context
----------------	-------------

7.16.2.63 FliSdk_saveBuffer_V2()

```
EXPORTED bool FliSdk_saveBuffer_V2 (
    FliContext context,
    const char * path,
    uint32_t startIndex,
    uint32_t endIndex )
```

Save the buffer at path.

Parameters

<i>context</i>	SDK context
<i>path</i>	: path of the file
<i>startIndex</i>	: start index of the buffer
<i>endIndex</i>	: end index of the buffer

7.16.2.64 FliSdk_saveBufferWithOptions_V2()

```
EXPORTED bool FliSdk_saveBufferWithOptions_V2 (
    FliContext context,
    const char * path,
    uint32_t startIndex,
    uint32_t endIndex,
    saveBufferProgressionCallback func,
    bool withMetadata,
    uint16_t offset,
    bool forceUnsigned,
    uint16_t decimation )
```

Save the buffer at path.

Parameters

<i>context</i>	SDK context
<i>path</i>	: path of the file
<i>startIndex</i>	: start index of the buffer
<i>endIndex</i>	: end index of the buffer
<i>func</i>	: a callback to notify the progression of the save, return false to stop the save
<i>withMetadata</i>	: true to include camera conf in metadata
<i>offset</i>	: apply an offset on all the pixels of all images
<i>forceUnsigned</i>	: force the save with unsigned pixels type
<i>decimation</i>	: apply a decimation on the index of saved images

7.16.2.65 FliSdk_setBufferSize_V2()

```
EXPORTED void FliSdk_setBufferSize_V2 (
    FliContext context,
    uint16_t sizeMo )
```

Change the buffer capacity in Mo.

Parameters

<i>context</i>	SDK context
<i>sizeMo</i>	: capacity of the ring buffer in Mo

7.16.2.66 FliSdk_setBufferSizeInImages_V2()

```
EXPORTED void FliSdk_setBufferSizeInImages_V2 (
    FliContext context,
    uint64_t nbImages )
```

Change the buffer capacity in number of images.

Parameters

<i>context</i>	SDK context
<i>nbImages</i>	: capacity of the ring buffer in nb images

7.16.2.67 FliSdk_setBurstFilter_V2()

```
EXPORTED void FliSdk_setBurstFilter_V2 (
    FliContext context,
    int16_t id )
```

Set the burst filter for id.

Parameters

<i>context</i>	SDK context
<i>id</i>	: id to display

7.16.2.68 FliSdk_setCamera_V2()

```
EXPORTED bool FliSdk_setCamera_V2 (
```

```
FliContext context,
const char * cameraName )
```

Set the camera to be used.

Parameters

<i>context</i>	SDK context
<i>cameraName</i>	name of the camera

Returns

true if camera exists else false

Attention

Call update() to apply.

7.16.2.69 FliSdk_setCroppingState_V2()

```
EXPORTED bool FliSdk_setCroppingState_V2 (
    FliContext context,
    bool enable,
    CroppingData_C croppingData )
```

Set the cropping data.

Parameters

<i>context</i>	SDK context
<i>enable</i>	enable or disable cropping
<i>croppingData</i>	cropping data

7.16.2.70 FliSdk_setFowlerOffset_V2()

```
EXPORTED void FliSdk_setFowlerOffset_V2 (
    FliContext context,
    int16_t offset )
```

Set the value of the Fowler offset to apply on sum of the images.

Parameters

<i>context</i>	SDK context
<i>offset</i>	the value of the offset (between 0 and 65535), default is 0

7.16.2.71 FliSdk_setFpsTrigger_V2()

```
EXPORTED void FliSdk_setFpsTrigger_V2 (
    FliContext context,
    callbackHandler ctx,
    uint16_t fps )
```

Add a callback in order to receive frames.

Parameters

<i>context</i>	SDK context
<i>ctx</i>	callback handler
<i>fps</i>	at which fps does the callback want to receive images, if 0 then full speed

7.16.2.72 FliSdk_setGrabber_V2()

```
EXPORTED bool FliSdk_setGrabber_V2 (
    FliContext context,
    const char * grabberName )
```

Set the grabber to be used.

Parameters

<i>context</i>	SDK context
<i>grabberName</i>	name of the grabber

Returns

true if grabber exists else false

Attention

Call update() to apply.

7.16.2.73 FliSdk_setImageDimension_V2()

```
EXPORTED void FliSdk_setImageDimension_V2 (
    FliContext context,
    uint16_t width,
    uint16_t height )
```

Force the image dimension apply to the grabber.

Parameters

<i>context</i>	SDK context
<i>width</i>	width of image
<i>height</i>	height of image

7.16.2.74 FliSdk_setMode_V2()

```
EXPORTED void FliSdk_setMode_V2 (
    FliContext context,
    Mode mode )
```

Set the mode of use of the SDK.

Parameters

<i>context</i>	SDK context
<i>mode</i>	mode used (full, grabOnly, configOnly)

Attention

Call update() to apply.

7.16.2.75 FliSdk_setNbImagesPerBuffer_V2()

```
EXPORTED void FliSdk_setNbImagesPerBuffer_V2 (
    FliContext context,
    uint8_t nbImages )
```

Set set number of images the grabber should acquire before trigger, use this function for high FPS.

Parameters

<i>context</i>	SDK context
<i>nbImages</i>	: number of images.

7.16.2.76 FliSdk_setOcamFrameNumberOffset_V2()

```
EXPORTED void FliSdk_setOcamFrameNumberOffset_V2 (
    FliContext context,
    uint8_t offset )
```

Set the offset for the frame number.

Parameters

<i>context</i>	SDK context
<i>offset</i>	0 for simulator, 8 for camera

7.16.2.77 FliSdk_start_V2()

```
EXPORTED bool FliSdk_start_V2 (  
    FliContext context )
```

Start the grabber (must be initialized before)

Parameters

<i>context</i>	SDK context
----------------	-------------

7.16.2.78 FliSdk_stop_V2()

```
EXPORTED bool FliSdk_stop_V2 (  
    FliContext context )
```

Stop the grabber.

Parameters

<i>context</i>	SDK context
----------------	-------------

7.16.2.79 FliSdk_update_V2()

```
EXPORTED bool FliSdk_update_V2 (  
    FliContext context )
```

Update the changes, must be call after setCamera, setGrabber or setMode to take effects.

Parameters

<i>context</i>	SDK context
----------------	-------------

7.17 FliSerialCamera.h File Reference

Classes

- class [FliSerialCamera](#)

This class is the base class of all serial camera (CameraLink). It implements common C-RED and Ocam functions.

7.18 FliSfncCamera.h File Reference

Classes

- class [FliSfncCamera](#)

This class defined all the register of an SFNC compliant camera.

7.19 IFliSdkObserver.h File Reference

Classes

- class [IFliSdkObserver](#)

This interface defines an observer to observe some SDK states.

7.20 ImageProcessing.h File Reference

Classes

- class [ImageProcessing](#)

7.21 ImageProcessing.h File Reference

Classes

- class [ImageProcessing](#)

This class manages all the processing of the images such as : statistics (mean stddev), clipping, flipping, sharpen etc...

7.22 ImageRingBuffer.h File Reference

Classes

- class [ImageRingBuffer](#)

This class derive from pure virtual [FliRingBuffer](#) and manages the images inside a ring buffer with some put methods to add one or several images or nro or iota buffers into the ring buffer.

Index

- [_cameraModel](#)
 - [FliGenicamCamera, 147](#)
 - [FliSerialCamera, 208](#)
 - [_conf](#)
 - [FliOcam2K, 153](#)
 - [_croppingFromFunction](#)
 - [FliSerialCamera, 208](#)
 - [_customSerial](#)
 - [FliSerialCamera, 208](#)
 - [_grabber](#)
 - [FliGenicamCamera, 147](#)
 - [FliSerialCamera, 208](#)
 - [_needEcho](#)
 - [FliSerialCamera, 208](#)
 - [_observers](#)
 - [FliSerialCamera, 209](#)
 - [_stringToFeature](#)
 - [FliGenicamCamera, 147](#)
 - [~FliGenicamCamera](#)
 - [FliGenicamCamera, 136](#)
 - [~FliSdk](#)
 - [FliSdk, 169](#)
 - [~FliSerialCamera](#)
 - [FliSerialCamera, 202](#)
 - [~FliSfncCamera](#)
 - [FliSfncCamera, 238](#)
 - [~ImageProcessing](#)
 - [ImageProcessing, 376](#)
 - [~ImageRingBuffer](#)
 - [ImageRingBuffer, 398](#)
 - [abortBuildNuc](#)
 - [FliCredThree, 77](#)
 - [FliCredTwo, 105](#)
 - [AcquisitionAbort](#)
 - [FliSfncCamera, 238](#)
 - [AcquisitionArm](#)
 - [FliSfncCamera, 238](#)
 - [AcquisitionBurstFrameCount](#)
 - [FliSfncCamera, 239](#)
 - [AcquisitionFrameCount](#)
 - [FliSfncCamera, 239](#)
 - [AcquisitionFrameRate](#)
 - [FliSfncCamera, 239](#)
 - [AcquisitionFrameRateEnable](#)
 - [FliSfncCamera, 239](#)
 - [AcquisitionFrameRateMaxReg](#)
 - [FliCblueOne, 36](#)
 - [AcquisitionFrameRateMinReg](#)
 - [FliCblueOne, 36](#)
 - [AcquisitionLineRate](#)
 - [FliSfncCamera, 239](#)
 - [AcquisitionLineRateEnable](#)
 - [FliSfncCamera, 239](#)
 - [AcquisitionMode](#)
 - [FliSfncCamera, 240](#)
 - [AcquisitionModeEnum](#)
 - [FliCblueSfncEnum, 19](#)
 - [AcquisitionModeString](#)
 - [FliCblueSfncEnum, 24](#)
 - [AcquisitionStart](#)
 - [FliSfncCamera, 240](#)
 - [AcquisitionStatus](#)
 - [FliSfncCamera, 240](#)
 - [AcquisitionStatusSelector](#)
 - [FliSfncCamera, 240](#)
 - [AcquisitionStop](#)
 - [FliSfncCamera, 240](#)
 - [AcquisitionStopMode](#)
 - [FliSfncCamera, 240](#)
 - [ActionDeviceKey](#)
 - [FliSfncCamera, 241](#)
 - [ActionGroupKey](#)
 - [FliSfncCamera, 241](#)
 - [ActionGroupMask](#)
 - [FliSfncCamera, 241](#)
 - [ActionQueueSize](#)
 - [FliSfncCamera, 241](#)
 - [ActionSelector](#)
 - [FliSfncCamera, 241](#)
 - [ActionUnconditionalMode](#)
 - [FliSfncCamera, 241](#)
 - [Active](#)
 - [FliCblueSfncEnum, 22](#)
 - [addCallbackAllRegisters](#)
 - [FliGenicamCamera, 137](#)
 - [addCallbackDimensionsRegisters](#)
 - [FliGenicamCamera, 137](#)
 - [addEthernetCamera](#)
 - [FliSdk, 170](#)
 - [addFakeEthernetCamera](#)
 - [FliSdk, 170](#)
 - [addImageProcessing](#)
 - [FliSdk, 171](#)
 - [addObserver](#)
 - [FliSdk, 171](#)
 - [FliSerialCamera, 203](#)
 - [addRawImageReceivedObserver](#)
 - [FliSdk, 171](#)

- aduToDegrees
 - ImageProcessing, [376](#), [377](#)
- AgcParam
 - FliCredThree, [77](#)
- alarm
 - FliOcam2K.h, [421](#)
- All
 - FliCblueSfncEnum, [20](#)
- AnalogAll
 - FliCblueSfncEnum, [22](#)
- aPAUSEMACCtrlFramesReceived
 - FliSfncCamera, [242](#)
- aPAUSEMACCtrlFramesTransmitted
 - FliSfncCamera, [242](#)
- Areascan
 - FliCblueSfncEnum, [22](#)
- autoClipEnabled
 - ImageProcessing, [355](#)
 - ImageProcessing, [377](#)
- AUTOMATIC
 - FliCredTwoLite, [132](#)
- Automatic
 - FliCblueOneEnum, [10](#), [12](#)
- Average
 - FliCblueTwoEnum, [29](#)
- BadPixelsAlgo
 - ImageProcessing, [354](#)
- badPixelsCartoLoaded
 - ImageProcessing, [355](#)
 - ImageProcessing, [377](#)
- BalanceRatio
 - FliSfncCamera, [242](#)
- BalanceRatioSelector
 - FliSfncCamera, [242](#)
- BalanceWhiteAuto
 - FliSfncCamera, [242](#)
- BinningHorizontal
 - FliCblueTwo, [45](#)
 - FliSfncCamera, [242](#)
- BinningHorizontalMode
 - FliCblueTwo, [45](#)
 - FliSfncCamera, [243](#)
- BinningHorizontalModeEnum
 - FliCblueTwoEnum, [29](#)
- BinningHorizontalModeStringToValue
 - FliCblueTwoEnum, [30](#)
- binningOffset
 - Ocam2Conf, [413](#)
- BinningSelector
 - FliSfncCamera, [243](#)
- BinningSelectorEnum
 - FliCblueTwoEnum, [29](#)
- BinningSelectorStringToValue
 - FliCblueTwoEnum, [30](#)
- BinningVertical
 - FliCblueTwo, [45](#)
 - FliSfncCamera, [243](#)
- BinningVerticalMode
 - FliCblueTwo, [46](#)
 - FliSfncCamera, [243](#)
- BinningVerticalModeEnum
 - FliCblueTwoEnum, [29](#)
- BinningVerticalModeStringToValue
 - FliCblueTwoEnum, [30](#)
- Black
 - FliCblueOneEnum, [13](#)
- BlackLevel
 - FliSfncCamera, [243](#)
- BlackLevelAuto
 - FliSfncCamera, [243](#)
- BlackLevelAutoBalance
 - FliSfncCamera, [244](#)
- BlackLevelAutoEnum
 - FliCblueSfncEnum, [19](#)
- BlackLevelAutoString
 - FliCblueSfncEnum, [24](#)
- BlackLevelSelector
 - FliSfncCamera, [244](#)
- BlackLevelSelectorEnum
 - FliCblueSfncEnum, [19](#)
- BlackLevelSelectorString
 - FliCblueSfncEnum, [24](#)
- buildBias
 - FliCred, [49](#)
- buildBiasNuc
 - FliCredThree, [77](#)
 - FliCredTwo, [105](#)
- buildFlat
 - FliCred, [49](#)
- buildFlatHdrC1
 - FliCredThree, [78](#)
 - FliCredTwo, [106](#)
- buildFlatHdrC1Nuc
 - FliCredThree, [78](#)
 - FliCredTwo, [106](#)
- buildFlatHdrC2
 - FliCredThree, [78](#)
 - FliCredTwo, [106](#)
- buildFlatHdrC2Nuc
 - FliCredThree, [78](#)
 - FliCredTwo, [106](#)
- buildFlatNuc
 - FliCredThree, [78](#)
 - FliCredTwo, [106](#)
- callbackHandler
 - FliSdk_C_V2.h, [425](#)
- Camera
 - ImageProcessing, [354](#)
- camera
 - FliSdk, [171](#)
- CameraPresence
 - FliSfncCamera, [244](#)
- Case
 - FliCblueOneEnum, [11](#)
- cblueOne
 - FliSdk, [172](#)

- cblueSfnc
 - FliSdk, [172](#)
- cblueTwo
 - FliSdk, [172](#)
- CELSIUS
 - ImageProcessing, [354](#)
- ChunkBinningHorizontal
 - FliSfncCamera, [244](#)
- ChunkBinningVertical
 - FliSfncCamera, [244](#)
- ChunkBlackLevel
 - FliSfncCamera, [244](#)
- ChunkBlackLevelSelector
 - FliSfncCamera, [245](#)
- ChunkComponentID
 - FliSfncCamera, [245](#)
- ChunkComponentIDValue
 - FliSfncCamera, [245](#)
- ChunkComponentSelector
 - FliSfncCamera, [245](#)
- ChunkCounterSelector
 - FliSfncCamera, [245](#)
- ChunkCounterValue
 - FliSfncCamera, [245](#)
- ChunkDecimationHorizontal
 - FliSfncCamera, [246](#)
- ChunkDecimationVertical
 - FliSfncCamera, [246](#)
- ChunkEnable
 - FliSfncCamera, [246](#)
- ChunkEncoderSelector
 - FliSfncCamera, [246](#)
- ChunkEncoderStatus
 - FliSfncCamera, [246](#)
- ChunkEncoderValue
 - FliSfncCamera, [246](#)
- ChunkExposureTime
 - FliSfncCamera, [247](#)
- ChunkExposureTimeSelector
 - FliSfncCamera, [247](#)
- ChunkFrameID
 - FliSfncCamera, [247](#)
- ChunkGain
 - FliSfncCamera, [247](#)
- ChunkGainSelector
 - FliSfncCamera, [247](#)
- ChunkGroupID
 - FliSfncCamera, [247](#)
- ChunkGroupIDValue
 - FliSfncCamera, [248](#)
- ChunkGroupSelector
 - FliSfncCamera, [248](#)
- ChunkHeight
 - FliSfncCamera, [248](#)
- ChunkLinePitch
 - FliSfncCamera, [248](#)
- ChunkLineStatusAll
 - FliSfncCamera, [248](#)
- ChunkModeActive
 - FliSfncCamera, [248](#)
- ChunkOffsetX
 - FliSfncCamera, [249](#)
- ChunkOffsetY
 - FliSfncCamera, [249](#)
- ChunkPixelDynamicRangeMax
 - FliSfncCamera, [249](#)
- ChunkPixelDynamicRangeMin
 - FliSfncCamera, [249](#)
- ChunkPixelFormat
 - FliSfncCamera, [249](#)
- ChunkRegionID
 - FliSfncCamera, [249](#)
- ChunkRegionIDValue
 - FliSfncCamera, [250](#)
- ChunkRegionSelector
 - FliSfncCamera, [250](#)
- ChunkReverseX
 - FliSfncCamera, [250](#)
- ChunkReverseY
 - FliSfncCamera, [250](#)
- ChunkScan3dAxisMax
 - FliSfncCamera, [250](#)
- ChunkScan3dAxisMin
 - FliSfncCamera, [250](#)
- ChunkScan3dBaseline
 - FliSfncCamera, [251](#)
- ChunkScan3dCoordinateOffset
 - FliSfncCamera, [251](#)
- ChunkScan3dCoordinateReferenceSelector
 - FliSfncCamera, [251](#)
- ChunkScan3dCoordinateReferenceValue
 - FliSfncCamera, [251](#)
- ChunkScan3dCoordinateScale
 - FliSfncCamera, [251](#)
- ChunkScan3dCoordinateSelector
 - FliSfncCamera, [251](#)
- ChunkScan3dCoordinateSystem
 - FliSfncCamera, [252](#)
- ChunkScan3dCoordinateSystemReference
 - FliSfncCamera, [252](#)
- ChunkScan3dCoordinateTransformSelector
 - FliSfncCamera, [252](#)
- ChunkScan3dDistanceUnit
 - FliSfncCamera, [252](#)
- ChunkScan3dFocalLength
 - FliSfncCamera, [252](#)
- ChunkScan3dInvalidDataFlag
 - FliSfncCamera, [252](#)
- ChunkScan3dInvalidDataValue
 - FliSfncCamera, [253](#)
- ChunkScan3dOutputMode
 - FliSfncCamera, [253](#)
- ChunkScan3dPrincipalPointU
 - FliSfncCamera, [253](#)
- ChunkScan3dPrincipalPointV
 - FliSfncCamera, [253](#)

- ChunkScan3dTransformValue
 - FliSfncCamera, [253](#)
- ChunkScanLineSelector
 - FliSfncCamera, [253](#)
- ChunkSelector
 - FliSfncCamera, [254](#)
- ChunkSequencerSetActive
 - FliSfncCamera, [254](#)
- ChunkSourceID
 - FliSfncCamera, [254](#)
- ChunkSourceIDValue
 - FliSfncCamera, [254](#)
- ChunkSourceSelector
 - FliSfncCamera, [254](#)
- ChunkStreamChannelID
 - FliSfncCamera, [254](#)
- ChunkTimerSelector
 - FliSfncCamera, [255](#)
- ChunkTimerValue
 - FliSfncCamera, [255](#)
- ChunkTimestamp
 - FliSfncCamera, [255](#)
- ChunkTimestampLatchValue
 - FliSfncCamera, [255](#)
- ChunkTransferBlockID
 - FliSfncCamera, [255](#)
- ChunkTransferQueueCurrentBlockCount
 - FliSfncCamera, [255](#)
- ChunkTransferStreamID
 - FliSfncCamera, [256](#)
- ChunkWidth
 - FliSfncCamera, [256](#)
- ChunkXMLEnable
 - FliSfncCamera, [256](#)
- CIConfiguration
 - FliSfncCamera, [256](#)
- clip
 - ImageProcessing, [355](#)
 - ImageProcessing, [377](#)
- ClippingType
 - ImageProcessing, [354](#)
- CITimeSlotsCount
 - FliSfncCamera, [256](#)
- ColorTransformationEnable
 - FliSfncCamera, [256](#)
- ColorTransformationSelector
 - FliSfncCamera, [257](#)
- ColorTransformationValue
 - FliSfncCamera, [257](#)
- ColorTransformationValueSelector
 - FliSfncCamera, [257](#)
- ComponentEnable
 - FliSfncCamera, [257](#)
- ComponentIDValue
 - FliSfncCamera, [257](#)
- ComponentSelector
 - FliSfncCamera, [257](#)
- configFile
 - Ocam2Conf, [413](#)
- ConfigOnly
 - FliSdk, [169](#)
- ConnectionLockLoss
 - FliCblueSfncEnum, [20](#)
- continueStarting
 - FliCred, [49](#)
- Continuous
 - FliCblueSfncEnum, [19](#)
- ControlPacketCrc
 - FliCblueSfncEnum, [20](#)
- ConversionEfficiency
 - FliCblueOne, [36](#)
- ConversionEfficiencyEnum
 - FliCblueOneEnum, [10](#)
- ConversionEfficiencyStringToValue
 - FliCblueOneEnum, [14](#)
- CoolingMode
 - FliCredTwoLite, [132](#)
- CounterActive
 - FliCblueSfncEnum, [20](#)
- CounterDuration
 - FliSfncCamera, [258](#)
- CounterEventActivation
 - FliSfncCamera, [258](#)
- CounterEventSource
 - FliSfncCamera, [258](#)
- CounterOverflow
 - FliCblueSfncEnum, [20](#)
- CounterReset
 - FliSfncCamera, [258](#)
- CounterResetActivation
 - FliSfncCamera, [258](#)
- CounterResetSource
 - FliSfncCamera, [258](#)
- CounterSelector
 - FliSfncCamera, [259](#)
- CounterStatus
 - FliSfncCamera, [259](#)
- CounterTriggerActivation
 - FliSfncCamera, [259](#)
- CounterTriggerSource
 - FliSfncCamera, [259](#)
- CounterValue
 - FliSfncCamera, [259](#)
- CounterValueAtReset
 - FliSfncCamera, [259](#)
- CPU
 - FliCblueOneEnum, [11](#)
- cred
 - FliSdk, [172](#)
- credOne
 - FliSdk, [173](#)
- credThree
 - FliSdk, [173](#)
- credTwo
 - FliSdk, [173](#)
- credTwoLite

- FliSdk, [173](#)
- CurrentIPAddress
 - FliCblueOne, [36](#)
- CurrentSubnetMask
 - FliCblueOne, [36](#)
- CUSTOM_STEPS
 - FliCredTwoLite, [132](#)
- CXP10_X2
 - FliCblueSfncEnum, [21](#)
- CXP12_X1
 - FliCblueSfncEnum, [21](#)
- CXP12_X2
 - FliCblueSfncEnum, [21](#)
- CXP1_X1
 - FliCblueSfncEnum, [21](#)
- CXP1_X2
 - FliCblueSfncEnum, [21](#)
- CXP6_X2
 - FliCblueSfncEnum, [21](#)
- CxpConnectionSelector
 - FliSfncCamera, [260](#)
- CxpConnectionTestErrorCount
 - FliSfncCamera, [260](#)
- CxpConnectionTestMode
 - FliSfncCamera, [260](#)
- CxpConnectionTestModeEnum
 - FliCblueSfncEnum, [20](#)
- CxpConnectionTestModeString
 - FliCblueSfncEnum, [24](#)
- CxpConnectionTestPacketCount
 - FliSfncCamera, [260](#)
- CxpErrorCounterReset
 - FliSfncCamera, [260](#)
- CxpErrorCounterSelector
 - FliSfncCamera, [260](#)
- CxpErrorCounterSelectorEnum
 - FliCblueSfncEnum, [20](#)
- CxpErrorCounterSelectorString
 - FliCblueSfncEnum, [25](#)
- CxpErrorCounterStatus
 - FliSfncCamera, [261](#)
- CxpErrorCounterStatusEnum
 - FliCblueSfncEnum, [20](#)
- CxpErrorCounterStatusString
 - FliCblueSfncEnum, [25](#)
- CxpErrorCounterValue
 - FliSfncCamera, [261](#)
- CxpFirstLineTriggerWithFrameStart
 - FliSfncCamera, [261](#)
- CxpLinkConfiguration
 - FliSfncCamera, [261](#)
- CxpLinkConfigurationEnum
 - FliCblueSfncEnum, [21](#)
- CxpLinkConfigurationPreferred
 - FliSfncCamera, [261](#)
- CxpLinkConfigurationPreferredEnum
 - FliCblueSfncEnum, [21](#)
- CxpLinkConfigurationPreferredString
 - FliCblueSfncEnum, [25](#)
- CxpLinkConfigurationStatus
 - FliSfncCamera, [261](#)
- CxpLinkConfigurationStatusEnum
 - FliCblueSfncEnum, [21](#)
- CxpLinkConfigurationStatusString
 - FliCblueSfncEnum, [25](#)
- CxpLinkConfigurationString
 - FliCblueSfncEnum, [26](#)
- CxpLinkSharingDuplicateStripe
 - FliSfncCamera, [262](#)
- CxpLinkSharingEnable
 - FliSfncCamera, [262](#)
- CxpLinkSharingHorizontalOverlap
 - FliSfncCamera, [262](#)
- CxpLinkSharingHorizontalStripeCount
 - FliSfncCamera, [262](#)
- CxpLinkSharingStatus
 - FliSfncCamera, [262](#)
- CxpLinkSharingSubDeviceSelector
 - FliSfncCamera, [262](#)
- CxpLinkSharingSubDeviceType
 - FliSfncCamera, [263](#)
- CxpLinkSharingVerticalOverlap
 - FliSfncCamera, [263](#)
- CxpLinkSharingVerticalStripeCount
 - FliSfncCamera, [263](#)
- CxpPoCxpAuto
 - FliSfncCamera, [263](#)
- CxpPoCxpStatus
 - FliSfncCamera, [263](#)
- CxpPoCxpTripReset
 - FliSfncCamera, [263](#)
- CxpPoCxpTurnOff
 - FliSfncCamera, [264](#)
- CxpSendReceiveSelector
 - FliSfncCamera, [264](#)
- CxpSendReceiveSelectorEnum
 - FliCblueSfncEnum, [21](#)
- CxpSendReceiveSelectorString
 - FliCblueSfncEnum, [26](#)
- DecimationHorizontal
 - FliSfncCamera, [264](#)
- DecimationHorizontalMode
 - FliSfncCamera, [264](#)
- DecimationVertical
 - FliSfncCamera, [264](#)
- DecimationVerticalMode
 - FliSfncCamera, [264](#)
- Default12bits
 - FliCblueOneEnum, [14](#)
- Default8bits
 - FliCblueOneEnum, [13, 14](#)
- defineGrabOnlySlowMode
 - FliSdk, [174](#)
- Deinterlacing
 - FliSfncCamera, [265](#)
- deleteLicense

- FliCredThree, [78](#)
- FliCredTwo, [106](#)
- detectCameras
 - FliSdk, [175](#)
- detectEthernetCameras
 - FliSdk, [175](#)
- detectGrabbers
 - FliSdk, [175](#)
- detectOneCamera
 - FliSdk, [176](#)
- DeviceCharacterSet
 - FliSfncCamera, [265](#)
- DeviceClockFrequency
 - FliSfncCamera, [265](#)
- DeviceClockSelector
 - FliSfncCamera, [265](#)
- DeviceConnectionSelector
 - FliSfncCamera, [265](#)
- DeviceConnectionSpeed
 - FliSfncCamera, [265](#)
- DeviceConnectionStatus
 - FliSfncCamera, [266](#)
- DeviceCoolingEnable
 - FliCblueOne, [36](#)
- DeviceCoolingSetpoint
 - FliCblueOne, [37](#)
- DeviceEventChannelCount
 - FliSfncCamera, [266](#)
- DeviceFamilyName
 - FliSfncCamera, [266](#)
- DeviceFanMode
 - FliCblueOne, [37](#)
- DeviceFanModeEnum
 - FliCblueOneEnum, [10](#)
- DeviceFanModeStringToValue
 - FliCblueOneEnum, [15](#)
- DeviceFanSpeed
 - FliCblueOne, [37](#)
- DeviceFeaturePersistenceEnd
 - FliSfncCamera, [266](#)
- DeviceFeaturePersistenceStart
 - FliSfncCamera, [266](#)
- DeviceFirmwareVersion
 - FliSfncCamera, [266](#)
- DeviceGenCPVersionMajor
 - FliSfncCamera, [267](#)
- DeviceGenCPVersionMinor
 - FliSfncCamera, [267](#)
- DeviceIndicatorMode
 - FliSfncCamera, [267](#)
- DeviceIndicatorModeEnum
 - FliCblueSfncEnum, [22](#)
- DeviceIndicatorModeString
 - FliCblueSfncEnum, [26](#)
- DeviceLinkCommandTimeout
 - FliSfncCamera, [267](#)
- DeviceLinkConnectionCount
 - FliSfncCamera, [267](#)
- DeviceLinkHeartbeatMode
 - FliSfncCamera, [267](#)
- DeviceLinkHeartbeatTimeout
 - FliSfncCamera, [268](#)
- DeviceLinkSelector
 - FliSfncCamera, [268](#)
- DeviceLinkSpeed
 - FliSfncCamera, [268](#)
- DeviceLinkThroughputLimit
 - FliSfncCamera, [268](#)
- DeviceLinkThroughputLimitMode
 - FliSfncCamera, [268](#)
- DeviceManifestEntrySelector
 - FliSfncCamera, [268](#)
- DeviceManifestPrimaryURL
 - FliSfncCamera, [269](#)
- DeviceManifestSchemaMajorVersion
 - FliSfncCamera, [269](#)
- DeviceManifestSchemaMinorVersion
 - FliSfncCamera, [269](#)
- DeviceManifestSecondaryURL
 - FliSfncCamera, [269](#)
- DeviceManifestXMLMajorVersion
 - FliSfncCamera, [269](#)
- DeviceManifestXMLMinorVersion
 - FliSfncCamera, [269](#)
- DeviceManifestXMLSubMinorVersion
 - FliSfncCamera, [270](#)
- DeviceManufacturerInfo
 - FliSfncCamera, [270](#)
- DeviceMaxThroughput
 - FliSfncCamera, [270](#)
- DeviceModelName
 - FliSfncCamera, [270](#)
- DeviceRegistersCheck
 - FliSfncCamera, [270](#)
- DeviceRegistersEndianness
 - FliSfncCamera, [270](#)
- DeviceRegistersStreamingEnd
 - FliSfncCamera, [271](#)
- DeviceRegistersStreamingStart
 - FliSfncCamera, [271](#)
- DeviceRegistersValid
 - FliSfncCamera, [271](#)
- DeviceReset
 - FliSfncCamera, [271](#)
- DeviceScanType
 - FliSfncCamera, [271](#)
- DeviceScanTypeEnum
 - FliCblueSfncEnum, [22](#)
- DeviceScanTypeString
 - FliCblueSfncEnum, [26](#)
- DeviceSerialNumber
 - FliSfncCamera, [271](#)
- DeviceSerialPortBaudRate
 - FliSfncCamera, [272](#)
- DeviceSerialPortSelector
 - FliSfncCamera, [272](#)

- DeviceSFNCVersionMajor
 - FliSfncCamera, [272](#)
- DeviceSFNCVersionMinor
 - FliSfncCamera, [272](#)
- DeviceSFNCVersionSubMinor
 - FliSfncCamera, [272](#)
- DeviceShutdown
 - FliCblueOne, [37](#)
- DeviceStatus
 - FliCblueOne, [37](#)
- DeviceStatusDetailed
 - FliCblueOne, [37](#)
- DeviceStreamChannelCount
 - FliSfncCamera, [272](#)
- DeviceStreamChannelEndianness
 - FliSfncCamera, [273](#)
- DeviceStreamChannelLink
 - FliSfncCamera, [273](#)
- DeviceStreamChannelPacketSize
 - FliSfncCamera, [273](#)
- DeviceStreamChannelSelector
 - FliSfncCamera, [273](#)
- DeviceStreamChannelType
 - FliSfncCamera, [273](#)
- DeviceTapGeometry
 - FliSfncCamera, [273](#)
- DeviceTecCurrent
 - FliCblueOne, [38](#)
- DeviceTecPower
 - FliCblueOne, [38](#)
- DeviceTecSelector
 - FliCblueOne, [38](#)
- DeviceTecSelectorEnum
 - FliCblueOneEnum, [11](#)
- DeviceTecSelectorStringToValue
 - FliCblueOneEnum, [15](#)
- DeviceTecVoltage
 - FliCblueOne, [38](#)
- DeviceTemperature
 - FliSfncCamera, [274](#)
- DeviceTemperatureSelector
 - FliCblueOne, [38](#)
 - FliSfncCamera, [274](#)
- DeviceTemperatureSelectorEnum
 - FliCblueOneEnum, [11](#)
- DeviceTemperatureSelectorStringToValue
 - FliCblueOneEnum, [15](#)
- DeviceTLType
 - FliSfncCamera, [274](#)
- DeviceTLVersionMajor
 - FliSfncCamera, [274](#)
- DeviceTLVersionMinor
 - FliSfncCamera, [274](#)
- DeviceTLVersionSubMinor
 - FliSfncCamera, [274](#)
- DeviceType
 - FliSfncCamera, [275](#)
- DeviceUserID
 - FliSfncCamera, [275](#)
- DeviceVendorName
 - FliSfncCamera, [275](#)
- DeviceVersion
 - FliSfncCamera, [275](#)
- DigitalAll
 - FliCblueSfncEnum, [22](#)
- disableCooling
 - FliOcam2K, [149](#)
- disableGrabN
 - FliRingBuffer, [159](#)
 - FliSdk, [176](#)
 - ImageRingBuffer, [399](#)
- disableLicense
 - FliCredThree, [78](#)
 - FliCredTwo, [106](#)
- display16bImage
 - FliSdk, [176](#)
- display8bImage
 - FliSdk, [177](#)
- Done
 - FliCblueOneEnum, [11, 12](#)
 - FliCblueTwoEnum, [30](#)
- DuplicatedCharactersCorrected
 - FliCblueSfncEnum, [20](#)
- DuplicatedCharactersUncorrected
 - FliCblueSfncEnum, [20](#)
- enable
 - FliRingBuffer, [159](#)
 - ImageRingBuffer, [399](#)
- enable8BitsPixel
 - ImageRingBuffer, [399](#)
- enable8bitsPixel
 - ImageProcessing, [355](#)
 - ImageProcessing, [377](#)
- enable8BitsPixelThermo
 - ImageRingBuffer, [399](#)
- enableAccumulationMode
 - FliRingBuffer, [159](#)
 - ImageRingBuffer, [399](#)
- enableAdaptBias
 - FliCredThree, [79](#)
- enableAgc
 - FliCredThree, [79](#)
 - FliCredTwo, [107](#)
- enableAntiBlooming
 - FliCredThree, [79](#)
 - FliCredTwo, [107](#)
- enableAutoClip
 - ImageProcessing, [355](#)
 - ImageProcessing, [378](#)
- enableAutoExposure
 - ImageProcessing, [355](#)
 - ImageProcessing, [378](#)
- enableBadPixel
 - FliCredThree, [79](#)
 - FliCredTwo, [107](#)
- enableBadPixelsCarto

- IImageProcessing, [356](#)
 - ImageProcessing, [378](#)
- enableBias
 - FliOcam2K, [149](#)
 - FliSerialCamera, [203](#)
- enableClahe
 - IImageProcessing, [356](#)
 - ImageProcessing, [378](#)
- enableCooling
 - FliCredOne, [64](#)
- enableCropping
 - FliCred, [49](#)
- enableDenoising
 - IImageProcessing, [356](#)
 - ImageProcessing, [378](#)
- enableDisplayInfos
 - IImageProcessing, [356](#)
 - ImageProcessing, [378](#)
- enableEvents
 - FliCred, [50](#)
- enableExtSynchro
 - FliCred, [50](#)
- enableFactoryCorrection
 - FliCredTwo, [107](#)
- enableFilters
 - IImageProcessing, [356](#)
 - ImageProcessing, [379](#)
- enableFlat
 - FliOcam2K, [149](#)
 - FliSerialCamera, [203](#)
- enableFollowUpTheRamp
 - FliSdk, [177](#)
 - ImageRingBuffer, [399](#)
- enableFowler
 - FliCredOne, [65](#)
- enableFowlerProcessing
 - FliSdk, [177](#)
- enableGrabN
 - FliRingBuffer, [160](#)
 - FliSdk, [177](#)
 - ImageRingBuffer, [400](#)
- enableHdr
 - FliCredThree, [79](#)
 - FliCredTwo, [107](#)
- enableHdrExtended
 - FliCredThree, [79](#)
 - FliCredTwo, [107](#)
- enableImagesAccumulation
 - IImageProcessing, [356](#)
 - ImageProcessing, [379](#)
- enableImageTags
 - FliCred, [50](#)
- enableImageTagStateChanged
 - FliSdk, [179](#)
- enableIndependentMode
 - IImageProcessing, [357](#)
 - ImageProcessing, [379](#)
- enableIOsForCCsFrameGrabber
 - FliSdk, [179](#)
- enableLed
 - FliCred, [50](#)
- enableLicense
 - FliCredThree, [79](#)
 - FliCredTwo, [107](#)
- enableManualClippingCoeff
 - IImageProcessing, [357](#)
 - ImageProcessing, [379](#)
- enableModelmro
 - ImageRingBuffer, [400](#)
- enableMono8Pixel
 - FliSdk, [179](#)
- enableMono8PixelThermo
 - FliSdk, [179](#)
- enableObserversNotif
 - FliSdk, [179](#)
 - ImageRingBuffer, [400](#)
- enablePowerOverCXP
 - FliSdk, [180](#)
- enableRawImages
 - FliCredOne, [65](#)
 - FliCredThree, [80](#)
 - FliCredTwo, [108](#)
- enableRemoteMaintenance
 - FliCredOne, [65](#)
 - FliCredThree, [80](#)
 - FliCredTwo, [108](#)
- enableRingBuffer
 - FliSdk, [180](#)
- enableSharpen
 - IImageProcessing, [357](#)
 - ImageProcessing, [379](#)
- enableShutter
 - FliOcam2S, [155](#)
- enableShutterBlockOnRead
 - FliOcam2S, [155](#)
- enableShutterCorrectGlitch
 - FliOcam2S, [155](#)
- enableSmoothImage
 - IImageProcessing, [357](#)
 - ImageProcessing, [379](#)
- enableStandby
 - FliCredOne, [65](#)
- enableSubstractMode
 - FliRingBuffer, [160](#)
 - FliSdk, [180](#)
 - ImageRingBuffer, [400](#)
- enableSwSynchro
 - FliCredThree, [80](#)
 - FliCredTwo, [108](#)
- enableTcdsAdjust
 - FliCredThree, [80](#)
 - FliCredTwo, [108](#)
- enableTelnet
 - FliCred, [50](#)
- enableTestPattern
 - FliCredOne, [65](#)

- enableThermo
 - ImageProcessing, [357](#)
 - ImageProcessing, [380](#)
- enableTintGranularity
 - FliCredThree, [80](#)
 - FliCredTwo, [108](#)
- enableUnsignedPixel
 - FliSdk, [180](#)
- enableUnsignedPixels
 - FliCredThree, [80](#)
 - FliCredTwo, [108](#)
- enableVrefAdjust
 - FliCredThree, [80](#)
 - FliCredTwo, [108](#)
- EncoderDivider
 - FliSfncCamera, [275](#)
- EncoderMode
 - FliSfncCamera, [275](#)
- EncoderOutputMode
 - FliSfncCamera, [276](#)
- EncoderReset
 - FliSfncCamera, [276](#)
- EncoderResetActivation
 - FliSfncCamera, [276](#)
- EncoderResetSource
 - FliSfncCamera, [276](#)
- EncoderResolution
 - FliSfncCamera, [276](#)
- EncoderSelector
 - FliSfncCamera, [276](#)
- EncoderSourceA
 - FliSfncCamera, [277](#)
- EncoderSourceB
 - FliSfncCamera, [277](#)
- EncoderStatus
 - FliSfncCamera, [277](#)
- EncoderTimeout
 - FliSfncCamera, [277](#)
- EncoderValue
 - FliSfncCamera, [277](#)
- EncoderValueAtReset
 - FliSfncCamera, [277](#)
- Encoding
 - FliCblueSfncEnum, [20](#)
- ErrorStatus
 - FliCblueSfncEnum, [22](#)
- EventAcquisitionEnd
 - FliSfncCamera, [278](#)
- EventAcquisitionEndFrameID
 - FliSfncCamera, [278](#)
- EventAcquisitionEndTimestamp
 - FliSfncCamera, [278](#)
- EventAcquisitionError
 - FliSfncCamera, [278](#)
- EventAcquisitionErrorFrameID
 - FliSfncCamera, [278](#)
- EventAcquisitionErrorTimestamp
 - FliSfncCamera, [278](#)
- EventAcquisitionStart
 - FliSfncCamera, [279](#)
- EventAcquisitionStartFrameID
 - FliSfncCamera, [279](#)
- EventAcquisitionStartTimestamp
 - FliSfncCamera, [279](#)
- EventAcquisitionTransferEnd
 - FliSfncCamera, [279](#)
- EventAcquisitionTransferEndFrameID
 - FliSfncCamera, [279](#)
- EventAcquisitionTransferEndTimestamp
 - FliSfncCamera, [279](#)
- EventAcquisitionTransferStart
 - FliSfncCamera, [280](#)
- EventAcquisitionTransferStartFrameID
 - FliSfncCamera, [280](#)
- EventAcquisitionTransferStartTimestamp
 - FliSfncCamera, [280](#)
- EventAcquisitionTrigger
 - FliSfncCamera, [280](#)
- EventAcquisitionTriggerFrameID
 - FliSfncCamera, [280](#)
- EventAcquisitionTriggerMissed
 - FliSfncCamera, [280](#)
- EventAcquisitionTriggerMissedFrameID
 - FliSfncCamera, [281](#)
- EventAcquisitionTriggerMissedTimestamp
 - FliSfncCamera, [281](#)
- EventAcquisitionTriggerTimestamp
 - FliSfncCamera, [281](#)
- EventActionLate
 - FliSfncCamera, [281](#)
- EventActionLateFrameID
 - FliSfncCamera, [281](#)
- EventActionLateTimestamp
 - FliSfncCamera, [281](#)
- EventCounter0End
 - FliSfncCamera, [282](#)
- EventCounter0EndFrameID
 - FliSfncCamera, [282](#)
- EventCounter0EndTimestamp
 - FliSfncCamera, [282](#)
- EventCounter0Start
 - FliSfncCamera, [282](#)
- EventCounter0StartFrameID
 - FliSfncCamera, [282](#)
- EventCounter0StartTimestamp
 - FliSfncCamera, [282](#)
- EventCounter1End
 - FliSfncCamera, [283](#)
- EventCounter1EndFrameID
 - FliSfncCamera, [283](#)
- EventCounter1EndTimestamp
 - FliSfncCamera, [283](#)
- EventCounter1Start
 - FliSfncCamera, [283](#)
- EventCounter1StartFrameID
 - FliSfncCamera, [283](#)

- EventCounter1StartTimestamp
 - FliSfncCamera, [283](#)
- EventEncoder0Restarted
 - FliSfncCamera, [284](#)
- EventEncoder0RestartedFrameID
 - FliSfncCamera, [284](#)
- EventEncoder0RestartedTimestamp
 - FliSfncCamera, [284](#)
- EventEncoder0Stopped
 - FliSfncCamera, [284](#)
- EventEncoder0StoppedFrameID
 - FliSfncCamera, [284](#)
- EventEncoder0StoppedTimestamp
 - FliSfncCamera, [284](#)
- EventEncoder1Restarted
 - FliSfncCamera, [285](#)
- EventEncoder1RestartedFrameID
 - FliSfncCamera, [285](#)
- EventEncoder1RestartedTimestamp
 - FliSfncCamera, [285](#)
- EventEncoder1Stopped
 - FliSfncCamera, [285](#)
- EventEncoder1StoppedFrameID
 - FliSfncCamera, [285](#)
- EventEncoder1StoppedTimestamp
 - FliSfncCamera, [285](#)
- EventError
 - FliSfncCamera, [286](#)
- EventErrorCode
 - FliSfncCamera, [286](#)
- EventErrorFrameID
 - FliSfncCamera, [286](#)
- EventErrorTimestamp
 - FliSfncCamera, [286](#)
- EventExposureEnd
 - FliSfncCamera, [286](#)
- EventExposureEndFrameID
 - FliSfncCamera, [286](#)
- EventExposureEndTimestamp
 - FliSfncCamera, [287](#)
- EventExposureStart
 - FliSfncCamera, [287](#)
- EventExposureStartFrameID
 - FliSfncCamera, [287](#)
- EventExposureStartTimestamp
 - FliSfncCamera, [287](#)
- EventFrameBurstEnd
 - FliSfncCamera, [287](#)
- EventFrameBurstEndFrameID
 - FliSfncCamera, [287](#)
- EventFrameBurstEndTimestamp
 - FliSfncCamera, [288](#)
- EventFrameBurstStart
 - FliSfncCamera, [288](#)
- EventFrameBurstStartFrameID
 - FliSfncCamera, [288](#)
- EventFrameBurstStartTimestamp
 - FliSfncCamera, [288](#)
- EventFrameEnd
 - FliSfncCamera, [288](#)
- EventFrameEndFrameID
 - FliSfncCamera, [288](#)
- EventFrameEndTimestamp
 - FliSfncCamera, [289](#)
- EventFrameStart
 - FliSfncCamera, [289](#)
- EventFrameStartFrameID
 - FliSfncCamera, [289](#)
- EventFrameStartTimestamp
 - FliSfncCamera, [289](#)
- EventFrameTransferEnd
 - FliSfncCamera, [289](#)
- EventFrameTransferEndFrameID
 - FliSfncCamera, [289](#)
- EventFrameTransferEndTimestamp
 - FliSfncCamera, [290](#)
- EventFrameTransferStart
 - FliSfncCamera, [290](#)
- EventFrameTransferStartFrameID
 - FliSfncCamera, [290](#)
- EventFrameTransferStartTimestamp
 - FliSfncCamera, [290](#)
- EventFrameTrigger
 - FliSfncCamera, [290](#)
- EventFrameTriggerFrameID
 - FliSfncCamera, [290](#)
- EventFrameTriggerMissed
 - FliSfncCamera, [291](#)
- EventFrameTriggerMissedFrameID
 - FliSfncCamera, [291](#)
- EventFrameTriggerMissedTimestamp
 - FliSfncCamera, [291](#)
- EventFrameTriggerTimestamp
 - FliSfncCamera, [291](#)
- EventLine0AnyEdge
 - FliSfncCamera, [291](#)
- EventLine0AnyEdgeFrameID
 - FliSfncCamera, [291](#)
- EventLine0AnyEdgeTimestamp
 - FliSfncCamera, [292](#)
- EventLine0FallingEdge
 - FliSfncCamera, [292](#)
- EventLine0FallingEdgeFrameID
 - FliSfncCamera, [292](#)
- EventLine0FallingEdgeTimestamp
 - FliSfncCamera, [292](#)
- EventLine0RisingEdge
 - FliSfncCamera, [292](#)
- EventLine0RisingEdgeFrameID
 - FliSfncCamera, [292](#)
- EventLine0RisingEdgeTimestamp
 - FliSfncCamera, [293](#)
- EventLine1AnyEdge
 - FliSfncCamera, [293](#)
- EventLine1AnyEdgeFrameID
 - FliSfncCamera, [293](#)

- EventLine1AnyEdgeTimestamp
 - FliSfncCamera, [293](#)
- EventLine1FallingEdge
 - FliSfncCamera, [293](#)
- EventLine1FallingEdgeFrameID
 - FliSfncCamera, [293](#)
- EventLine1FallingEdgeTimestamp
 - FliSfncCamera, [294](#)
- EventLine1RisingEdge
 - FliSfncCamera, [294](#)
- EventLine1RisingEdgeFrameID
 - FliSfncCamera, [294](#)
- EventLine1RisingEdgeTimestamp
 - FliSfncCamera, [294](#)
- EventLineEnd
 - FliSfncCamera, [294](#)
- EventLineEndFrameID
 - FliSfncCamera, [294](#)
- EventLineEndTimestamp
 - FliSfncCamera, [295](#)
- EventLineStart
 - FliSfncCamera, [295](#)
- EventLineStartFrameID
 - FliSfncCamera, [295](#)
- EventLineStartTimestamp
 - FliSfncCamera, [295](#)
- EventLineTrigger
 - FliSfncCamera, [295](#)
- EventLineTriggerFrameID
 - FliSfncCamera, [295](#)
- EventLineTriggerMissed
 - FliSfncCamera, [296](#)
- EventLineTriggerMissedFrameID
 - FliSfncCamera, [296](#)
- EventLineTriggerMissedTimestamp
 - FliSfncCamera, [296](#)
- EventLineTriggerTimestamp
 - FliSfncCamera, [296](#)
- EventLinkSpeedChange
 - FliSfncCamera, [296](#)
- EventLinkSpeedChangeFrameID
 - FliSfncCamera, [296](#)
- EventLinkSpeedChangeTimestamp
 - FliSfncCamera, [297](#)
- EventLinkTrigger0
 - FliSfncCamera, [297](#)
- EventLinkTrigger0FrameID
 - FliSfncCamera, [297](#)
- EventLinkTrigger0Timestamp
 - FliSfncCamera, [297](#)
- EventLinkTrigger1
 - FliSfncCamera, [297](#)
- EventLinkTrigger1FrameID
 - FliSfncCamera, [297](#)
- EventLinkTrigger1Timestamp
 - FliSfncCamera, [298](#)
- EventNotification
 - FliSfncCamera, [298](#)
- EventPacketCrc
 - FliCblueSfncEnum, [20](#)
- EventPrimaryApplicationSwitch
 - FliSfncCamera, [298](#)
- EventPrimaryApplicationSwitchFrameID
 - FliSfncCamera, [298](#)
- EventPrimaryApplicationSwitchTimestamp
 - FliSfncCamera, [298](#)
- EventSelector
 - FliSfncCamera, [298](#)
- EventSequencerSetChange
 - FliSfncCamera, [299](#)
- EventSequencerSetChangeFrameID
 - FliSfncCamera, [299](#)
- EventSequencerSetChangeTimestamp
 - FliSfncCamera, [299](#)
- EventStream0TransferBlockEnd
 - FliSfncCamera, [299](#)
- EventStream0TransferBlockEndFrameID
 - FliSfncCamera, [299](#)
- EventStream0TransferBlockEndTimestamp
 - FliSfncCamera, [299](#)
- EventStream0TransferBlockStart
 - FliSfncCamera, [300](#)
- EventStream0TransferBlockStartFrameID
 - FliSfncCamera, [300](#)
- EventStream0TransferBlockStartTimestamp
 - FliSfncCamera, [300](#)
- EventStream0TransferBlockTrigger
 - FliSfncCamera, [300](#)
- EventStream0TransferBlockTriggerFrameID
 - FliSfncCamera, [300](#)
- EventStream0TransferBlockTriggerTimestamp
 - FliSfncCamera, [300](#)
- EventStream0TransferBurstEnd
 - FliSfncCamera, [301](#)
- EventStream0TransferBurstEndFrameID
 - FliSfncCamera, [301](#)
- EventStream0TransferBurstEndTimestamp
 - FliSfncCamera, [301](#)
- EventStream0TransferBurstStart
 - FliSfncCamera, [301](#)
- EventStream0TransferBurstStartFrameID
 - FliSfncCamera, [301](#)
- EventStream0TransferBurstStartTimestamp
 - FliSfncCamera, [301](#)
- EventStream0TransferEnd
 - FliSfncCamera, [302](#)
- EventStream0TransferEndFrameID
 - FliSfncCamera, [302](#)
- EventStream0TransferEndTimestamp
 - FliSfncCamera, [302](#)
- EventStream0TransferOverflow
 - FliSfncCamera, [302](#)
- EventStream0TransferOverflowFrameID
 - FliSfncCamera, [302](#)
- EventStream0TransferOverflowTimestamp
 - FliSfncCamera, [302](#)

- EventStream0TransferPause
 - FliSfncCamera, 303
- EventStream0TransferPauseFrameID
 - FliSfncCamera, 303
- EventStream0TransferPauseTimestamp
 - FliSfncCamera, 303
- EventStream0TransferResume
 - FliSfncCamera, 303
- EventStream0TransferResumeFrameID
 - FliSfncCamera, 303
- EventStream0TransferResumeTimestamp
 - FliSfncCamera, 303
- EventStream0TransferStart
 - FliSfncCamera, 304
- EventStream0TransferStartFrameID
 - FliSfncCamera, 304
- EventStream0TransferStartTimestamp
 - FliSfncCamera, 304
- EventTest
 - FliSfncCamera, 304
- EventTestTimestamp
 - FliSfncCamera, 304
- EventTimer0End
 - FliSfncCamera, 304
- EventTimer0EndFrameID
 - FliSfncCamera, 305
- EventTimer0EndTimestamp
 - FliSfncCamera, 305
- EventTimer0Start
 - FliSfncCamera, 305
- EventTimer0StartFrameID
 - FliSfncCamera, 305
- EventTimer0StartTimestamp
 - FliSfncCamera, 305
- EventTimer1End
 - FliSfncCamera, 305
- EventTimer1EndFrameID
 - FliSfncCamera, 306
- EventTimer1EndTimestamp
 - FliSfncCamera, 306
- EventTimer1Start
 - FliSfncCamera, 306
- EventTimer1StartFrameID
 - FliSfncCamera, 306
- EventTimer1StartTimestamp
 - FliSfncCamera, 306
- executeFeature
 - FliGenicamCamera, 137
- exitAllGrabbers
 - FliSdk, 181
- ExposureAuto
 - FliSfncCamera, 306
- ExposureMode
 - FliSfncCamera, 307
- ExposureModeEnum
 - FliCblueSfncEnum, 22
- ExposureModeString
 - FliCblueSfncEnum, 27
- ExposureTime
 - FliSfncCamera, 307
- ExposureTimeMaxReg
 - FliCblueOne, 38
- ExposureTimeMinReg
 - FliCblueOne, 39
- ExposureTimeMode
 - FliSfncCamera, 307
- ExposureTimeSelector
 - FliSfncCamera, 307
- Failed
 - FliCblueOneEnum, 11, 12
 - FliCblueTwoEnum, 30
- featuresListString
 - FliCblueSfncEnum, 27
- FileAccessLength
 - FliSfncCamera, 307
- FileAccessOffset
 - FliSfncCamera, 307
- FileOpenMode
 - FliSfncCamera, 308
- FileOperationExecute
 - FliSfncCamera, 308
- FileOperationResult
 - FliSfncCamera, 308
- FileOperationSelector
 - FliSfncCamera, 308
- FileOperationStatus
 - FliSfncCamera, 308
- FileSelector
 - FliSfncCamera, 308
- FileSize
 - FliSfncCamera, 309
- FirmwareUpdateAbort
 - FliCblueOne, 39
- FirmwareUpdateExecute
 - FliCblueOne, 39
- FirmwareUpdateStatus
 - FliCblueOne, 39
 - FliCblueTwo, 46
- FirmwareUpdateStatusEnum
 - FliCblueOneEnum, 11
 - FliCblueTwoEnum, 30
- FirmwareUpdateStatusRefresh
 - FliCblueOne, 39
- FirmwareUpdateStatusStringToValue
 - FliCblueOneEnum, 15
 - FliCblueTwoEnum, 30
- FirmwareUpdateUri
 - FliCblueOne, 39
- FliCamera
 - FliCred.h, 419
- FliCblueOne, 33
 - AcquisitionFrameRateMaxReg, 36
 - AcquisitionFrameRateMinReg, 36
 - ConversionEfficiency, 36
 - CurrentIPAddress, 36
 - CurrentSubnetMask, 36

- DeviceCoolingEnable, 36
- DeviceCoolingSetpoint, 37
- DeviceFanMode, 37
- DeviceFanSpeed, 37
- DeviceShutdown, 37
- DeviceStatus, 37
- DeviceStatusDetailed, 37
- DeviceTecCurrent, 38
- DeviceTecPower, 38
- DeviceTecSelector, 38
- DeviceTecVoltage, 38
- DeviceTemperatureSelector, 38
- ExposureTimeMaxReg, 38
- ExposureTimeMinReg, 39
- FirmwareUpdateAbort, 39
- FirmwareUpdateExecute, 39
- FirmwareUpdateStatus, 39
- FirmwareUpdateStatusRefresh, 39
- FirmwareUpdateUri, 39
- FliCblueOne, 35
- GlowReduction, 40
- IPMode, 40
- IPReconfigure, 40
- LogCollect, 40
- LogCollectAbort, 40
- LogCollectStatus, 40
- LogCollectStatusRefresh, 41
- LogHistoryDepth, 41
- LogServe, 41
- LogServeAbort, 41
- LogServeUri, 41
- Sparse, 41
- SparseHeight, 42
- SparseMode, 42
- SparseOffsetX, 42
- SparseOffsetY, 42
- SparseSelector, 42
- SparseWidth, 42
- StaticAlternateDomainNameServer, 43
- StaticDefaultGateway, 43
- StaticDomainNameServer, 43
- StaticIPAddress, 43
- StaticSubnetMask, 43
- TestPattern, 43
- TestPatternGeneratorSelector, 44
- UserSetDefault, 44
- UserSetSelector, 44
- FliCblueOne.h, 415
- FliCblueOneEnum, 9
 - Automatic, 10, 12
 - Black, 13
 - Case, 11
 - ConversionEfficiencyEnum, 10
 - ConversionEfficiencyStringValue, 14
 - CPU, 11
 - Default12bits, 14
 - Default8bits, 13, 14
 - DeviceFanModeEnum, 10
 - DeviceFanModeStringValue, 15
 - DeviceTecSelectorEnum, 11
 - DeviceTecSelectorStringValue, 15
 - DeviceTemperatureSelectorEnum, 11
 - DeviceTemperatureSelectorStringValue, 15
 - Done, 11, 12
 - Failed, 11, 12
 - FirmwareUpdateStatusEnum, 11
 - FirmwareUpdateStatusStringValue, 15
 - Frontend, 11
 - GlowReductionEnum, 11
 - GlowReductionStringValue, 15
 - GreyHorizontalRamp, 13
 - Heatsink, 11
 - High, 10
 - HighSensitivity12bits, 14
 - HighSensitivity8bits, 14
 - Idle, 11, 12
 - InProgress, 11, 12
 - IPModeEnum, 12
 - IPModeStringValue, 16
 - LogCollectStatusEnum, 12
 - LogCollectStatusStringValue, 16
 - Low, 10
 - Manual, 10, 12
 - Off, 12, 13
 - On, 12
 - Power, 11
 - Region0, 13
 - Region1, 13
 - Region2, 13
 - Region3, 13
 - Region4, 13
 - Region5, 13
 - Region6, 13
 - Region7, 13
 - Sensor, 11, 13
 - Simulator, 13
 - SimulatorGreyHorizontalRamp, 13
 - SimulatorGreyHorizontalRampMoving, 13
 - SparseModeEnum, 12
 - SparseModeStringValue, 16
 - SparseSelectorEnum, 12
 - SparseSelectorStringValue, 16
 - TEC1, 11
 - TestPatternEnum, 13
 - TestPatternGeneratorSelectorEnum, 13
 - TestPatternGeneratorSelectorStringValue, 17
 - TestPatternStringValue, 17
 - UserSet0, 14
 - UserSet1, 14
 - UserSet2, 14
 - UserSet3, 14
 - UserSet4, 14
 - UserSet5, 14
 - UserSet6, 14
 - UserSet7, 14
 - UserSet8, 14

- UserSet9, 14
- UserSetDefaultEnum, 13
- UserSetDefaultStringToValue, 17
- UserSetSelectorEnum, 14
- UserSetSelectorStringToValue, 17
- White, 13
- FliCblueOneEnum.h, 415
- FliCblueSfncEnum, 18
 - AcquisitionModeEnum, 19
 - AcquisitionModeString, 24
 - Active, 22
 - All, 20
 - AnalogAll, 22
 - Areascan, 22
 - BlackLevelAutoEnum, 19
 - BlackLevelAutoString, 24
 - BlackLevelSelectorEnum, 19
 - BlackLevelSelectorString, 24
 - ConnectionLockLoss, 20
 - Continuous, 19
 - ControlPacketCrc, 20
 - CounterActive, 20
 - CounterOverflow, 20
 - CXP10_X2, 21
 - CXP12_X1, 21
 - CXP12_X2, 21
 - CXP1_X1, 21
 - CXP1_X2, 21
 - CXP6_X2, 21
 - CxpConnectionTestModeEnum, 20
 - CxpConnectionTestModeString, 24
 - CxpErrorCounterSelectorEnum, 20
 - CxpErrorCounterSelectorString, 25
 - CxpErrorCounterStatusEnum, 20
 - CxpErrorCounterStatusString, 25
 - CxpLinkConfigurationEnum, 21
 - CxpLinkConfigurationPreferredEnum, 21
 - CxpLinkConfigurationPreferredString, 25
 - CxpLinkConfigurationStatusEnum, 21
 - CxpLinkConfigurationStatusString, 25
 - CxpLinkConfigurationString, 26
 - CxpSendReceiveSelectorEnum, 21
 - CxpSendReceiveSelectorString, 26
 - DeviceIndicatorModeEnum, 22
 - DeviceIndicatorModeString, 26
 - DeviceScanTypeEnum, 22
 - DeviceScanTypeString, 26
 - DigitalAll, 22
 - DuplicatedCharactersCorrected, 20
 - DuplicatedCharactersUncorrected, 20
 - Encoding, 20
 - ErrorStatus, 22
 - EventPacketCrc, 20
 - ExposureModeEnum, 22
 - ExposureModeString, 27
 - featuresListString, 27
 - GainSelectorEnum, 22
 - GainSelectorString, 27
 - Global, 24
 - GlobalReset, 24
 - Inactive, 22
 - Mode1, 20
 - Mono10, 23
 - Mono12, 23
 - Mono8, 23
 - Off, 19, 20, 23
 - On, 23
 - PixelFormatEnum, 23
 - PixelFormatString, 27
 - Receive, 22
 - Region0, 23
 - RegionDestinationEnum, 23
 - RegionDestinationString, 27
 - RegionModeEnum, 23
 - RegionModeString, 28
 - RegionSelectorEnum, 23
 - RegionSelectorString, 28
 - Rolling, 24
 - Send, 22
 - SensorShutterModeEnum, 23
 - SensorShutterModeString, 28
 - Stream0, 23
 - StreamDataPacketCrc, 20
 - Timed, 22
- FliCblueSfncEnum.h, 417
- FliCblueTwo, 44
 - BinningHorizontal, 45
 - BinningHorizontalMode, 45
 - BinningVertical, 45
 - BinningVerticalMode, 46
 - FirmwareUpdateStatus, 46
 - FliCblueTwo, 45
- FliCblueTwo.h, 418
- FliCblueTwoEnum, 28
 - Average, 29
 - BinningHorizontalModeEnum, 29
 - BinningHorizontalModeStringToValue, 30
 - BinningSelectorEnum, 29
 - BinningSelectorStringToValue, 30
 - BinningVerticalModeEnum, 29
 - BinningVerticalModeStringToValue, 30
 - Done, 30
 - Failed, 30
 - FirmwareUpdateStatusEnum, 30
 - FirmwareUpdateStatusStringToValue, 30
 - Idle, 30
 - InProgress, 30
 - Sensor, 29
 - Sum, 29
- FliCblueTwoEnum.h, 418
- FliCred, 46
 - buildBias, 49
 - buildFlat, 49
 - continueStarting, 49
 - enableCropping, 49
 - enableEvents, 50

- enableExtSynchro, 50
- enableImageTags, 50
- enableLed, 50
- enableTelnet, 50
- FliCred, 48
- FliSerialCamera, 207
- getAduOffset, 50
- getBadPixelModeOnOff, 50
- getBiasState, 51
- getCameraType, 51
- getCheckTag4by4, 51
- getEventsState, 51
- getExcludeBorder, 51
- getExtSynchroState, 51
- getFilteringModeOnOff, 51
- getFlatState, 52
- getHwuid, 52
- getImageTagsState, 52
- getIpConfig, 52
- getIsSlowMode, 52
- getKindOfBadPixelCorrection, 52
- getLedState, 53
- getLogs, 53
- getPassword, 54
- getStatus, 54
- getStatusDetailed, 54
- getThresholdingOnOff, 54
- getUserConvolutionMatrix, 54
- getUserConvolutionMatrixIndex_V2, 55
- getVersionFirmware, 55
- getVersionFirmwareBuild, 55
- getVersionFirmwareDetailed, 55
- getVersionFpga, 55
- getVersionHardware, 55
- getVersions, 56
- restoreFactory, 56
- saveCameraSettings, 56
- sendBiasFile, 56
- sendBiasFromUrl, 57
- sendFlatFile, 57
- sendFlatFromUrl, 57
- setAduOffset, 58
- setBadPixelModeOnOff, 58
- setExcludeBorderOnOff, 58
- setFilteringModeOnOff, 58
- setIpAddress, 58
- setIpAlternateDns, 58
- setIpAutomatic, 59
- setIpDns, 59
- setIpGateway, 59
- setIpManual, 59
- setIpNetmask, 59
- setIpRefresh, 59
- setKindOfBadPixelCorrection, 59
- setPassword, 60
- setSlowMode, 60
- setThresholdingLevelsValues, 60
- setThresholdingOnOff, 60
- setUserConvolutionMatrix, 61
- setUserConvolutionMatrixIndex_V2, 61
- shutDown, 61
- upgradeFirmware, 62
- FliCred.h, 419
 - FliCamera, 419
- FliCredOne, 62
 - enableCooling, 64
 - enableFowler, 65
 - enableRawImages, 65
 - enableRemoteMaintenance, 65
 - enableStandby, 65
 - enableTestPattern, 65
 - FliCredOne, 64
 - FliSerialCamera, 207
 - getAll, 65
 - getAllTemp, 65
 - getCoolingState, 66
 - getCropping, 66
 - getFowlerState, 66
 - getGain, 66
 - getNbReadWoReset, 66
 - getNbRegenGetter, 66
 - getNbSamplePixel, 67
 - getNloop, 67
 - getPhotoCurrent, 67
 - getPowerGetter, 67
 - getPowerPulseTube, 67
 - getPowers, 67
 - getPressure, 67
 - getPulseTubeReady, 68
 - getRawImagesState, 68
 - getReadOutMode, 68
 - getRegenRemainingTime, 68
 - getRemoteMaintenanceState, 68
 - getResetWidth, 68
 - getStandbyState, 68
 - getTelnetState, 69
 - getTempDiode, 69
 - getTempFrontEnd, 69
 - getTempMotherBoard, 69
 - getTempPowerBoard, 69
 - getTempPtController, 69
 - getTempPtMcu, 69
 - getTempSetpoint, 70
 - getTempWater, 70
 - getTestPatternState, 70
 - getVersionFpgaDetailed, 70
 - globalResetBursts, 64
 - globalResetCds, 64
 - globalResetSingle, 64
 - isCroppingValid, 70
 - Mode, 64
 - reboot, 71
 - rollingResetIota, 64
 - rollingResetNro, 64
 - rollingResetSingle, 64
 - sendTestPatternFromUrl, 71

- setCropping, 71
- setCroppingColumns, 71
- setCroppingRows, 71
- setGain, 72
- setMode, 72
- setNbReadWoReset, 72
- setNloop, 72
- setNsamplePixel, 72
- setResetWidth, 72
- startVacuumRegen, 72
- undefined, 64
- FliCredOne.h, 419
- FliCredThree, 73
 - abortBuildNuc, 77
 - AgcParam, 77
 - buildBiasNuc, 77
 - buildFlatHdrC1, 78
 - buildFlatHdrC1Nuc, 78
 - buildFlatHdrC2, 78
 - buildFlatHdrC2Nuc, 78
 - buildFlatNuc, 78
 - deleteLicense, 78
 - disableLicense, 78
 - enableAdaptBias, 79
 - enableAgc, 79
 - enableAntiBlooming, 79
 - enableBadPixel, 79
 - enableHdr, 79
 - enableHdrExtended, 79
 - enableLicense, 79
 - enableRawImages, 80
 - enableRemoteMaintenance, 80
 - enableSwSynchro, 80
 - enableTcdsAdjust, 80
 - enableTintGranularity, 80
 - enableUnsignedPixels, 80
 - enableVrefAdjust, 80
- FliSerialCamera, 207
- getAccumulatedUptime, 81
- getAdaptBiasState, 81
- getAgcParam, 81
- getAgcPriority, 81
- getAgcRoi, 81
- getAgcState, 81
- getAllTemp, 82
- getAntiBloomingState, 82
- getBadPixelState, 82
- getBuildNucProgress, 82
- getConversionGain, 82
- getCropping, 82
- getDarkOptimLevel, 83
- getDate, 83
- getExtMarkerSource, 83
- getExtSynchroExposure, 83
- getExtSynchroPolarity, 83
- getFactoryBadPixelMap, 83
- getHardwareFeatures, 84
- getHdrCalibrationMode, 84
- getHdrExtendedState, 84
- getHdrState, 84
- getImagePattern, 84
- getIpAddress, 84
- getIpAlternateDns, 84
- getIpDns, 85
- getIpGateway, 85
- getIpMode, 85
- getIpNetmask, 85
- getLicenses, 85
- getMaxFpsUsb, 85
- getMaxSyncDelay, 85
- getMaxTintLtr, 86
- getMinFps, 86
- getMinSyncDelay, 86
- getNbFramesPerSwTrig, 86
- getPreset, 86
- getRawImagesState, 86
- getRemoteMaintenanceState, 86
- getSnakeParam, 87
- getSoftwareFeatures, 87
- getStepSyncDelay, 87
- getSwSynchroState, 87
- getSyncDelay, 87
- getSyncSignalSource, 87
- getTcdsAdjustState, 87
- getTelnetState, 88
- getTempAmbiant, 88
- getTempBackEnd, 88
- getTempCpu, 88
- getTempInterface, 88
- getTempSnake, 88
- getTint, 88
- getTintGranularityState, 89
- getTintRange, 89
- getTintStep, 89
- getTlsydel, 89
- getTotalUptime, 89
- getTriggerSource, 89
- getTuning, 89
- getUnsignedPixelsState, 90
- getUploadFirmwareConnectionInfo, 90
- getUptime, 90
- getUserBadPixelMap, 90
- getVrefAdjustState, 90
- isCroppingValid, 90
- level_pix_high_hg, 77
- level_pix_high_mg, 77
- level_pix_low_lg, 77
- level_pix_low_mg, 77
- reboot, 91
- sendBadPixelFile, 91
- sendBadPixelFromUrl, 91
- sendBiasHdrC1File, 91
- sendBiasHdrC1FromUrl, 91
- sendBiasHdrC2File, 91
- sendBiasHdrC2FromUrl, 91

- sendFlatHdrC1File, [92](#)
- sendFlatHdrC1FromUrl, [92](#)
- sendFlatHdrC2File, [92](#)
- sendFlatHdrC2FromUrl, [92](#)
- sendLicenseFile, [92](#)
- setAgcParam, [92](#)
- setAgcPriorityNone, [92](#)
- setAgcPriorityOverExposed, [93](#)
- setAgcPriorityUnderExposed, [93](#)
- setAgcRoi, [93](#)
- setConversionGainHigh, [93](#)
- setConversionGainLow, [93](#)
- setConversionGainMedium, [93](#)
- setCropping, [93](#)
- setCroppingColumns, [94](#)
- setCroppingRows, [94](#)
- setDarkOptimLevel, [94](#)
- setExtSynchroExposureExternal, [94](#)
- setExtSynchroExposureInternal, [94](#)
- setExtSynchroPolarityInverted, [94](#)
- setExtSynchroPolarityStandard, [95](#)
- setFactoryBadPixelMap, [95](#)
- setFrameMarkerSourceCC1, [95](#)
- setFrameMarkerSourceCC2, [95](#)
- setFrameMarkerSourceCC3, [95](#)
- setFrameMarkerSourceCC4, [95](#)
- setFrameMarkerSourceExternal, [95](#)
- setHdrCalibrationC1, [96](#)
- setHdrCalibrationC2, [96](#)
- setHdrCalibrationOff, [96](#)
- setImagePatternConstant, [96](#)
- setImagePatternOff, [96](#)
- setImagePatternRamp, [96](#)
- setNbFramesPerSwTrig, [96](#)
- setPreset, [97](#)
- setPresetNumber, [97](#)
- setSnakeParam, [97](#)
- setSyncDelay, [97](#)
- setSyncSignalSourceCC1, [97](#)
- setSyncSignalSourceCC2, [97](#)
- setSyncSignalSourceCC3, [97](#)
- setSyncSignalSourceCC4, [98](#)
- setSyncSignalSourceExternal, [98](#)
- setTint, [98](#)
- setTlsyDel, [98](#)
- setTriggerSourceExternal, [98](#)
- setTriggerSourceSoftware, [98](#)
- setTuningGeneralUse, [98](#)
- setTuningLongExposure, [99](#)
- setTuningShortExposure, [99](#)
- setUserBadPixelMap, [99](#)
- setVoltageVref, [99](#)
- softwareTrig, [99](#)
- startEthernetGrabber, [99](#)
- startHttpServer, [99](#)
- stopEthernetGrabber, [100](#)
- stopHttpServer, [100](#)
- trigger_nb_frames_hg_to_mg, [77](#)
- trigger_nb_frames_lg_to_mg, [77](#)
- trigger_nb_frames_mg_to_hg, [77](#)
- trigger_nb_frames_mg_to_lg, [77](#)
- trigger_ratio_pixels_hg_to_mg, [77](#)
- trigger_ratio_pixels_lg_to_mg, [77](#)
- trigger_ratio_pixels_mg_to_hg, [77](#)
- trigger_ratio_pixels_mg_to_lg, [77](#)
- xSendBadPixelFile, [100](#)
- xSendBiasFile, [100](#)
- xSendBiasHdrC1File, [100](#)
- xSendBiasHdrC2File, [100](#)
- xSendFlatFile, [100](#)
- xSendFlatHdrC1File, [101](#)
- xSendFlatHdrC2File, [101](#)
- xSendLicenseFile, [101](#)
- FliCredThree.h, [420](#)
- FliCredTwo, [101](#)
 - abortBuildNuc, [105](#)
 - buildBiasNuc, [105](#)
 - buildFlatHdrC1, [106](#)
 - buildFlatHdrC1Nuc, [106](#)
 - buildFlatHdrC2, [106](#)
 - buildFlatHdrC2Nuc, [106](#)
 - buildFlatNuc, [106](#)
 - deleteLicense, [106](#)
 - disableLicense, [106](#)
 - enableAgc, [107](#)
 - enableAntiBlooming, [107](#)
 - enableBadPixel, [107](#)
 - enableFactoryCorrection, [107](#)
 - enableHdr, [107](#)
 - enableHdrExtended, [107](#)
 - enableLicense, [107](#)
 - enableRawImages, [108](#)
 - enableRemoteMaintenance, [108](#)
 - enableSwSynchro, [108](#)
 - enableTcdsAdjust, [108](#)
 - enableTintGranularity, [108](#)
 - enableUnsignedPixels, [108](#)
 - enableVrefAdjust, [108](#)
- FliCredTwo, [105](#)
- FliSerialCamera, [207](#)
- getAccumulatedUptime, [109](#)
- getAgcPriority, [109](#)
- getAgcRoi, [109](#)
- getAgcState, [109](#)
- getAllTemp, [109](#)
- getAntiBloomingState, [109](#)
- getBadPixelState, [110](#)
- getBuildNucProgress, [110](#)
- getConversionGain, [110](#)
- getCropping, [110](#)
- getDarkOptimLevel, [110](#)
- getDate, [110](#)
- getExtMarkerSource, [111](#)
- getExtSynchroExposure, [111](#)
- getExtSynchroPolarity, [111](#)
- getFactoryBadPixelMap, [111](#)

getFactoryCorrectionState, 111
getFanMode, 111
getFanSpeed, 111
getHardwareFeatures, 112
getHdrCalibrationMode, 112
getHdrExtendedState, 112
getHdrState, 112
getImagePattern, 112
getIpAddress, 112
getIpAlternateDns, 112
getIpDns, 113
getIpGateway, 113
getIpMode, 113
getIpNetmask, 113
getLicenses, 113
getMaxFpsUsb, 113
getMaxSyncDelay, 113
getMaxTintItr, 114
getMinFps, 114
getMinSyncDelay, 114
getNbFramesPerSwTrig, 114
getNbReadWoReset, 114
getPowerExternalPeltier, 114
getPowers, 114
getPowerSensor, 115
getPreset, 115
getRawImagesState, 115
getRemoteMaintenanceState, 115
getSnakeParam, 115
getSoftwareFeatures, 115
getStepSyncDelay, 116
getSwSynchroState, 116
getSyncDelay, 116
getSynchronization, 116
getSyncSignalSource, 116
getTcadsAdjustState, 116
getTelnetState, 116
getTempFrontEnd, 117
getTempHeatSink, 117
getTempMotherBoard, 117
getTempPeltier, 117
getTempPowerBoard, 117
getTempSnake, 117
getTempSnakeSetpoint, 117
getTint, 118
getTintGranularityState, 118
getTintRange, 118
getTintStep, 118
getTlsydel, 118
getTotalUptime, 118
getTriggerSource, 118
getTuning, 119
getUnsignedPixelsState, 119
getUploadFirmwareConnectionInfo, 119
getUptime, 119
getUserBadPixelMap, 119
getVoltageVref, 119
getVrefAdjustState, 119
isCroppingValid, 120
reboot, 120
sendBadPixelFile, 120
sendBadPixelFromUrl, 120
sendBiasHdrC1File, 120
sendBiasHdrC1FromUrl, 120
sendBiasHdrC2File, 120
sendBiasHdrC2FromUrl, 121
sendFlatHdrC1File, 121
sendFlatHdrC1FromUrl, 121
sendFlatHdrC2File, 121
sendFlatHdrC2FromUrl, 121
sendLicenseFile, 121
setAgcPriorityNone, 121
setAgcPriorityOverExposed, 122
setAgcPriorityUnderExposed, 122
setAgcRoi, 122
setConversionGainHigh, 122
setConversionGainLow, 122
setConversionGainMedium, 122
setCropping, 122
setCroppingColumns, 123
setCroppingRows, 123
setDarkOptimLevel, 123
setExtSynchroExposureExternal, 123
setExtSynchroExposureInternal, 123
setExtSynchroPolarityInverted, 123
setExtSynchroPolarityStandard, 124
setFactoryBadPixelMap, 124
setFanModeAutomatic, 124
setFanModeManual, 124
setFanSpeed, 124
setFrameMarkerSourceCC1, 124
setFrameMarkerSourceCC2, 124
setFrameMarkerSourceCC3, 125
setFrameMarkerSourceCC4, 125
setFrameMarkerSourceExternal, 125
setHdrCalibrationC1, 125
setHdrCalibrationC2, 125
setHdrCalibrationOff, 125
setImagePatternConstant, 125
setImagePatternOff, 125
setImagePatternRamp, 126
setNbFramesPerSwTrig, 126
setNbReadWoReset, 126
setPreset, 126
setPresetNumber, 126
setSensorTemp, 126
setSnakeParam, 126
setSyncDelay, 127
setSynchronizationCmos, 127
setSynchronizationFullCmos, 127
setSynchronizationLvds, 127
setSyncSignalSourceCC1, 127
setSyncSignalSourceCC2, 127
setSyncSignalSourceCC3, 127
setSyncSignalSourceCC4, 128
setSyncSignalSourceExternal, 128

- setTint, 128
- setTlsyDel, 128
- setTriggerSourceExternal, 128
- setTriggerSourceSoftware, 128
- setTuningGeneralUse, 128
- setTuningLongExposure, 129
- setTuningShortExposure, 129
- setUserBadPixelMap, 129
- setVoltageVref, 129
- softwareTrig, 129
- startEthernetGrabber, 129
- startHttpServer, 129
- stopEthernetGrabber, 130
- stopHttpServer, 130
- xSendBadPixelFile, 130
- xSendBiasFile, 130
- xSendBiasHdrC1File, 130
- xSendBiasHdrC2File, 130
- xSendFlatFile, 130
- xSendFlatHdrC1File, 131
- xSendFlatHdrC2File, 131
- xSendLicenseFile, 131
- FliCredTwo.h, 420
- FliCredTwoLite, 131
 - AUTOMATIC, 132
 - CoolingMode, 132
 - CUSTOM_STEPS, 132
 - FliCredTwoLite, 132
 - FliSerialCamera, 207
 - getCoolingFirstPoint, 133
 - getCoolingMode, 133
 - getCoolingState, 133
 - getCoolingStepWidth, 133
 - getCurrentStep, 133
 - getSensorSetpoint, 133
 - getTecPower, 134
 - MANUAL, 132
 - setCoolingFirstPoint, 134
 - setCoolingMode, 134
 - setCoolingState, 134
 - setCoolingStepWidth, 134
 - setSensorSetpoint, 134
- FliCredTwoLite.h, 420
- FliGenicamCamera, 135
 - _cameraModel, 147
 - _grabber, 147
 - _stringToFeature, 147
 - ~FliGenicamCamera, 136
 - addCallbackAllRegisters, 137
 - addCallbackDimensionsRegisters, 137
 - executeFeature, 137
 - FliGenicamCamera, 136
 - getAccessMode, 138
 - getAssociatedGrabber, 138
 - getBooleanFeature, 138
 - getCameraModel, 139
 - getDoubleFeature, 139
 - getDoubleIncrementFeature, 139
 - getDoubleMaxFeature, 140
 - getDoubleMinFeature, 140
 - getFeatureLength, 140
 - getFeaturesList, 141
 - getIntegerFeature, 141
 - getIntegerIncrementFeature, 141
 - getIntegerMaxFeature, 142
 - getIntegerMinFeature, 142
 - getPollingInterval, 143
 - getRawData, 143
 - getRepresentation, 143
 - getStringFeature, 144
 - getVisibility, 144
 - removeCallbackAllRegisters, 145
 - setBooleanFeature, 145
 - setDoubleFeature, 145
 - setIntegerFeature, 146
 - setRawData, 146
 - setStringFeature, 146
- FliGenicamCamera.h, 420
- FliOcam2K, 147
 - _conf, 153
 - disableCooling, 149
 - enableBias, 149
 - enableFlat, 149
 - FliOcam2K, 149
 - FliSerialCamera, 207
 - getAllTemp, 150
 - getConf, 150
 - getCoolingState, 150
 - getCoolingValue, 150
 - getFps, 150
 - getFpsMax, 150
 - isInterface0, 151
 - isOldFirmware, 151
 - protectionReset, 151
 - resetCoolingAlarm, 151
 - sendBiasFile, 151
 - sendFlatFile, 152
 - sendInterface0Command, 152
 - sendInterface1Command, 152
 - setBiasOffset, 152
 - setCoolingValue, 153
 - setFps, 153
 - setFpsMax, 153
 - setGain, 153
 - setWorkMode, 153
- FliOcam2K.h, 420
 - alarm, 421
 - OCAM2_2_TRACK, 421
 - OCAM2_4_TRACK, 421
 - OCAM2_BINNING, 421
 - OCAM2_BINNING1x3, 421
 - OCAM2_BINNING1x4, 421
 - OCAM2_BINNING2x2, 421
 - OCAM2_BINNING3x3, 421
 - OCAM2_BINNING4x4, 421
 - OCAM2_CROPPING240x120, 421

- OCAM2_CROPPING240x128, 421
- OCAM2_NORMAL, 421
- OCAM2_UNKNOWN, 421
- Ocam2CoolingState, 421
- Ocam2Mode, 421
- off, 421
- on, 421
- FliOcam2S, 154
 - enableShutter, 155
 - enableShutterBlockOnRead, 155
 - enableShutterCorrectGlitch, 155
 - FliOcam2S, 154
 - FliSerialCamera, 207
 - getShutterState, 156
 - sendShutterBias, 156
 - setShutterBlanking, 156
 - setShutterBurst, 156
 - setShutterEnd, 156
 - setShutterExternal, 157
 - setShutterInternal, 157
 - setShutterPulseCount, 157
 - setShutterPulsePosition, 157
 - setShutterPulseWidth, 157
 - setShutterSingle, 157
 - setShutterStep, 157
 - setShutterSweepMode, 158
- FliOcam2S.h, 421
- flipHorizontally
 - lImageProcessing, 357
 - ImageProcessing, 380
- flipVertically
 - lImageProcessing, 358
 - ImageProcessing, 380
- FliRingBuffer, 158
 - disableGrabN, 159
 - enable, 159
 - enableAccumulationMode, 159
 - enableGrabN, 160
 - enableSubstractMode, 160
 - getFilling, 160
 - getLastImageIndex, 160
 - getNbCountError, 161
 - getNumberOfWrap, 161
 - getSizeInFrames, 161
 - getSizeInMo, 161
 - isEnabled, 162
 - isGrabNEnabled, 162
 - isGrabNFinished, 162
 - nbFramesInAccumulation, 162
 - reset, 163
 - resetAccumulation, 163
 - setFowlerOffset, 163
 - setSizeInFrames, 163
 - setSizeInFramesThermo, 163
 - setSizeInMo, 164
- FliRingBuffer.h, 421
- FliSdk, 164
 - ~FliSdk, 169
- addEthernetCamera, 170
- addFakeEthernetCamera, 170
- addImageProcessing, 171
- addObserver, 171
- addRawImageReceivedObserver, 171
- camera, 171
- cblueOne, 172
- cblueSfnc, 172
- cblueTwo, 172
- ConfigOnly, 169
- cred, 172
- credOne, 173
- credThree, 173
- credTwo, 173
- credTwoLite, 173
- defineGrabOnlySlowMode, 174
- detectCameras, 175
- detectEthernetCameras, 175
- detectGrabbers, 175
- detectOneCamera, 176
- disableGrabN, 176
- display16bImage, 176
- display8bImage, 177
- enableFollowUpTheRamp, 177
- enableFowlerProcessing, 177
- enableGrabN, 177
- enableImageTagStateChanged, 179
- enableIOsForCCsFrameGrabber, 179
- enableMono8Pixel, 179
- enableMono8PixelThermo, 179
- enableObserversNotif, 179
- enablePowerOverCXP, 180
- enableRingBuffer, 180
- enableSubstractMode, 180
- enableUnsignedPixel, 180
- exitAllGrabbers, 181
- FliSdk, 169, 170
- forceCurrentCameraModel, 181
- Full, 169
- genicamCamera, 181
- getAvailableSaveFormats, 181
- getBufferFilling, 182
- getBufferNbTimesFull, 182
- getBufferSize, 182
- getBurstFilter, 182
- getCroppingState, 182
- getCurrentCameraModel, 183
- getCurrentCameraName, 183
- getCurrentGrabber, 183
- getCurrentImageDimension, 183
- getDetectedCameras, 184
- getDetectedGrabbers, 184
- getGrabberIsUSB, 184
- getImage, 184
- getImage16b, 185
- getImagesCapacity, 185
- getMode, 185
- getNbCountError, 185

- getOcamFrameNumber, 186
- getRawImage, 186
- getRealFps, 186
- getSize, 187
- GrabOnly, 169
- imageProcessing, 187
- initLog, 187
- isCroppingDataValid, 188
- isCurrentCameraLink, 188
- isGrabNEnabled, 189
- isGrabNFinished, 189
- isMono8Pixel, 189
- isStarted, 189
- isUnsignedPixel, 189
- listAllGrabbers, 190
- loadBuffer, 190, 191
- log, 191
- logOutside, 191
- Mode, 169
- observersNotifEnabled, 192
- ocam2k, 192
- ocam2s, 192
- openMatroxGenicamBrowser, 192
- operator=, 193
- removeImageProcessing, 193
- removeObserver, 193
- removeRawImageReceivedObserver, 193
- resetBuffer, 194
- ringBuffer, 194
- saveBuffer, 194
- serialCamera, 195
- setBufferSize, 195
- setBufferSizeInImages, 195
- setBurstFilter, 196
- setCamera, 196
- setCroppingState, 196
- setFowlerOffset, 197
- setGrabber, 197
- setImageDimension, 197
- setImageDimensionImageRingBuffer, 198
- setImageDimensionImageRingBufferThermo, 198
- setMode, 198
- setNbImagesPerBuffer, 199
- setOcamFrameNumberOffset, 199
- setupFixedCCsFrameGrabber, 199
- sfncCamera, 200
- start, 200
- stop, 200
- update, 200
- version, 200
- FliSdk.h, 422
- FliSdk_addCallbackNewImage_V2
 - FliSdk_C_V2.h, 426
- FliSdk_addEthernetCamera_V2
 - FliSdk_C_V2.h, 426
- FliSdk_addImageProcessing_V2
 - FliSdk_C_V2.h, 427
- FliSdk_C_V2.h, 422
- callbackHandler, 425
- FliSdk_addCallbackNewImage_V2, 426
- FliSdk_addEthernetCamera_V2, 426
- FliSdk_addImageProcessing_V2, 427
- FliSdk_detectCameras_V2, 427
- FliSdk_detectGrabbers_V2, 427
- FliSdk_disableGrabN_V2, 428
- FliSdk_display16bImage_V2, 428
- FliSdk_display8bImage_V2, 428
- FliSdk_enableFollowUpTheRamp_V2, 429
- FliSdk_enableFowlerProcessing_V2, 429
- FliSdk_enableGrabN_V2, 429
- FliSdk_enableRingBuffer_V2, 431
- FliSdk_enableSubstractMode_V2, 431
- FliSdk_enableUnsignedPixel_V2, 431
- FliSdk_exit_V2, 432
- FliSdk_forceCurrentCameraModel_V2, 432
- FliSdk_getAvailableSaveFormats_V2, 432
- FliSdk_getBufferFilling_V2, 433
- FliSdk_getBufferSize_V2, 433
- FliSdk_getBufferWithInfo_V2, 433
- FliSdk_getBurstFilter_V2, 434
- FliSdk_getCameraModelAsString_V2, 434
- FliSdk_getCroppingState_V2, 434
- FliSdk_getCurrentCameraModel_V2, 435
- FliSdk_getCurrentCameraName_V2, 435
- FliSdk_getCurrentImageDimension_V2, 435
- FliSdk_getDetectedCameras_V2, 437
- FliSdk_getDetectedGrabbers_V2, 437
- FliSdk_getFps_V2, 437
- FliSdk_getImagesCapacity_V2, 438
- FliSdk_getNbCountError_V2, 438
- FliSdk_getOcamFrameNumber_V2, 438
- FliSdk_getProcessedImage16b_lv_V2, 439
- FliSdk_getProcessedImage16b_V2, 439
- FliSdk_getProcessedImage_lv_V2, 439
- FliSdk_getProcessedImage_V2, 440
- FliSdk_getRawImage_lv_V2, 440
- FliSdk_getRawImage_V2, 441
- FliSdk_init_V2, 441
- FliSdk_initLog_V2, 441
- FliSdk_isCblueOne_V2, 441
- FliSdk_isCblueSfnc_V2, 442
- FliSdk_isCblueTwo_V2, 442
- FliSdk_isCred_V2, 442
- FliSdk_isCredOne_V2, 443
- FliSdk_isCredThree_V2, 443
- FliSdk_isCredTwo_V2, 443
- FliSdk_isCredTwoLite_V2, 444
- FliSdk_isCroppingDataValid_V2, 444
- FliSdk_isGrabNEnabled_V2, 444
- FliSdk_isGrabNFinished_V2, 446
- FliSdk_isMono8Pixel_V2, 446
- FliSdk_isOcam2k_V2, 446
- FliSdk_isOcam2s_V2, 447
- FliSdk_isSerialCamera_V2, 447
- FliSdk_isStarted_V2, 447
- FliSdk_isUnsignedPixel_V2, 448

- FliSdk_loadBufferFromFile_V2, 448
- FliSdk_loadBufferRaw_V2, 448
- FliSdk_removeCallbackNewImage_V2, 449
- FliSdk_removeImageProcessing_V2, 449
- FliSdk_resetBuffer_V2, 449
- FliSdk_saveBuffer_V2, 450
- FliSdk_saveBufferWithOptions_V2, 450
- FliSdk_setBufferSize_V2, 450
- FliSdk_setBufferSizeInImages_V2, 451
- FliSdk_setBurstFilter_V2, 451
- FliSdk_setCamera_V2, 451
- FliSdk_setCroppingState_V2, 452
- FliSdk_setFowlerOffset_V2, 452
- FliSdk_setFpsTrigger_V2, 453
- FliSdk_setGrabber_V2, 453
- FliSdk_setImageDimension_V2, 453
- FliSdk_setMode_V2, 454
- FliSdk_setNbImagesPerBuffer_V2, 454
- FliSdk_setOcamFrameNumberOffset_V2, 454
- FliSdk_start_V2, 455
- FliSdk_stop_V2, 455
- FliSdk_update_V2, 455
- newImageAvailableCallback, 425
- saveBufferProgressionCallback, 425
- FliSdk_detectCameras_V2
 - FliSdk_C_V2.h, 427
- FliSdk_detectGrabbers_V2
 - FliSdk_C_V2.h, 427
- FliSdk_disableGrabN_V2
 - FliSdk_C_V2.h, 428
- FliSdk_display16bImage_V2
 - FliSdk_C_V2.h, 428
- FliSdk_display8bImage_V2
 - FliSdk_C_V2.h, 428
- FliSdk_enableFollowUpTheRamp_V2
 - FliSdk_C_V2.h, 429
- FliSdk_enableFowlerProcessing_V2
 - FliSdk_C_V2.h, 429
- FliSdk_enableGrabN_V2
 - FliSdk_C_V2.h, 429
- FliSdk_enableRingBuffer_V2
 - FliSdk_C_V2.h, 431
- FliSdk_enableSubstractMode_V2
 - FliSdk_C_V2.h, 431
- FliSdk_enableUnsignedPixel_V2
 - FliSdk_C_V2.h, 431
- FliSdk_exit_V2
 - FliSdk_C_V2.h, 432
- FliSdk_forceCurrentCameraModel_V2
 - FliSdk_C_V2.h, 432
- FliSdk_getAvailableSaveFormats_V2
 - FliSdk_C_V2.h, 432
- FliSdk_getBufferFilling_V2
 - FliSdk_C_V2.h, 433
- FliSdk_getBufferSize_V2
 - FliSdk_C_V2.h, 433
- FliSdk_getBufferWithInfo_V2
 - FliSdk_C_V2.h, 433
- FliSdk_getBurstFilter_V2
 - FliSdk_C_V2.h, 434
- FliSdk_getCameraModelAsString_V2
 - FliSdk_C_V2.h, 434
- FliSdk_getCroppingState_V2
 - FliSdk_C_V2.h, 434
- FliSdk_getCurrentCameraModel_V2
 - FliSdk_C_V2.h, 435
- FliSdk_getCurrentCameraName_V2
 - FliSdk_C_V2.h, 435
- FliSdk_getCurrentImageDimension_V2
 - FliSdk_C_V2.h, 435
- FliSdk_getDetectedCameras_V2
 - FliSdk_C_V2.h, 437
- FliSdk_getDetectedGrabbers_V2
 - FliSdk_C_V2.h, 437
- FliSdk_getFps_V2
 - FliSdk_C_V2.h, 437
- FliSdk_getImagesCapacity_V2
 - FliSdk_C_V2.h, 438
- FliSdk_getNbCountError_V2
 - FliSdk_C_V2.h, 438
- FliSdk_getOcamFrameNumber_V2
 - FliSdk_C_V2.h, 438
- FliSdk_getProcessedImage16b_lv_V2
 - FliSdk_C_V2.h, 439
- FliSdk_getProcessedImage16b_V2
 - FliSdk_C_V2.h, 439
- FliSdk_getProcessedImage_lv_V2
 - FliSdk_C_V2.h, 439
- FliSdk_getProcessedImage_V2
 - FliSdk_C_V2.h, 440
- FliSdk_getRawImage_lv_V2
 - FliSdk_C_V2.h, 440
- FliSdk_getRawImage_V2
 - FliSdk_C_V2.h, 441
- FliSdk_init_V2
 - FliSdk_C_V2.h, 441
- FliSdk_initLog_V2
 - FliSdk_C_V2.h, 441
- FliSdk_isCblueOne_V2
 - FliSdk_C_V2.h, 441
- FliSdk_isCblueSfnc_V2
 - FliSdk_C_V2.h, 442
- FliSdk_isCblueTwo_V2
 - FliSdk_C_V2.h, 442
- FliSdk_isCred_V2
 - FliSdk_C_V2.h, 442
- FliSdk_isCredOne_V2
 - FliSdk_C_V2.h, 443
- FliSdk_isCredThree_V2
 - FliSdk_C_V2.h, 443
- FliSdk_isCredTwo_V2
 - FliSdk_C_V2.h, 443
- FliSdk_isCredTwoLite_V2
 - FliSdk_C_V2.h, 444
- FliSdk_isCroppingDataValid_V2
 - FliSdk_C_V2.h, 444

- FliSdk_isGrabNEnabled_V2
 - FliSdk_C_V2.h, [444](#)
- FliSdk_isGrabNFinished_V2
 - FliSdk_C_V2.h, [446](#)
- FliSdk_isMono8Pixel_V2
 - FliSdk_C_V2.h, [446](#)
- FliSdk_isOcam2k_V2
 - FliSdk_C_V2.h, [446](#)
- FliSdk_isOcam2s_V2
 - FliSdk_C_V2.h, [447](#)
- FliSdk_isSerialCamera_V2
 - FliSdk_C_V2.h, [447](#)
- FliSdk_isStarted_V2
 - FliSdk_C_V2.h, [447](#)
- FliSdk_isUnsignedPixel_V2
 - FliSdk_C_V2.h, [448](#)
- FliSdk_loadBufferFromFile_V2
 - FliSdk_C_V2.h, [448](#)
- FliSdk_loadBufferRaw_V2
 - FliSdk_C_V2.h, [448](#)
- FliSdk_removeCallbackNewImage_V2
 - FliSdk_C_V2.h, [449](#)
- FliSdk_removeImageProcessing_V2
 - FliSdk_C_V2.h, [449](#)
- FliSdk_resetBuffer_V2
 - FliSdk_C_V2.h, [449](#)
- FliSdk_saveBuffer_V2
 - FliSdk_C_V2.h, [450](#)
- FliSdk_saveBufferWithOptions_V2
 - FliSdk_C_V2.h, [450](#)
- FliSdk_setBufferSize_V2
 - FliSdk_C_V2.h, [450](#)
- FliSdk_setBufferSizeInImages_V2
 - FliSdk_C_V2.h, [451](#)
- FliSdk_setBurstFilter_V2
 - FliSdk_C_V2.h, [451](#)
- FliSdk_setCamera_V2
 - FliSdk_C_V2.h, [451](#)
- FliSdk_setCroppingState_V2
 - FliSdk_C_V2.h, [452](#)
- FliSdk_setFowlerOffset_V2
 - FliSdk_C_V2.h, [452](#)
- FliSdk_setFpsTrigger_V2
 - FliSdk_C_V2.h, [453](#)
- FliSdk_setGrabber_V2
 - FliSdk_C_V2.h, [453](#)
- FliSdk_setImageDimension_V2
 - FliSdk_C_V2.h, [453](#)
- FliSdk_setMode_V2
 - FliSdk_C_V2.h, [454](#)
- FliSdk_setNbImagesPerBuffer_V2
 - FliSdk_C_V2.h, [454](#)
- FliSdk_setOcamFrameNumberOffset_V2
 - FliSdk_C_V2.h, [454](#)
- FliSdk_start_V2
 - FliSdk_C_V2.h, [455](#)
- FliSdk_stop_V2
 - FliSdk_C_V2.h, [455](#)
- FliSdk_update_V2
 - FliSdk_C_V2.h, [455](#)
- FliSdkImpl
 - FliSerialCamera, [208](#)
- FliSdkImplCL
 - FliSerialCamera, [208](#)
- FliSerialCamera, [201](#)
 - _cameraModel, [208](#)
 - _croppingFromFunction, [208](#)
 - _customSerial, [208](#)
 - _grabber, [208](#)
 - _needEcho, [208](#)
 - _observers, [209](#)
 - ~FliSerialCamera, [202](#)
 - addObserver, [203](#)
 - enableBias, [203](#)
 - enableFlat, [203](#)
 - FliCred, [207](#)
 - FliCredOne, [207](#)
 - FliCredThree, [207](#)
 - FliCredTwo, [207](#)
 - FliCredTwoLite, [207](#)
 - FliOcam2K, [207](#)
 - FliOcam2S, [207](#)
 - FliSdkImpl, [208](#)
 - FliSdkImplCL, [208](#)
 - FliSerialCamera, [202](#)
 - getCurrentImageDimension, [203](#)
 - getFps, [203](#)
 - getFpsMax, [203](#)
 - getModel, [203](#)
 - getRawData, [204](#)
 - grabberReadSerial, [204](#)
 - grabberWriteSerial, [204](#)
 - isCameraConnected, [204](#)
 - isNumber, [204](#)
 - notifyObservers, [204](#)
 - purgeSerial, [204](#)
 - readSerial, [205](#)
 - removeObserver, [205](#)
 - resynchronizeSerial, [205](#)
 - sendCommand, [205](#), [206](#)
 - setCustomSerial, [206](#)
 - setFps, [206](#)
 - sleep, [206](#)
 - writeSerial, [206](#)
- FliSerialCamera.h, [456](#)
- FliSfncCamera, [209](#)
 - ~FliSfncCamera, [238](#)
 - AcquisitionAbort, [238](#)
 - AcquisitionArm, [238](#)
 - AcquisitionBurstFrameCount, [239](#)
 - AcquisitionFrameCount, [239](#)
 - AcquisitionFrameRate, [239](#)
 - AcquisitionFrameRateEnable, [239](#)
 - AcquisitionLineRate, [239](#)
 - AcquisitionLineRateEnable, [239](#)
 - AcquisitionMode, [240](#)

- AcquisitionStart, [240](#)
- AcquisitionStatus, [240](#)
- AcquisitionStatusSelector, [240](#)
- AcquisitionStop, [240](#)
- AcquisitionStopMode, [240](#)
- ActionDeviceKey, [241](#)
- ActionGroupKey, [241](#)
- ActionGroupMask, [241](#)
- ActionQueueSize, [241](#)
- ActionSelector, [241](#)
- ActionUnconditionalMode, [241](#)
- aPAUSEMACCtrlFramesReceived, [242](#)
- aPAUSEMACCtrlFramesTransmitted, [242](#)
- BalanceRatio, [242](#)
- BalanceRatioSelector, [242](#)
- BalanceWhiteAuto, [242](#)
- BinningHorizontal, [242](#)
- BinningHorizontalMode, [243](#)
- BinningSelector, [243](#)
- BinningVertical, [243](#)
- BinningVerticalMode, [243](#)
- BlackLevel, [243](#)
- BlackLevelAuto, [243](#)
- BlackLevelAutoBalance, [244](#)
- BlackLevelSelector, [244](#)
- CameraPresence, [244](#)
- ChunkBinningHorizontal, [244](#)
- ChunkBinningVertical, [244](#)
- ChunkBlackLevel, [244](#)
- ChunkBlackLevelSelector, [245](#)
- ChunkComponentID, [245](#)
- ChunkComponentIDValue, [245](#)
- ChunkComponentSelector, [245](#)
- ChunkCounterSelector, [245](#)
- ChunkCounterValue, [245](#)
- ChunkDecimationHorizontal, [246](#)
- ChunkDecimationVertical, [246](#)
- ChunkEnable, [246](#)
- ChunkEncoderSelector, [246](#)
- ChunkEncoderStatus, [246](#)
- ChunkEncoderValue, [246](#)
- ChunkExposureTime, [247](#)
- ChunkExposureTimeSelector, [247](#)
- ChunkFrameID, [247](#)
- ChunkGain, [247](#)
- ChunkGainSelector, [247](#)
- ChunkGroupID, [247](#)
- ChunkGroupIDValue, [248](#)
- ChunkGroupSelector, [248](#)
- ChunkHeight, [248](#)
- ChunkLinePitch, [248](#)
- ChunkLineStatusAll, [248](#)
- ChunkModeActive, [248](#)
- ChunkOffsetX, [249](#)
- ChunkOffsetY, [249](#)
- ChunkPixelDynamicRangeMax, [249](#)
- ChunkPixelDynamicRangeMin, [249](#)
- ChunkPixelFormat, [249](#)
- ChunkRegionID, [249](#)
- ChunkRegionIDValue, [250](#)
- ChunkRegionSelector, [250](#)
- ChunkReverseX, [250](#)
- ChunkReverseY, [250](#)
- ChunkScan3dAxisMax, [250](#)
- ChunkScan3dAxisMin, [250](#)
- ChunkScan3dBaseline, [251](#)
- ChunkScan3dCoordinateOffset, [251](#)
- ChunkScan3dCoordinateReferenceSelector, [251](#)
- ChunkScan3dCoordinateReferenceValue, [251](#)
- ChunkScan3dCoordinateScale, [251](#)
- ChunkScan3dCoordinateSelector, [251](#)
- ChunkScan3dCoordinateSystem, [252](#)
- ChunkScan3dCoordinateSystemReference, [252](#)
- ChunkScan3dCoordinateTransformSelector, [252](#)
- ChunkScan3dDistanceUnit, [252](#)
- ChunkScan3dFocalLength, [252](#)
- ChunkScan3dInvalidDataFlag, [252](#)
- ChunkScan3dInvalidDataValue, [253](#)
- ChunkScan3dOutputMode, [253](#)
- ChunkScan3dPrincipalPointU, [253](#)
- ChunkScan3dPrincipalPointV, [253](#)
- ChunkScan3dTransformValue, [253](#)
- ChunkScanLineSelector, [253](#)
- ChunkSelector, [254](#)
- ChunkSequencerSetActive, [254](#)
- ChunkSourceID, [254](#)
- ChunkSourceIDValue, [254](#)
- ChunkSourceSelector, [254](#)
- ChunkStreamChannelID, [254](#)
- ChunkTimerSelector, [255](#)
- ChunkTimerValue, [255](#)
- ChunkTimestamp, [255](#)
- ChunkTimestampLatchValue, [255](#)
- ChunkTransferBlockID, [255](#)
- ChunkTransferQueueCurrentBlockCount, [255](#)
- ChunkTransferStreamID, [256](#)
- ChunkWidth, [256](#)
- ChunkXMLEnable, [256](#)
- CIConfiguration, [256](#)
- CITimeSlotsCount, [256](#)
- ColorTransformationEnable, [256](#)
- ColorTransformationSelector, [257](#)
- ColorTransformationValue, [257](#)
- ColorTransformationValueSelector, [257](#)
- ComponentEnable, [257](#)
- ComponentIDValue, [257](#)
- ComponentSelector, [257](#)
- CounterDuration, [258](#)
- CounterEventActivation, [258](#)
- CounterEventSource, [258](#)
- CounterReset, [258](#)
- CounterResetActivation, [258](#)
- CounterResetSource, [258](#)
- CounterSelector, [259](#)
- CounterStatus, [259](#)
- CounterTriggerActivation, [259](#)

- CounterTriggerSource, [259](#)
- CounterValue, [259](#)
- CounterValueAtReset, [259](#)
- CxpConnectionSelector, [260](#)
- CxpConnectionTestErrorCount, [260](#)
- CxpConnectionTestMode, [260](#)
- CxpConnectionTestPacketCount, [260](#)
- CxpErrorCounterReset, [260](#)
- CxpErrorCounterSelector, [260](#)
- CxpErrorCounterStatus, [261](#)
- CxpErrorCounterValue, [261](#)
- CxpFirstLineTriggerWithFrameStart, [261](#)
- CxpLinkConfiguration, [261](#)
- CxpLinkConfigurationPreferred, [261](#)
- CxpLinkConfigurationStatus, [261](#)
- CxpLinkSharingDuplicateStripe, [262](#)
- CxpLinkSharingEnable, [262](#)
- CxpLinkSharingHorizontalOverlap, [262](#)
- CxpLinkSharingHorizontalStripeCount, [262](#)
- CxpLinkSharingStatus, [262](#)
- CxpLinkSharingSubDeviceSelector, [262](#)
- CxpLinkSharingSubDeviceType, [263](#)
- CxpLinkSharingVerticalOverlap, [263](#)
- CxpLinkSharingVerticalStripeCount, [263](#)
- CxpPoCxpAuto, [263](#)
- CxpPoCxpStatus, [263](#)
- CxpPoCxpTripReset, [263](#)
- CxpPoCxpTurnOff, [264](#)
- CxpSendReceiveSelector, [264](#)
- DecimationHorizontal, [264](#)
- DecimationHorizontalMode, [264](#)
- DecimationVertical, [264](#)
- DecimationVerticalMode, [264](#)
- Deinterlacing, [265](#)
- DeviceCharacterSet, [265](#)
- DeviceClockFrequency, [265](#)
- DeviceClockSelector, [265](#)
- DeviceConnectionSelector, [265](#)
- DeviceConnectionSpeed, [265](#)
- DeviceConnectionStatus, [266](#)
- DeviceEventChannelCount, [266](#)
- DeviceFamilyName, [266](#)
- DeviceFeaturePersistenceEnd, [266](#)
- DeviceFeaturePersistenceStart, [266](#)
- DeviceFirmwareVersion, [266](#)
- DeviceGenCPVersionMajor, [267](#)
- DeviceGenCPVersionMinor, [267](#)
- DeviceIndicatorMode, [267](#)
- DeviceLinkCommandTimeout, [267](#)
- DeviceLinkConnectionCount, [267](#)
- DeviceLinkHeartbeatMode, [267](#)
- DeviceLinkHeartbeatTimeout, [268](#)
- DeviceLinkSelector, [268](#)
- DeviceLinkSpeed, [268](#)
- DeviceLinkThroughputLimit, [268](#)
- DeviceLinkThroughputLimitMode, [268](#)
- DeviceManifestEntrySelector, [268](#)
- DeviceManifestPrimaryURL, [269](#)
- DeviceManifestSchemaMajorVersion, [269](#)
- DeviceManifestSchemaMinorVersion, [269](#)
- DeviceManifestSecondaryURL, [269](#)
- DeviceManifestXMLMajorVersion, [269](#)
- DeviceManifestXMLMinorVersion, [269](#)
- DeviceManifestXMLSubMinorVersion, [270](#)
- DeviceManufacturerInfo, [270](#)
- DeviceMaxThroughput, [270](#)
- DeviceModelName, [270](#)
- DeviceRegistersCheck, [270](#)
- DeviceRegistersEndianness, [270](#)
- DeviceRegistersStreamingEnd, [271](#)
- DeviceRegistersStreamingStart, [271](#)
- DeviceRegistersValid, [271](#)
- DeviceReset, [271](#)
- DeviceScanType, [271](#)
- DeviceSerialNumber, [271](#)
- DeviceSerialPortBaudRate, [272](#)
- DeviceSerialPortSelector, [272](#)
- DeviceSFNCVersionMajor, [272](#)
- DeviceSFNCVersionMinor, [272](#)
- DeviceSFNCVersionSubMinor, [272](#)
- DeviceStreamChannelCount, [272](#)
- DeviceStreamChannelEndianness, [273](#)
- DeviceStreamChannelLink, [273](#)
- DeviceStreamChannelPacketSize, [273](#)
- DeviceStreamChannelSelector, [273](#)
- DeviceStreamChannelType, [273](#)
- DeviceTapGeometry, [273](#)
- DeviceTemperature, [274](#)
- DeviceTemperatureSelector, [274](#)
- DeviceTLType, [274](#)
- DeviceTLVersionMajor, [274](#)
- DeviceTLVersionMinor, [274](#)
- DeviceTLVersionSubMinor, [274](#)
- DeviceType, [275](#)
- DeviceUserID, [275](#)
- DeviceVendorName, [275](#)
- DeviceVersion, [275](#)
- EncoderDivider, [275](#)
- EncoderMode, [275](#)
- EncoderOutputMode, [276](#)
- EncoderReset, [276](#)
- EncoderResetActivation, [276](#)
- EncoderResetSource, [276](#)
- EncoderResolution, [276](#)
- EncoderSelector, [276](#)
- EncoderSourceA, [277](#)
- EncoderSourceB, [277](#)
- EncoderStatus, [277](#)
- EncoderTimeout, [277](#)
- EncoderValue, [277](#)
- EncoderValueAtReset, [277](#)
- EventAcquisitionEnd, [278](#)
- EventAcquisitionEndFrameID, [278](#)
- EventAcquisitionEndTimestamp, [278](#)
- EventAcquisitionError, [278](#)
- EventAcquisitionErrorFrameID, [278](#)

EventAcquisitionErrorTimestamp, 278
EventAcquisitionStart, 279
EventAcquisitionStartFrameID, 279
EventAcquisitionStartTimestamp, 279
EventAcquisitionTransferEnd, 279
EventAcquisitionTransferEndFrameID, 279
EventAcquisitionTransferEndTimestamp, 279
EventAcquisitionTransferStart, 280
EventAcquisitionTransferStartFrameID, 280
EventAcquisitionTransferStartTimestamp, 280
EventAcquisitionTrigger, 280
EventAcquisitionTriggerFrameID, 280
EventAcquisitionTriggerMissed, 280
EventAcquisitionTriggerMissedFrameID, 281
EventAcquisitionTriggerMissedTimestamp, 281
EventAcquisitionTriggerTimestamp, 281
EventActionLate, 281
EventActionLateFrameID, 281
EventActionLateTimestamp, 281
EventCounter0End, 282
EventCounter0EndFrameID, 282
EventCounter0EndTimestamp, 282
EventCounter0Start, 282
EventCounter0StartFrameID, 282
EventCounter0StartTimestamp, 282
EventCounter1End, 283
EventCounter1EndFrameID, 283
EventCounter1EndTimestamp, 283
EventCounter1Start, 283
EventCounter1StartFrameID, 283
EventCounter1StartTimestamp, 283
EventEncoder0Restarted, 284
EventEncoder0RestartedFrameID, 284
EventEncoder0RestartedTimestamp, 284
EventEncoder0Stopped, 284
EventEncoder0StoppedFrameID, 284
EventEncoder0StoppedTimestamp, 284
EventEncoder1Restarted, 285
EventEncoder1RestartedFrameID, 285
EventEncoder1RestartedTimestamp, 285
EventEncoder1Stopped, 285
EventEncoder1StoppedFrameID, 285
EventEncoder1StoppedTimestamp, 285
EventError, 286
EventErrorCode, 286
EventErrorFrameID, 286
EventErrorTimestamp, 286
EventExposureEnd, 286
EventExposureEndFrameID, 286
EventExposureEndTimestamp, 287
EventExposureStart, 287
EventExposureStartFrameID, 287
EventExposureStartTimestamp, 287
EventFrameBurstEnd, 287
EventFrameBurstEndFrameID, 287
EventFrameBurstEndTimestamp, 288
EventFrameBurstStart, 288
EventFrameBurstStartFrameID, 288
EventFrameBurstStartTimestamp, 288
EventFrameEnd, 288
EventFrameEndFrameID, 288
EventFrameEndTimestamp, 289
EventFrameStart, 289
EventFrameStartFrameID, 289
EventFrameStartTimestamp, 289
EventFrameTransferEnd, 289
EventFrameTransferEndFrameID, 289
EventFrameTransferEndTimestamp, 290
EventFrameTransferStart, 290
EventFrameTransferStartFrameID, 290
EventFrameTransferStartTimestamp, 290
EventFrameTrigger, 290
EventFrameTriggerFrameID, 290
EventFrameTriggerMissed, 291
EventFrameTriggerMissedFrameID, 291
EventFrameTriggerMissedTimestamp, 291
EventFrameTriggerTimestamp, 291
EventLine0AnyEdge, 291
EventLine0AnyEdgeFrameID, 291
EventLine0AnyEdgeTimestamp, 292
EventLine0FallingEdge, 292
EventLine0FallingEdgeFrameID, 292
EventLine0FallingEdgeTimestamp, 292
EventLine0RisingEdge, 292
EventLine0RisingEdgeFrameID, 292
EventLine0RisingEdgeTimestamp, 293
EventLine1AnyEdge, 293
EventLine1AnyEdgeFrameID, 293
EventLine1AnyEdgeTimestamp, 293
EventLine1FallingEdge, 293
EventLine1FallingEdgeFrameID, 293
EventLine1FallingEdgeTimestamp, 294
EventLine1RisingEdge, 294
EventLine1RisingEdgeFrameID, 294
EventLine1RisingEdgeTimestamp, 294
EventLineEnd, 294
EventLineEndFrameID, 294
EventLineEndTimestamp, 295
EventLineStart, 295
EventLineStartFrameID, 295
EventLineStartTimestamp, 295
EventLineTrigger, 295
EventLineTriggerFrameID, 295
EventLineTriggerMissed, 296
EventLineTriggerMissedFrameID, 296
EventLineTriggerMissedTimestamp, 296
EventLineTriggerTimestamp, 296
EventLinkSpeedChange, 296
EventLinkSpeedChangeFrameID, 296
EventLinkSpeedChangeTimestamp, 297
EventLinkTrigger0, 297
EventLinkTrigger0FrameID, 297
EventLinkTrigger0Timestamp, 297
EventLinkTrigger1, 297
EventLinkTrigger1FrameID, 297
EventLinkTrigger1Timestamp, 298

EventNotification, 298
EventPrimaryApplicationSwitch, 298
EventPrimaryApplicationSwitchFrameID, 298
EventPrimaryApplicationSwitchTimestamp, 298
EventSelector, 298
EventSequencerSetChange, 299
EventSequencerSetChangeFrameID, 299
EventSequencerSetChangeTimestamp, 299
EventStream0TransferBlockEnd, 299
EventStream0TransferBlockEndFrameID, 299
EventStream0TransferBlockEndTimestamp, 299
EventStream0TransferBlockStart, 300
EventStream0TransferBlockStartFrameID, 300
EventStream0TransferBlockStartTimestamp, 300
EventStream0TransferBlockTrigger, 300
EventStream0TransferBlockTriggerFrameID, 300
EventStream0TransferBlockTriggerTimestamp, 300
EventStream0TransferBurstEnd, 301
EventStream0TransferBurstEndFrameID, 301
EventStream0TransferBurstEndTimestamp, 301
EventStream0TransferBurstStart, 301
EventStream0TransferBurstStartFrameID, 301
EventStream0TransferBurstStartTimestamp, 301
EventStream0TransferEnd, 302
EventStream0TransferEndFrameID, 302
EventStream0TransferEndTimestamp, 302
EventStream0TransferOverflow, 302
EventStream0TransferOverflowFrameID, 302
EventStream0TransferOverflowTimestamp, 302
EventStream0TransferPause, 303
EventStream0TransferPauseFrameID, 303
EventStream0TransferPauseTimestamp, 303
EventStream0TransferResume, 303
EventStream0TransferResumeFrameID, 303
EventStream0TransferResumeTimestamp, 303
EventStream0TransferStart, 304
EventStream0TransferStartFrameID, 304
EventStream0TransferStartTimestamp, 304
EventTest, 304
EventTestTimestamp, 304
EventTimer0End, 304
EventTimer0EndFrameID, 305
EventTimer0EndTimestamp, 305
EventTimer0Start, 305
EventTimer0StartFrameID, 305
EventTimer0StartTimestamp, 305
EventTimer1End, 305
EventTimer1EndFrameID, 306
EventTimer1EndTimestamp, 306
EventTimer1Start, 306
EventTimer1StartFrameID, 306
EventTimer1StartTimestamp, 306
ExposureAuto, 306
ExposureMode, 307
ExposureTime, 307
ExposureTimeMode, 307
ExposureTimeSelector, 307
FileAccessLength, 307
FileAccessOffset, 307
FileOpenMode, 308
FileOperationExecute, 308
FileOperationResult, 308
FileOperationSelector, 308
FileOperationStatus, 308
FileSelector, 308
FileSize, 309
FliSfncCamera, 238
Gain, 309
GainAuto, 309
GainAutoBalance, 309
GainSelector, 309
Gamma, 309
GenDCStreamingMode, 310
GenDCStreamingStatus, 310
GevActiveLinkCount, 310
GevCCP, 310
GevCurrentDefaultGateway, 310
GevCurrentIPAddress, 310
GevCurrentIPConfigurationDHCP, 311
GevCurrentIPConfigurationLLA, 311
GevCurrentIPConfigurationPersistentIP, 311
GevCurrentPhysicalLinkConfiguration, 311
GevCurrentSubnetMask, 311
GevDiscoveryAckDelay, 311
GevFirstURL, 312
GevGVCPEExtendedStatusCodes, 312
GevGVCPEExtendedStatusCodesSelector, 312
GevGVCPPendingAck, 312
GevGVSPExtendedIDMode, 312
GevInterfaceSelector, 312
GevIPConfigurationStatus, 313
GevMACAddress, 313
GevMCDA, 313
GevMCPHostPort, 313
GevMCRC, 313
GevMCSP, 313
GevMCTT, 314
GevPAUSEFrameReception, 314
GevPAUSEFrameTransmission, 314
GevPersistentDefaultGateway, 314
GevPersistentIPAddress, 314
GevPersistentSubnetMask, 314
GevPhysicalLinkConfiguration, 315
GevPrimaryApplicationIPAddress, 315
GevPrimaryApplicationSocket, 315
GevPrimaryApplicationSwitchoverKey, 315
GevSCCFGAllInTransmission, 315
GevSCCFGExtendedChunkData, 315
GevSCCFGPacketResendDestination, 316
GevSCCFGUnconditionalStreaming, 316
GevSCDA, 316
GevSCPD, 316
GevSCPHostPort, 316
GevSCPIInterfaceIndex, 317
GevSCPSDoNotFragment, 317
GevSCPSFireTestPacket, 317

- GevSCSPPacketSize, 317
- GevSCSP, 317
- GevSCZoneConfigurationLock, 317
- GevSCZoneCount, 318
- GevSCZoneDirectionAll, 318
- GevSecondURL, 318
- GevStreamChannelSelector, 318
- GevSupportedOption, 318
- GevSupportedOptionSelector, 318
- GroupIDValue, 319
- GroupSelector, 319
- Height, 319
- HeightMax, 319
- ImageCompressionBitrate, 319
- ImageCompressionJPEGFormatOption, 319
- ImageCompressionMode, 320
- ImageCompressionQuality, 320
- ImageCompressionRateOption, 320
- LightBrightness, 320
- LightConnectionStatus, 320
- LightControllerSelector, 321
- LightControllerSource, 321
- LightCurrentMeasured, 321
- LightCurrentRating, 321
- LightVoltageMeasured, 321
- LightVoltageRating, 321
- LineFormat, 322
- LineInverter, 322
- LineMode, 322
- LinePitch, 322
- LinePitchEnable, 322
- LineSelector, 322
- LineSource, 323
- LineStatus, 323
- LineStatusAll, 323
- LogicBlockFunction, 323
- LogicBlockInputInverter, 323
- LogicBlockInputNumber, 323
- LogicBlockInputSelector, 324
- LogicBlockInputSource, 324
- LogicBlockLUTIndex, 324
- LogicBlockLUTSelector, 324
- LogicBlockLUTValue, 324
- LogicBlockLUTValueAll, 324
- LogicBlockSelector, 325
- LUTEnable, 325
- LUTIndex, 325
- LUTSelector, 325
- LUTValue, 325
- MultiSlopeExposureGradient, 325
- MultiSlopeExposureLimit, 326
- MultiSlopeIntensityLimit, 326
- MultiSlopeKneePointCount, 326
- MultiSlopeKneePointSelector, 326
- MultiSlopeMode, 326
- MultiSlopeSaturationThreshold, 326
- OffsetX, 327
- OffsetY, 327
- PayloadSize, 327
- PixelColorFilter, 327
- PixelDynamicRangeMax, 327
- PixelDynamicRangeMin, 327
- PixelFormat, 328
- PixelFormatInfoID, 328
- PixelFormatInfoSelector, 328
- PixelSize, 328
- PtpClockAccuracy, 328
- PtpClockID, 328
- PtpDataSetLatch, 329
- PtpEnable, 329
- PtpGrandmasterClockID, 329
- PtpOffsetFromMaster, 329
- PtpParentClockID, 329
- PtpServoStatus, 329
- PtpStatus, 330
- RegionDestination, 330
- RegionIDValue, 330
- RegionMode, 330
- RegionSelector, 330
- ReverseX, 330
- ReverseY, 331
- Scan3dAxisMax, 331
- Scan3dAxisMin, 331
- Scan3dBaseline, 331
- Scan3dCoordinateOffset, 331
- Scan3dCoordinateReferenceSelector, 331
- Scan3dCoordinateReferenceValue, 332
- Scan3dCoordinateScale, 332
- Scan3dCoordinateSelector, 332
- Scan3dCoordinateSystem, 332
- Scan3dCoordinateSystemReference, 332
- Scan3dCoordinateTransformSelector, 332
- Scan3dDistanceUnit, 333
- Scan3dExtractionMethod, 333
- Scan3dExtractionSelector, 333
- Scan3dExtractionSource, 333
- Scan3dFocalLength, 333
- Scan3dInvalidDataFlag, 333
- Scan3dInvalidDataValue, 334
- Scan3dOutputMode, 334
- Scan3dPrincipalPointU, 334
- Scan3dPrincipalPointV, 334
- Scan3dTransformValue, 334
- SensorDigitizationTaps, 335
- SensorHeight, 335
- SensorName, 335
- SensorPixelHeight, 335
- SensorPixelWidth, 335
- SensorShutterMode, 335
- SensorTaps, 336
- SensorWidth, 336
- SequencerConfigurationMode, 336
- SequencerFeatureEnable, 336
- SequencerFeatureSelector, 336
- SequencerMode, 336
- SequencerPathSelector, 337

- SequencerSetActive, [337](#)
- SequencerSetLoad, [337](#)
- SequencerSetNext, [337](#)
- SequencerSetSave, [337](#)
- SequencerSetSelector, [337](#)
- SequencerSetStart, [338](#)
- SequencerTriggerActivation, [338](#)
- SequencerTriggerSource, [338](#)
- SoftwareSignalPulse, [338](#)
- SoftwareSignalSelector, [338](#)
- SourceCount, [338](#)
- SourceIDValue, [339](#)
- SourceSelector, [339](#)
- TestEventGenerate, [339](#)
- TestPattern, [339](#)
- TestPatternGeneratorSelector, [339](#)
- TestPayloadFormatMode, [339](#)
- TestPendingAck, [340](#)
- TimerDelay, [340](#)
- TimerDuration, [340](#)
- TimerReset, [340](#)
- TimerSelector, [340](#)
- TimerStatus, [340](#)
- TimerTriggerActivation, [341](#)
- TimerTriggerArmDelay, [341](#)
- TimerTriggerSource, [341](#)
- TimerValue, [341](#)
- Timestamp, [341](#)
- TimestampLatch, [341](#)
- TimestampLatchValue, [342](#)
- TimestampReset, [342](#)
- TLParamsLocked, [342](#)
- TLParamsLockedSelector, [342](#)
- TLParamsLockedState, [342](#)
- TransferAbort, [342](#)
- TransferBlockCount, [343](#)
- TransferBurstCount, [343](#)
- TransferComponentSelector, [343](#)
- TransferControlMode, [343](#)
- TransferOperationMode, [343](#)
- TransferPause, [343](#)
- TransferQueueCurrentBlockCount, [344](#)
- TransferQueueMaxBlockCount, [344](#)
- TransferQueueMode, [344](#)
- TransferResume, [344](#)
- TransferSelector, [344](#)
- TransferStart, [344](#)
- TransferStatus, [345](#)
- TransferStatusSelector, [345](#)
- TransferStop, [345](#)
- TransferStreamChannel, [345](#)
- TransferTriggerActivation, [345](#)
- TransferTriggerMode, [345](#)
- TransferTriggerSelector, [346](#)
- TransferTriggerSource, [346](#)
- TriggerActivation, [346](#)
- TriggerDelay, [346](#)
- TriggerDivider, [346](#)
- TriggerMode, [346](#)
- TriggerMultiplier, [347](#)
- TriggerOverlap, [347](#)
- TriggerSelector, [347](#)
- TriggerSoftware, [347](#)
- TriggerSource, [347](#)
- UserOutputSelector, [347](#)
- UserOutputValue, [348](#)
- UserOutputValueAll, [348](#)
- UserOutputValueAllMask, [348](#)
- UserSetDefault, [348](#)
- UserSetDescription, [348](#)
- UserSetFeatureEnable, [348](#)
- UserSetFeatureSelector, [349](#)
- UserSetLoad, [349](#)
- UserSetSave, [349](#)
- UserSetSelector, [349](#)
- WhiteClip, [349](#)
- WhiteClipSelector, [349](#)
- Width, [350](#)
- WidthMax, [350](#)
- FliSfncCamera.h, [456](#)
- forceCalcMeanStdDevAndHist16b
 - ImageProcessing, [358](#)
 - ImageProcessing, [380](#)
- forceCurrentCameraModel
 - FliSdk, [181](#)
- fpsMax
 - Ocam2Conf, [413](#)
- fpsTrigger
 - IRawImageReceivedObserver, [411](#)
- Frontend
 - FliCblueOneEnum, [11](#)
- Full
 - FliSdk, [169](#)
- Gain
 - FliSfncCamera, [309](#)
- GainAuto
 - FliSfncCamera, [309](#)
- GainAutoBalance
 - FliSfncCamera, [309](#)
- GainSelector
 - FliSfncCamera, [309](#)
- GainSelectorEnum
 - FliCblueSfncEnum, [22](#)
- GainSelectorString
 - FliCblueSfncEnum, [27](#)
- Gamma
 - FliSfncCamera, [309](#)
- GAMMA_CLIPPING
 - ImageProcessing, [354](#)
- GenDCStreamingMode
 - FliSfncCamera, [310](#)
- GenDCStreamingStatus
 - FliSfncCamera, [310](#)
- genicamCamera
 - FliSdk, [181](#)
- getAccessMode

- FliGenicamCamera, 138
- getAccumulatedUptime
 - FliCredThree, 81
 - FliCredTwo, 109
- getAdaptBiasState
 - FliCredThree, 81
- getAduOffset
 - FliCred, 50
- getAgcParam
 - FliCredThree, 81
- getAgcPriority
 - FliCredThree, 81
 - FliCredTwo, 109
- getAgcRoi
 - FliCredThree, 81
 - FliCredTwo, 109
- getAgcState
 - FliCredThree, 81
 - FliCredTwo, 109
- getAll
 - FliCredOne, 65
- getAllTemp
 - FliCredOne, 65
 - FliCredThree, 82
 - FliCredTwo, 109
 - FliOcam2K, 150
- getAntiBloomingState
 - FliCredThree, 82
 - FliCredTwo, 109
- getAssociatedGrabber
 - FliGenicamCamera, 138
- getAvailableSaveFormats
 - FliSdk, 181
- getBadPixelModeOnOff
 - FliCred, 50
- getBadPixelState
 - FliCredThree, 82
 - FliCredTwo, 110
- getBiasState
 - FliCred, 51
- getBooleanFeature
 - FliGenicamCamera, 138
- getBufferFilling
 - FliSdk, 182
- getBufferNbTimesFull
 - FliSdk, 182
- getBufferSize
 - FliSdk, 182
- getBuildNucProgress
 - FliCredThree, 82
 - FliCredTwo, 110
- getBurstFilter
 - FliSdk, 182
- getCameraModel
 - FliGenicamCamera, 139
- getCameraType
 - FliCred, 51
- getCheckTag4by4
 - FliCred, 51
- getClipBlack
 - ImageProcessing, 358
 - ImageProcessing, 380
- getClippingType
 - ImageProcessing, 358
 - ImageProcessing, 380
- getClippingTypeList
 - ImageProcessing, 358
 - ImageProcessing, 381
- getClipWhite
 - ImageProcessing, 358
 - ImageProcessing, 381
- getCoeffA
 - ImageProcessing, 359
 - ImageProcessing, 381
- getCoeffB
 - ImageProcessing, 359
 - ImageProcessing, 381
- getColorMapImage
 - ImageProcessing, 359
 - ImageProcessing, 381
- getColorMapList
 - ImageProcessing, 359
 - ImageProcessing, 381
- getConf
 - FliOcam2K, 150
- getConversionGain
 - FliCredThree, 82
 - FliCredTwo, 110
- getCoolingFirstPoint
 - FliCredTwoLite, 133
- getCoolingMode
 - FliCredTwoLite, 133
- getCoolingState
 - FliCredOne, 66
 - FliCredTwoLite, 133
 - FliOcam2K, 150
- getCoolingStepWidth
 - FliCredTwoLite, 133
- getCoolingValue
 - FliOcam2K, 150
- getCropping
 - FliCredOne, 66
 - FliCredThree, 82
 - FliCredTwo, 110
- getCroppingState
 - FliSdk, 182
- getCurrentCameraModel
 - FliSdk, 183
- getCurrentCameraName
 - FliSdk, 183
- getCurrentGrabber
 - FliSdk, 183
- getCurrentImageDimension
 - FliSdk, 183
 - FliSerialCamera, 203
- getCurrentStep

- FliCredTwoLite, [133](#)
- getDarkOptimLevel
 - FliCredThree, [83](#)
 - FliCredTwo, [110](#)
- getDate
 - FliCredThree, [83](#)
 - FliCredTwo, [110](#)
- getDetectedCameras
 - FliSdk, [184](#)
- getDetectedGrabbers
 - FliSdk, [184](#)
- getDoubleFeature
 - FliGenicamCamera, [139](#)
- getDoubleIncrementFeature
 - FliGenicamCamera, [139](#)
- getDoubleMaxFeature
 - FliGenicamCamera, [140](#)
- getDoubleMinFeature
 - FliGenicamCamera, [140](#)
- getEventsState
 - FliCred, [51](#)
- getExcludeBorder
 - FliCred, [51](#)
- getExtMarkerSource
 - FliCredThree, [83](#)
 - FliCredTwo, [111](#)
- getExtSynchroExposure
 - FliCredThree, [83](#)
 - FliCredTwo, [111](#)
- getExtSynchroPolarity
 - FliCredThree, [83](#)
 - FliCredTwo, [111](#)
- getExtSynchroState
 - FliCred, [51](#)
- getFactoryBadPixelMap
 - FliCredThree, [83](#)
 - FliCredTwo, [111](#)
- getFactoryCorrectionState
 - FliCredTwo, [111](#)
- getFanMode
 - FliCredTwo, [111](#)
- getFanSpeed
 - FliCredTwo, [111](#)
- getFeatureLength
 - FliGenicamCamera, [140](#)
- getFeaturesList
 - FliGenicamCamera, [141](#)
- getFilling
 - FliRingBuffer, [160](#)
 - ImageRingBuffer, [401](#)
- getFilteringModeOnOff
 - FliCred, [51](#)
- getFlatState
 - FliCred, [52](#)
- getFowlerState
 - FliCredOne, [66](#)
- getFps
 - FliOcam2K, [150](#)
 - FliSerialCamera, [203](#)
 - ImageRingBuffer, [401](#)
- getFpsMax
 - FliOcam2K, [150](#)
 - FliSerialCamera, [203](#)
- getGain
 - FliCredOne, [66](#)
- getGamma
 - ImageProcessing, [359](#)
 - ImageProcessing, [382](#)
- getGrabberIsUSB
 - FliSdk, [184](#)
- getHardwareFeatures
 - FliCredThree, [84](#)
 - FliCredTwo, [112](#)
- getHdrCalibrationMode
 - FliCredThree, [84](#)
 - FliCredTwo, [112](#)
- getHdrExtendedState
 - FliCredThree, [84](#)
 - FliCredTwo, [112](#)
- getHdrState
 - FliCredThree, [84](#)
 - FliCredTwo, [112](#)
- getHistogram16b
 - ImageProcessing, [359](#)
 - ImageProcessing, [382](#)
- getHistogram16bNegative
 - ImageProcessing, [360](#)
 - ImageProcessing, [382](#)
- getHistogram16bNegativeNoCompute
 - ImageProcessing, [360](#)
 - ImageProcessing, [382](#)
- getHistogram16bNoCompute
 - ImageProcessing, [360](#)
 - ImageProcessing, [382](#)
- getHistogram8b
 - ImageProcessing, [360](#)
 - ImageProcessing, [382](#)
- getHwuid
 - FliCred, [52](#)
- getImage
 - FliSdk, [184](#)
 - ImageRingBuffer, [401](#)
- getImage16b
 - FliSdk, [185](#)
- getImageDimension
 - ImageRingBuffer, [401](#)
- getImagePattern
 - FliCredThree, [84](#)
 - FliCredTwo, [112](#)
- getImagesCapacity
 - FliSdk, [185](#)
- getImageTagsState
 - FliCred, [52](#)
- getIntegerFeature
 - FliGenicamCamera, [141](#)
- getIntegerIncrementFeature

- FliGenicamCamera, [141](#)
- getIntegerMaxFeature
 - FliGenicamCamera, [142](#)
- getIntegerMinFeature
 - FliGenicamCamera, [142](#)
- getIpAddress
 - FliCredThree, [84](#)
 - FliCredTwo, [112](#)
- getIpAlternateDns
 - FliCredThree, [84](#)
 - FliCredTwo, [112](#)
- getIpConfig
 - FliCred, [52](#)
- getIpDns
 - FliCredThree, [85](#)
 - FliCredTwo, [113](#)
- getIpGateway
 - FliCredThree, [85](#)
 - FliCredTwo, [113](#)
- getIpMode
 - FliCredThree, [85](#)
 - FliCredTwo, [113](#)
- getIpNetmask
 - FliCredThree, [85](#)
 - FliCredTwo, [113](#)
- getIsSlowMode
 - FliCred, [52](#)
- getKindOfBadPixelCorrection
 - FliCred, [52](#)
- getLastImageIndex
 - FliRingBuffer, [160](#)
 - ImageRingBuffer, [401](#)
- getLedState
 - FliCred, [53](#)
- getLicenses
 - FliCredThree, [85](#)
 - FliCredTwo, [113](#)
- getLogs
 - FliCred, [53](#)
- getMaxFpsUsb
 - FliCredThree, [85](#)
 - FliCredTwo, [113](#)
- getMaxSyncDelay
 - FliCredThree, [85](#)
 - FliCredTwo, [113](#)
- getMaxTintItr
 - FliCredThree, [86](#)
 - FliCredTwo, [114](#)
- getMaxVal
 - ImageProcessing, [360](#)
 - ImageProcessing, [383](#)
- getMean16b
 - ImageProcessing, [360](#)
 - ImageProcessing, [383](#)
- getMean16bNoCompute
 - ImageProcessing, [361](#)
 - ImageProcessing, [383](#)
- getMean8b
 - ImageProcessing, [361](#)
 - ImageProcessing, [383](#)
- getMinFps
 - FliCredThree, [86](#)
 - FliCredTwo, [114](#)
- getMinSyncDelay
 - FliCredThree, [86](#)
 - FliCredTwo, [114](#)
- getMinVal
 - ImageProcessing, [361](#)
 - ImageProcessing, [383](#)
- getMode
 - FliSdk, [185](#)
- getModel
 - FliSerialCamera, [203](#)
- getNbCountError
 - FliRingBuffer, [161](#)
 - FliSdk, [185](#)
 - ImageRingBuffer, [401](#)
- getNbFramesPerSwTrig
 - FliCredThree, [86](#)
 - FliCredTwo, [114](#)
- getNbReadWoReset
 - FliCredOne, [66](#)
 - FliCredTwo, [114](#)
- getNbRegenGetter
 - FliCredOne, [66](#)
- getNbSamplePixel
 - FliCredOne, [67](#)
- getNloop
 - FliCredOne, [67](#)
- getNumberOfWrap
 - FliRingBuffer, [161](#)
 - ImageRingBuffer, [402](#)
- getNumThreads
 - ImageProcessing, [361](#)
 - ImageProcessing, [383](#)
- getNumThreadsMax
 - ImageProcessing, [361](#)
 - ImageProcessing, [384](#)
- getOcamFrameNumber
 - FliSdk, [186](#)
 - ImageRingBuffer, [402](#)
- getPassword
 - FliCred, [54](#)
- getPercentOfMean
 - ImageProcessing, [361](#)
 - ImageProcessing, [384](#)
- getPhotoCurrent
 - FliCredOne, [67](#)
- getPollingInterval
 - FliGenicamCamera, [143](#)
- getPowerExternalPeltier
 - FliCredTwo, [114](#)
- getPowerGetter
 - FliCredOne, [67](#)
- getPowerPulseTube
 - FliCredOne, [67](#)

getPowers
 FliCredOne, 67
 FliCredTwo, 114

getPowerSensor
 FliCredTwo, 115

getPreset
 FliCredThree, 86
 FliCredTwo, 115

getPressure
 FliCredOne, 67

getProcessedImage
 IImageProcessing, 362
 ImageProcessing, 384

getProcessedImage16b
 ImageProcessing, 384

getPulseTubeReady
 FliCredOne, 68

getRawData
 FliGenicamCamera, 143
 FliSerialCamera, 204

getRawImage
 FliSdk, 186

getRawImagesState
 FliCredOne, 68
 FliCredThree, 86
 FliCredTwo, 115

getRawThermolImage
 IImageProcessing, 362
 ImageProcessing, 384, 385

getReadOutMode
 FliCredOne, 68

getRealFps
 FliSdk, 186

getRegenRemainingTime
 FliCredOne, 68

getRemoteMaintenanceState
 FliCredOne, 68
 FliCredThree, 86
 FliCredTwo, 115

getRepresentation
 FliGenicamCamera, 143

getResetWidth
 FliCredOne, 68

getRingBuffer
 ImageProcessing, 385

getSensorSetpoint
 FliCredTwoLite, 133

getShutterState
 FliOcam2S, 156

getSize
 FliSdk, 187
 IImageProcessing, 362
 ImageProcessing, 385

getSizeInFrames
 FliRingBuffer, 161
 ImageRingBuffer, 402

getSizeInMo
 FliRingBuffer, 161
 ImageRingBuffer, 402

getSnakeParam
 FliCredThree, 87
 FliCredTwo, 115

getSoftwareFeatures
 FliCredThree, 87
 FliCredTwo, 115

getSpatialStdDev16b
 IImageProcessing, 362
 ImageProcessing, 385

getSpatialStdDev16bNoCompute
 IImageProcessing, 362
 ImageProcessing, 385

getSpatialStdDev8b
 IImageProcessing, 363
 ImageProcessing, 385

getStandbyState
 FliCredOne, 68

getStatus
 FliCred, 54

getStatusDetailed
 FliCred, 54

getStdDevAndMeanSelection
 IImageProcessing, 363
 ImageProcessing, 386

getStepSyncDelay
 FliCredThree, 87
 FliCredTwo, 116

getStringFeature
 FliGenicamCamera, 144

getSwSynchroState
 FliCredThree, 87
 FliCredTwo, 116

getSyncDelay
 FliCredThree, 87
 FliCredTwo, 116

getSynchronization
 FliCredTwo, 116

getSyncSignalSource
 FliCredThree, 87
 FliCredTwo, 116

getTcdsAdjustState
 FliCredThree, 87
 FliCredTwo, 116

getTecPower
 FliCredTwoLite, 134

getTelnetState
 FliCredOne, 69
 FliCredThree, 88
 FliCredTwo, 116

getTempAmbiant
 FliCredThree, 88

getTempBackEnd
 FliCredThree, 88

getTempCpu
 FliCredThree, 88

getTempDiode
 FliCredOne, 69

- getTempFrontEnd
 - FliCredOne, 69
 - FliCredTwo, 117
- getTempHeatSink
 - FliCredTwo, 117
- getTempInterface
 - FliCredThree, 88
- getTempMotherBoard
 - FliCredOne, 69
 - FliCredTwo, 117
- getTempPeltier
 - FliCredTwo, 117
- getTempPowerBoard
 - FliCredOne, 69
 - FliCredTwo, 117
- getTempPtController
 - FliCredOne, 69
- getTempPtMcu
 - FliCredOne, 69
- getTempSetpoint
 - FliCredOne, 70
- getTempSnake
 - FliCredThree, 88
 - FliCredTwo, 117
- getTempSnakeSetpoint
 - FliCredTwo, 117
- getTempWater
 - FliCredOne, 70
- getTestPatternState
 - FliCredOne, 70
- getThermoCalibrationData
 - ImageProcessing, 363
 - ImageProcessing, 386
- getThermoUnit
 - ImageProcessing, 363
 - ImageProcessing, 386
- getThresholdingOnOff
 - FliCred, 54
- getTint
 - FliCredThree, 88
 - FliCredTwo, 118
- getTintGranularityState
 - FliCredThree, 89
 - FliCredTwo, 118
- getTintRange
 - FliCredThree, 89
 - FliCredTwo, 118
- getTintStep
 - FliCredThree, 89
 - FliCredTwo, 118
- getTlsydel
 - FliCredThree, 89
 - FliCredTwo, 118
- getTotalUptime
 - FliCredThree, 89
 - FliCredTwo, 118
- getTriggerSource
 - FliCredThree, 89
- FliCredTwo, 118
- getTuning
 - FliCredThree, 89
 - FliCredTwo, 119
- getUnsignedPixelsState
 - FliCredThree, 90
 - FliCredTwo, 119
- getUploadFirmwareConnectionInfo
 - FliCredThree, 90
 - FliCredTwo, 119
- getUptime
 - FliCredThree, 90
 - FliCredTwo, 119
- getUserBadPixelMap
 - FliCredThree, 90
 - FliCredTwo, 119
- getUserConvolutionMatrix
 - FliCred, 54
- getUserConvolutionMatrixIndex_V2
 - FliCred, 55
- getVersionFirmware
 - FliCred, 55
- getVersionFirmwareBuild
 - FliCred, 55
- getVersionFirmwareDetailed
 - FliCred, 55
- getVersionFpga
 - FliCred, 55
- getVersionFpgaDetailed
 - FliCredOne, 70
- getVersionHardware
 - FliCred, 55
- getVersions
 - FliCred, 56
- getVisibility
 - FliGenicamCamera, 144
- getVoltageVref
 - FliCredTwo, 119
- getVrefAdjustState
 - FliCredThree, 90
 - FliCredTwo, 119
- GevActiveLinkCount
 - FliSfncCamera, 310
- GevCCP
 - FliSfncCamera, 310
- GevCurrentDefaultGateway
 - FliSfncCamera, 310
- GevCurrentIPAddress
 - FliSfncCamera, 310
- GevCurrentIPConfigurationDHCP
 - FliSfncCamera, 311
- GevCurrentIPConfigurationLLA
 - FliSfncCamera, 311
- GevCurrentIPConfigurationPersistentIP
 - FliSfncCamera, 311
- GevCurrentPhysicalLinkConfiguration
 - FliSfncCamera, 311
- GevCurrentSubnetMask

- FliSfncCamera, [311](#)
- GevDiscoveryAckDelay
 - FliSfncCamera, [311](#)
- GevFirstURL
 - FliSfncCamera, [312](#)
- GevGVCPExtendedStatusCodes
 - FliSfncCamera, [312](#)
- GevGVCPExtendedStatusCodesSelector
 - FliSfncCamera, [312](#)
- GevGVCPPendingAck
 - FliSfncCamera, [312](#)
- GevGVSPExtendedIDMode
 - FliSfncCamera, [312](#)
- GevInterfaceSelector
 - FliSfncCamera, [312](#)
- GevIPConfigurationStatus
 - FliSfncCamera, [313](#)
- GevMACAddress
 - FliSfncCamera, [313](#)
- GevMCDA
 - FliSfncCamera, [313](#)
- GevMCPHostPort
 - FliSfncCamera, [313](#)
- GevMCRC
 - FliSfncCamera, [313](#)
- GevMCSP
 - FliSfncCamera, [313](#)
- GevMCTT
 - FliSfncCamera, [314](#)
- GevPAUSEFrameReception
 - FliSfncCamera, [314](#)
- GevPAUSEFrameTransmission
 - FliSfncCamera, [314](#)
- GevPersistentDefaultGateway
 - FliSfncCamera, [314](#)
- GevPersistentIPAddress
 - FliSfncCamera, [314](#)
- GevPersistentSubnetMask
 - FliSfncCamera, [314](#)
- GevPhysicalLinkConfiguration
 - FliSfncCamera, [315](#)
- GevPrimaryApplicationIPAddress
 - FliSfncCamera, [315](#)
- GevPrimaryApplicationSocket
 - FliSfncCamera, [315](#)
- GevPrimaryApplicationSwitchoverKey
 - FliSfncCamera, [315](#)
- GevSCCFGAllInTransmission
 - FliSfncCamera, [315](#)
- GevSCCFGExtendedChunkData
 - FliSfncCamera, [315](#)
- GevSCCFGPacketResendDestination
 - FliSfncCamera, [316](#)
- GevSCCFGUnconditionalStreaming
 - FliSfncCamera, [316](#)
- GevSCDA
 - FliSfncCamera, [316](#)
- GevSCPD
 - FliSfncCamera, [316](#)
- GevSCPHostPort
 - FliSfncCamera, [316](#)
- GevSCPInterfaceIndex
 - FliSfncCamera, [317](#)
- GevSCPSDoNotFragment
 - FliSfncCamera, [317](#)
- GevSCPSFireTestPacket
 - FliSfncCamera, [317](#)
- GevSCPSPacketSize
 - FliSfncCamera, [317](#)
- GevSCSP
 - FliSfncCamera, [317](#)
- GevSCZoneConfigurationLock
 - FliSfncCamera, [317](#)
- GevSCZoneCount
 - FliSfncCamera, [318](#)
- GevSCZoneDirectionAll
 - FliSfncCamera, [318](#)
- GevSecondURL
 - FliSfncCamera, [318](#)
- GevStreamChannelSelector
 - FliSfncCamera, [318](#)
- GevSupportedOption
 - FliSfncCamera, [318](#)
- GevSupportedOptionSelector
 - FliSfncCamera, [318](#)
- Global
 - FliCblueSfncEnum, [24](#)
- GlobalReset
 - FliCblueSfncEnum, [24](#)
- globalResetBursts
 - FliCredOne, [64](#)
- globalResetCds
 - FliCredOne, [64](#)
- globalResetSingle
 - FliCredOne, [64](#)
- GlowReduction
 - FliCblueOne, [40](#)
- GlowReductionEnum
 - FliCblueOneEnum, [11](#)
- GlowReductionStringToValue
 - FliCblueOneEnum, [15](#)
- grabberReadSerial
 - FliSerialCamera, [204](#)
- grabberWriteSerial
 - FliSerialCamera, [204](#)
- GrabOnly
 - FliSdk, [169](#)
- GreyHorizontalRamp
 - FliCblueOneEnum, [13](#)
- GroupIDValue
 - FliSfncCamera, [319](#)
- GroupSelector
 - FliSfncCamera, [319](#)
- Heatsink
 - FliCblueOneEnum, [11](#)
- Height

- FliSfncCamera, 319
- height
 - Ocam2Conf, 413
- HeightMax
 - FliSfncCamera, 319
- High
 - FliCblueOneEnum, 10
- HighSensitivity12bits
 - FliCblueOneEnum, 14
- HighSensitivity8bits
 - FliCblueOneEnum, 14
- Idle
 - FliCblueOneEnum, 11, 12
 - FliCblueTwoEnum, 30
- IFliSdkObserver, 350
 - onCameraChanged, 350
 - onFowlerProcessingStateChanged, 351
 - onGrabNStateChanged, 351
 - onResetBufferTriggered, 351
 - onStartedStateChanged, 351
- IFliSdkObserver.h, 456
- ImageProcessing, 351
 - autoClipEnabled, 355
 - BadPixelsAlgo, 354
 - badPixelsCartoLoaded, 355
 - Camera, 354
 - CELSIUS, 354
 - clip, 355
 - ClippingType, 354
 - enable8bitsPixel, 355
 - enableAutoClip, 355
 - enableAutoExposure, 355
 - enableBadPixelsCarto, 356
 - enableClahe, 356
 - enableDenoising, 356
 - enableDisplayInfos, 356
 - enableFilters, 356
 - enableImagesAccumulation, 356
 - enableIndependentMode, 357
 - enableManualClippingCoeff, 357
 - enableSharpen, 357
 - enableSmoothImage, 357
 - enableThermo, 357
 - flipHorizontally, 357
 - flipVertically, 358
 - forceCalcMeanStdDevAndHist16b, 358
 - GAMMA_CLIPPING, 354
 - getClipBlack, 358
 - getClippingType, 358
 - getClippingTypeList, 358
 - getClipWhite, 358
 - getCoeffA, 359
 - getCoeffB, 359
 - getColorMapImage, 359
 - getColorMapList, 359
 - getGamma, 359
 - getHistogram16b, 359
 - getHistogram16bNegative, 360
 - getHistogram16bNegativeNoCompute, 360
 - getHistogram16bNoCompute, 360
 - getHistogram8b, 360
 - getMaxVal, 360
 - getMean16b, 360
 - getMean16bNoCompute, 361
 - getMean8b, 361
 - getMinVal, 361
 - getNumThreads, 361
 - getNumThreadsMax, 361
 - getPercentOfMean, 361
 - getProcessedImage, 362
 - getRawThermolImage, 362
 - getSize, 362
 - getSpatialStdDev16b, 362
 - getSpatialStdDev16bNoCompute, 362
 - getSpatialStdDev8b, 363
 - getStdDevAndMeanSelection, 363
 - getThermoCalibrationData, 363
 - getThermoUnit, 363
 - isFlippedHorizontally, 363
 - isFlippedVertically, 363
 - isIndependent, 364
 - isThermoEnabled, 364
 - KELVIN, 354
 - LINEAR_CLIPPING, 354
 - LOG_CLIPPING, 354
 - None, 354
 - processMutex, 373
 - setBadPixelsCarto, 364
 - setClaheCliplimit, 364
 - setClaheTileGridSize, 364
 - setClipBlack, 364
 - setClipDepth, 365
 - setClipLimit, 365
 - setClippingAlpha, 365
 - setClippingBeta, 365
 - setClippingType, 365
 - setClipWhite, 366
 - setColorMapping, 366
 - setDenoisingH, 366
 - setDenoisingSearchWindowSize, 366
 - setDenoisingTemplateWindowSize, 366
 - setDimension, 366
 - setDragoBias, 367
 - setDragoGamma, 367
 - setDragoMultiplier, 367
 - setDragoSaturation, 367
 - setGamma, 367
 - setIsThermoThrRaw, 368
 - setMantiukGamma, 368
 - setMantiukMultiplier, 368
 - setMantiukSaturation, 368
 - setMantiukScale, 368
 - setnblImagesAccumulation, 368
 - setNumThreads, 369
 - setPercentOfMean, 369
 - setPixelSign, 369

- setReinhardColorAdapt, 369
- setReinhardGamma, 369
- setReinhardIntensity, 369
- setReinhardLightAdapt, 370
- setRotationAngle, 370
- setRotationAngleText, 370
- setSharpenAlpha, 370
- setSharpenBeta, 370
- setSharpenGamma, 370
- setSharpenKsize, 371
- setSharpenSigmaX, 371
- setSharpenSigmaY, 371
- setStdDevAndMeanSelection, 371
- setThermoCalibrationData, 371
- setThermoUnit, 372
- setToneMappingDrago, 372
- setToneMappingMantiuk, 372
- setToneMappingNormal, 372
- setToneMappingReinhard, 372
- Soft, 354
- ThermoUnit, 354
- updateAutoExposureParam, 372
- ImageProcessing.h, 456
- ImageCompressionBitrate
 - FliSfncCamera, 319
- ImageCompressionJPEGFormatOption
 - FliSfncCamera, 319
- ImageCompressionMode
 - FliSfncCamera, 320
- ImageCompressionQuality
 - FliSfncCamera, 320
- ImageCompressionRateOption
 - FliSfncCamera, 320
- ImageProcessing, 373
 - ~ImageProcessing, 376
 - aduToDegrees, 376, 377
 - autoClipEnabled, 377
 - badPixelsCartoLoaded, 377
 - clip, 377
 - enable8bitsPixel, 377
 - enableAutoClip, 378
 - enableAutoExposure, 378
 - enableBadPixelsCarto, 378
 - enableClahe, 378
 - enableDenoising, 378
 - enableDisplayInfos, 378
 - enableFilters, 379
 - enableImagesAccumulation, 379
 - enableIndependentMode, 379
 - enableManualClippingCoeff, 379
 - enableSharpen, 379
 - enableSmoothImage, 379
 - enableThermo, 380
 - flipHorizontally, 380
 - flipVertically, 380
 - forceCalcMeanStdDevAndHist16b, 380
 - getClipBlack, 380
 - getClippingType, 380
 - getClippingTypeList, 381
 - getClipWhite, 381
 - getCoeffA, 381
 - getCoeffB, 381
 - getColorMapImage, 381
 - getColorMapList, 381
 - getGamma, 382
 - getHistogram16b, 382
 - getHistogram16bNegative, 382
 - getHistogram16bNegativeNoCompute, 382
 - getHistogram16bNoCompute, 382
 - getHistogram8b, 382
 - getMaxVal, 383
 - getMean16b, 383
 - getMean16bNoCompute, 383
 - getMean8b, 383
 - getMinVal, 383
 - getNumThreads, 383
 - getNumThreadsMax, 384
 - getPercentOfMean, 384
 - getProcessedImage, 384
 - getProcessedImage16b, 384
 - getRawThermolImage, 384, 385
 - getRingBuffer, 385
 - getSize, 385
 - getSpatialStdDev16b, 385
 - getSpatialStdDev16bNoCompute, 385
 - getSpatialStdDev8b, 385
 - getStdDevAndMeanSelection, 386
 - getThermoCalibrationData, 386
 - getThermoUnit, 386
 - ImageProcessing, 376
 - isFlippedHorizontally, 386
 - isFlippedVertically, 386
 - isIndependent, 386
 - isThermoEnabled, 387
 - setBadPixelsCarto, 387
 - setCamera, 387
 - setClaheCliplimit, 387
 - setClaheTileGridSize, 387
 - setClipBlack, 388
 - setClipDepth, 388
 - setClipLimit, 388
 - setClippingAlpha, 388
 - setClippingBeta, 388
 - setClippingType, 389
 - setClipWhite, 389
 - setColorMapping, 389
 - setDenoisingH, 389
 - setDenoisingSearchWindowSize, 389
 - setDenoisingTemplateWindowSize, 390
 - setDimension, 390
 - setDragoBias, 390
 - setDragoGamma, 390
 - setDragoMultiplier, 390
 - setDragoSaturation, 391
 - setGamma, 391
 - setIsThermoThrRaw, 391

- setMantiukGamma, 391
- setMantiukMultiplier, 391
- setMantiukSaturation, 391
- setMantiukScale, 392
- setNbImagesAccumulation, 392
- setNumThreads, 392
- setPercentOfMean, 392
- setPixelSign, 392
- setReinhardColorAdapt, 392
- setReinhardGamma, 393
- setReinhardIntensity, 393
- setReinhardLightAdapt, 393
- setRingBuffer, 393
- setRotationAngle, 393
- setRotationAngleText, 393
- setSharpenAlpha, 394
- setSharpenBeta, 394
- setSharpenGamma, 394
- setSharpenKsize, 394
- setSharpenSigmaX, 394
- setSharpenSigmaY, 395
- setStdDevAndMeanSelection, 395
- setThermoCalibrationData, 395
- setThermoUnit, 395
- setToneMappingDrago, 395
- setToneMappingMantiuk, 396
- setToneMappingNormal, 396
- setToneMappingReinhard, 396
- updateAutoExposureParam, 396
- imageProcessing
 - FliSdk, 187
- ImageProcessing.h, 456
- imageReceived
 - IRawImageReceivedObserver, 411
- imageReceivedAfterBuffer
 - IRawImageReceivedObserver, 411
- imageReceivedBeforeBuffer
 - IRawImageReceivedObserver, 412
- ImageRingBuffer, 396
 - ~ImageRingBuffer, 398
 - disableGrabN, 399
 - enable, 399
 - enable8BitsPixel, 399
 - enable8BitsPixelThermo, 399
 - enableAccumulationMode, 399
 - enableFollowUpTheRamp, 399
 - enableGrabN, 400
 - enableModelmro, 400
 - enableObserversNotif, 400
 - enableSubstractMode, 400
 - getFilling, 401
 - getFps, 401
 - getImage, 401
 - getImageDimension, 401
 - getLastImageIndex, 401
 - getNbCountError, 401
 - getNumberOfWrap, 402
 - getOcamFrameNumber, 402
 - getSizeInFrames, 402
 - getSizeInMo, 402
 - ImageRingBuffer, 398
 - isEnabled, 403
 - isGrabNEnabled, 403
 - isGrabNFinished, 403
 - nbFramesInAccumulation, 403
 - put, 404, 405
 - putFollowUpTheRamp, 405
 - putFowler, 405
 - putlota, 406
 - putNro, 406
 - reset, 406
 - resetAccumulation, 406
 - resetCountError, 406
 - resetGrabN, 407
 - resetNbSecondsFps, 407
 - setCameraModel, 407
 - setDefaultCapacity, 407
 - setFowlerOffset, 407
 - setImageDimension, 407
 - setImageDimensionThermo, 408
 - setImageTagState, 408
 - setNbLoop, 408
 - setNbRead, 408
 - setNbReadImro, 408
 - setNbSampPix, 408
 - setObserverList, 409
 - setOcamFrameNumberOffset, 409
 - setSizeInFrames, 409
 - setSizeInFramesThermo, 409
 - setSizeInMo, 410
- ImageRingBuffer.h, 456
- Inactive
 - FliCblueSfncEnum, 22
- initLog
 - FliSdk, 187
- InProgress
 - FliCblueOneEnum, 11, 12
 - FliCblueTwoEnum, 30
- IPMode
 - FliCblueOne, 40
- IPModeEnum
 - FliCblueOneEnum, 12
- IPModeStringToValue
 - FliCblueOneEnum, 16
- IPReconfigure
 - FliCblueOne, 40
- IRawImageReceivedObserver, 410
 - fpsTrigger, 411
 - imageReceived, 411
 - imageReceivedAfterBuffer, 411
 - imageReceivedBeforeBuffer, 412
 - useDeprecatedFunction, 412
- isCameraConnected
 - FliSerialCamera, 204
- isCroppingDataValid
 - FliSdk, 188

- isCroppingValid
 - FliCredOne, 70
 - FliCredThree, 90
 - FliCredTwo, 120
- isCurrentCameraLink
 - FliSdk, 188
- isEnabled
 - FliRingBuffer, 162
 - ImageRingBuffer, 403
- isFlippedHorizontally
 - ImageProcessing, 363
 - ImageProcessing, 386
- isFlippedVertically
 - ImageProcessing, 363
 - ImageProcessing, 386
- isGrabNEnabled
 - FliRingBuffer, 162
 - FliSdk, 189
 - ImageRingBuffer, 403
- isGrabNFinished
 - FliRingBuffer, 162
 - FliSdk, 189
 - ImageRingBuffer, 403
- isIndependent
 - ImageProcessing, 364
 - ImageProcessing, 386
- isInterface0
 - FliOcam2K, 151
- isMono8Pixel
 - FliSdk, 189
- isNumber
 - FliSerialCamera, 204
- isOldFirmware
 - FliOcam2K, 151
- isStarted
 - FliSdk, 189
- isThermoEnabled
 - ImageProcessing, 364
 - ImageProcessing, 387
- isUnsignedPixel
 - FliSdk, 189
- KELVIN
 - ImageProcessing, 354
- level_pix_high_hg
 - FliCredThree, 77
- level_pix_high_mg
 - FliCredThree, 77
- level_pix_low_lg
 - FliCredThree, 77
- level_pix_low_mg
 - FliCredThree, 77
- LightBrightness
 - FliSfncCamera, 320
- LightConnectionStatus
 - FliSfncCamera, 320
- LightControllerSelector
 - FliSfncCamera, 321
- LightControllerSource
 - FliSfncCamera, 321
- LightCurrentMeasured
 - FliSfncCamera, 321
- LightCurrentRating
 - FliSfncCamera, 321
- LightVoltageMeasured
 - FliSfncCamera, 321
- LightVoltageRating
 - FliSfncCamera, 321
- LINEAR_CLIPPING
 - ImageProcessing, 354
- LineFormat
 - FliSfncCamera, 322
- LineInverter
 - FliSfncCamera, 322
- LineMode
 - FliSfncCamera, 322
- LinePitch
 - FliSfncCamera, 322
- LinePitchEnable
 - FliSfncCamera, 322
- LineSelector
 - FliSfncCamera, 322
- LineSource
 - FliSfncCamera, 323
- LineStatus
 - FliSfncCamera, 323
- LineStatusAll
 - FliSfncCamera, 323
- listAllGrabbers
 - FliSdk, 190
- loadBuffer
 - FliSdk, 190, 191
- log
 - FliSdk, 191
- LOG_CLIPPING
 - ImageProcessing, 354
- LogCollect
 - FliCblueOne, 40
- LogCollectAbort
 - FliCblueOne, 40
- LogCollectStatus
 - FliCblueOne, 40
- LogCollectStatusEnum
 - FliCblueOneEnum, 12
- LogCollectStatusRefresh
 - FliCblueOne, 41
- LogCollectStatusStringToValue
 - FliCblueOneEnum, 16
- LogHistoryDepth
 - FliCblueOne, 41
- LogicBlockFunction
 - FliSfncCamera, 323
- LogicBlockInputInverter
 - FliSfncCamera, 323
- LogicBlockInputNumber
 - FliSfncCamera, 323

- LogicBlockInputSelector
 - FliSfncCamera, 324
- LogicBlockInputSource
 - FliSfncCamera, 324
- LogicBlockLUTIndex
 - FliSfncCamera, 324
- LogicBlockLUTSelector
 - FliSfncCamera, 324
- LogicBlockLUTValue
 - FliSfncCamera, 324
- LogicBlockLUTValueAll
 - FliSfncCamera, 324
- LogicBlockSelector
 - FliSfncCamera, 325
- logOutside
 - FliSdk, 191
- LogServe
 - FliCblueOne, 41
- LogServeAbort
 - FliCblueOne, 41
- LogServeUri
 - FliCblueOne, 41
- Low
 - FliCblueOneEnum, 10
- LUTEnable
 - FliSfncCamera, 325
- LUTIndex
 - FliSfncCamera, 325
- LUTSelector
 - FliSfncCamera, 325
- LUTValue
 - FliSfncCamera, 325
- MANUAL
 - FliCredTwoLite, 132
- Manual
 - FliCblueOneEnum, 10, 12
- Mode
 - FliCredOne, 64
 - FliSdk, 169
- Mode1
 - FliCblueSfncEnum, 20
- Mono10
 - FliCblueSfncEnum, 23
- Mono12
 - FliCblueSfncEnum, 23
- Mono8
 - FliCblueSfncEnum, 23
- MultiSlopeExposureGradient
 - FliSfncCamera, 325
- MultiSlopeExposureLimit
 - FliSfncCamera, 326
- MultiSlopeIntensityLimit
 - FliSfncCamera, 326
- MultiSlopeKneePointCount
 - FliSfncCamera, 326
- MultiSlopeKneePointSelector
 - FliSfncCamera, 326
- MultiSlopeMode
 - FliSfncCamera, 326
- MultiSlopeSaturationThreshold
 - FliSfncCamera, 326
- nbFramesInAccumulation
 - FliRingBuffer, 162
 - ImageRingBuffer, 403
- nbIdenticPixels
 - Ocam2Conf, 413
- nbPixels
 - Ocam2Conf, 413
- newImageAvailableCallBack
 - FliSdk_C_V2.h, 425
- None
 - ImageProcessing, 354
- notifyObservers
 - FliSerialCamera, 204
- observersNotifEnabled
 - FliSdk, 192
- OCAM2_2_TRACK
 - FliOcam2K.h, 421
- OCAM2_4_TRACK
 - FliOcam2K.h, 421
- OCAM2_BINNING
 - FliOcam2K.h, 421
- OCAM2_BINNING1x3
 - FliOcam2K.h, 421
- OCAM2_BINNING1x4
 - FliOcam2K.h, 421
- OCAM2_BINNING2x2
 - FliOcam2K.h, 421
- OCAM2_BINNING3x3
 - FliOcam2K.h, 421
- OCAM2_BINNING4x4
 - FliOcam2K.h, 421
- OCAM2_CROPPING240x120
 - FliOcam2K.h, 421
- OCAM2_CROPPING240x128
 - FliOcam2K.h, 421
- OCAM2_NORMAL
 - FliOcam2K.h, 421
- OCAM2_UNKNOWN
 - FliOcam2K.h, 421
- Ocam2Conf, 412
 - binningOffset, 413
 - configFile, 413
 - fpsMax, 413
 - height, 413
 - nbIdenticPixels, 413
 - nbPixels, 413
 - rawHeight, 414
 - rawNbPixels, 414
 - rawWidth, 414
 - width, 414
 - wmode, 414
- Ocam2CoolingState
 - FliOcam2K.h, 421
- ocam2k

- FliSdk, [192](#)
- Ocam2Mode
 - FliOcam2K.h, [421](#)
- ocam2s
 - FliSdk, [192](#)
- Off
 - FliCblueOneEnum, [12](#), [13](#)
 - FliCblueSfncEnum, [19](#), [20](#), [23](#)
- off
 - FliOcam2K.h, [421](#)
- OffsetX
 - FliSfncCamera, [327](#)
- OffsetY
 - FliSfncCamera, [327](#)
- On
 - FliCblueOneEnum, [12](#)
 - FliCblueSfncEnum, [23](#)
- on
 - FliOcam2K.h, [421](#)
- onCameraChanged
 - IFliSdkObserver, [350](#)
- onFowlerProcessingStateChanged
 - IFliSdkObserver, [351](#)
- onGrabNStateChanged
 - IFliSdkObserver, [351](#)
- onResetBufferTriggered
 - IFliSdkObserver, [351](#)
- onStartedStateChanged
 - IFliSdkObserver, [351](#)
- openMatroxGenicamBrowser
 - FliSdk, [192](#)
- operator=
 - FliSdk, [193](#)
- PayloadSize
 - FliSfncCamera, [327](#)
- PixelColorFilter
 - FliSfncCamera, [327](#)
- PixelDynamicRangeMax
 - FliSfncCamera, [327](#)
- PixelDynamicRangeMin
 - FliSfncCamera, [327](#)
- PixelFormat
 - FliSfncCamera, [328](#)
- PixelFormatEnum
 - FliCblueSfncEnum, [23](#)
- PixelFormatInfoID
 - FliSfncCamera, [328](#)
- PixelFormatInfoSelector
 - FliSfncCamera, [328](#)
- PixelFormatString
 - FliCblueSfncEnum, [27](#)
- PixelSize
 - FliSfncCamera, [328](#)
- Power
 - FliCblueOneEnum, [11](#)
- processMutex
 - ImageProcessing, [373](#)
- protectionReset
 - FliOcam2K, [151](#)
- PtpClockAccuracy
 - FliSfncCamera, [328](#)
- PtpClockID
 - FliSfncCamera, [328](#)
- PtpDataSetLatch
 - FliSfncCamera, [329](#)
- PtpEnable
 - FliSfncCamera, [329](#)
- PtpGrandmasterClockID
 - FliSfncCamera, [329](#)
- PtpOffsetFromMaster
 - FliSfncCamera, [329](#)
- PtpParentClockID
 - FliSfncCamera, [329](#)
- PtpServoStatus
 - FliSfncCamera, [329](#)
- PtpStatus
 - FliSfncCamera, [330](#)
- purgeSerial
 - FliSerialCamera, [204](#)
- put
 - ImageRingBuffer, [404](#), [405](#)
- putFollowUpTheRamp
 - ImageRingBuffer, [405](#)
- putFowler
 - ImageRingBuffer, [405](#)
- putlota
 - ImageRingBuffer, [406](#)
- putNro
 - ImageRingBuffer, [406](#)
- rawHeight
 - Ocam2Conf, [414](#)
- rawNbPixels
 - Ocam2Conf, [414](#)
- rawWidth
 - Ocam2Conf, [414](#)
- readSerial
 - FliSerialCamera, [205](#)
- reboot
 - FliCredOne, [71](#)
 - FliCredThree, [91](#)
 - FliCredTwo, [120](#)
- Receive
 - FliCblueSfncEnum, [22](#)
- Region0
 - FliCblueOneEnum, [13](#)
 - FliCblueSfncEnum, [23](#)
- Region1
 - FliCblueOneEnum, [13](#)
- Region2
 - FliCblueOneEnum, [13](#)
- Region3
 - FliCblueOneEnum, [13](#)
- Region4
 - FliCblueOneEnum, [13](#)
- Region5
 - FliCblueOneEnum, [13](#)

- Region6
 - FliCblueOneEnum, 13
- Region7
 - FliCblueOneEnum, 13
- RegionDestination
 - FliSfncCamera, 330
- RegionDestinationEnum
 - FliCblueSfncEnum, 23
- RegionDestinationString
 - FliCblueSfncEnum, 27
- RegionIDValue
 - FliSfncCamera, 330
- RegionMode
 - FliSfncCamera, 330
- RegionModeEnum
 - FliCblueSfncEnum, 23
- RegionModeString
 - FliCblueSfncEnum, 28
- RegionSelector
 - FliSfncCamera, 330
- RegionSelectorEnum
 - FliCblueSfncEnum, 23
- RegionSelectorString
 - FliCblueSfncEnum, 28
- removeCallbackAllRegisters
 - FliGenicamCamera, 145
- removeImageProcessing
 - FliSdk, 193
- removeObserver
 - FliSdk, 193
 - FliSerialCamera, 205
- removeRawImageReceivedObserver
 - FliSdk, 193
- reset
 - FliRingBuffer, 163
 - ImageRingBuffer, 406
- resetAccumulation
 - FliRingBuffer, 163
 - ImageRingBuffer, 406
- resetBuffer
 - FliSdk, 194
- resetCoolingAlarm
 - FliOcam2K, 151
- resetCountError
 - ImageRingBuffer, 406
- resetGrabN
 - ImageRingBuffer, 407
- resetNbSecondsFps
 - ImageRingBuffer, 407
- restoreFactory
 - FliCred, 56
- resynchronizeSerial
 - FliSerialCamera, 205
- ReverseX
 - FliSfncCamera, 330
- ReverseY
 - FliSfncCamera, 331
- ringBuffer
 - FliSdk, 194
- Rolling
 - FliCblueSfncEnum, 24
- rollingResetIota
 - FliCredOne, 64
- rollingResetNro
 - FliCredOne, 64
- rollingResetSingle
 - FliCredOne, 64
- saveBuffer
 - FliSdk, 194
- saveBufferProgressionCallback
 - FliSdk_C_V2.h, 425
- saveCameraSettings
 - FliCred, 56
- Scan3dAxisMax
 - FliSfncCamera, 331
- Scan3dAxisMin
 - FliSfncCamera, 331
- Scan3dBaseline
 - FliSfncCamera, 331
- Scan3dCoordinateOffset
 - FliSfncCamera, 331
- Scan3dCoordinateReferenceSelector
 - FliSfncCamera, 331
- Scan3dCoordinateReferenceValue
 - FliSfncCamera, 332
- Scan3dCoordinateScale
 - FliSfncCamera, 332
- Scan3dCoordinateSelector
 - FliSfncCamera, 332
- Scan3dCoordinateSystem
 - FliSfncCamera, 332
- Scan3dCoordinateSystemReference
 - FliSfncCamera, 332
- Scan3dCoordinateTransformSelector
 - FliSfncCamera, 332
- Scan3dDistanceUnit
 - FliSfncCamera, 333
- Scan3dExtractionMethod
 - FliSfncCamera, 333
- Scan3dExtractionSelector
 - FliSfncCamera, 333
- Scan3dExtractionSource
 - FliSfncCamera, 333
- Scan3dFocalLength
 - FliSfncCamera, 333
- Scan3dInvalidDataFlag
 - FliSfncCamera, 333
- Scan3dInvalidDataValue
 - FliSfncCamera, 334
- Scan3dOutputMode
 - FliSfncCamera, 334
- Scan3dPrincipalPointU
 - FliSfncCamera, 334
- Scan3dPrincipalPointV
 - FliSfncCamera, 334
- Scan3dTransformValue

- FliSfncCamera, [334](#)
- Send
 - FliCblueSfncEnum, [22](#)
- sendBadPixelFile
 - FliCredThree, [91](#)
 - FliCredTwo, [120](#)
- sendBadPixelFromUrl
 - FliCredThree, [91](#)
 - FliCredTwo, [120](#)
- sendBiasFile
 - FliCred, [56](#)
 - FliOcam2K, [151](#)
- sendBiasFromUrl
 - FliCred, [57](#)
- sendBiasHdrC1File
 - FliCredThree, [91](#)
 - FliCredTwo, [120](#)
- sendBiasHdrC1FromUrl
 - FliCredThree, [91](#)
 - FliCredTwo, [120](#)
- sendBiasHdrC2File
 - FliCredThree, [91](#)
 - FliCredTwo, [120](#)
- sendBiasHdrC2FromUrl
 - FliCredThree, [91](#)
 - FliCredTwo, [121](#)
- sendCommand
 - FliSerialCamera, [205](#), [206](#)
- sendFlatFile
 - FliCred, [57](#)
 - FliOcam2K, [152](#)
- sendFlatFromUrl
 - FliCred, [57](#)
- sendFlatHdrC1File
 - FliCredThree, [92](#)
 - FliCredTwo, [121](#)
- sendFlatHdrC1FromUrl
 - FliCredThree, [92](#)
 - FliCredTwo, [121](#)
- sendFlatHdrC2File
 - FliCredThree, [92](#)
 - FliCredTwo, [121](#)
- sendFlatHdrC2FromUrl
 - FliCredThree, [92](#)
 - FliCredTwo, [121](#)
- sendInterface0Command
 - FliOcam2K, [152](#)
- sendInterface1Command
 - FliOcam2K, [152](#)
- sendLicenseFile
 - FliCredThree, [92](#)
 - FliCredTwo, [121](#)
- sendShutterBias
 - FliOcam2S, [156](#)
- sendTestPatternFromUrl
 - FliCredOne, [71](#)
- Sensor
 - FliCblueOneEnum, [11](#), [13](#)
 - FliCblueTwoEnum, [29](#)
- SensorDigitizationTaps
 - FliSfncCamera, [335](#)
- SensorHeight
 - FliSfncCamera, [335](#)
- SensorName
 - FliSfncCamera, [335](#)
- SensorPixelHeight
 - FliSfncCamera, [335](#)
- SensorPixelWidth
 - FliSfncCamera, [335](#)
- SensorShutterMode
 - FliSfncCamera, [335](#)
- SensorShutterModeEnum
 - FliCblueSfncEnum, [23](#)
- SensorShutterModeString
 - FliCblueSfncEnum, [28](#)
- SensorTaps
 - FliSfncCamera, [336](#)
- SensorWidth
 - FliSfncCamera, [336](#)
- SequencerConfigurationMode
 - FliSfncCamera, [336](#)
- SequencerFeatureEnable
 - FliSfncCamera, [336](#)
- SequencerFeatureSelector
 - FliSfncCamera, [336](#)
- SequencerMode
 - FliSfncCamera, [336](#)
- SequencerPathSelector
 - FliSfncCamera, [337](#)
- SequencerSetActive
 - FliSfncCamera, [337](#)
- SequencerSetLoad
 - FliSfncCamera, [337](#)
- SequencerSetNext
 - FliSfncCamera, [337](#)
- SequencerSetSave
 - FliSfncCamera, [337](#)
- SequencerSetSelector
 - FliSfncCamera, [337](#)
- SequencerSetStart
 - FliSfncCamera, [338](#)
- SequencerTriggerActivation
 - FliSfncCamera, [338](#)
- SequencerTriggerSource
 - FliSfncCamera, [338](#)
- serialCamera
 - FliSdk, [195](#)
- setAduOffset
 - FliCred, [58](#)
- setAgcParam
 - FliCredThree, [92](#)
- setAgcPriorityNone
 - FliCredThree, [92](#)
 - FliCredTwo, [121](#)
- setAgcPriorityOverExposed
 - FliCredThree, [93](#)

- FliCredTwo, [122](#)
- setAgcPriorityUnderExposed
 - FliCredThree, [93](#)
 - FliCredTwo, [122](#)
- setAgcRoi
 - FliCredThree, [93](#)
 - FliCredTwo, [122](#)
- setBadPixelModeOnOff
 - FliCred, [58](#)
- setBadPixelsCarto
 - ImageProcessing, [364](#)
 - ImageProcessing, [387](#)
- setBiasOffset
 - FliOcam2K, [152](#)
- setBooleanFeature
 - FliGenicamCamera, [145](#)
- setBufferSize
 - FliSdk, [195](#)
- setBufferSizeInImages
 - FliSdk, [195](#)
- setBurstFilter
 - FliSdk, [196](#)
- setCamera
 - FliSdk, [196](#)
 - ImageProcessing, [387](#)
- setCameraModel
 - ImageRingBuffer, [407](#)
- setClaheCliplimit
 - ImageProcessing, [364](#)
 - ImageProcessing, [387](#)
- setClaheTileGridSize
 - ImageProcessing, [364](#)
 - ImageProcessing, [387](#)
- setClipBlack
 - ImageProcessing, [364](#)
 - ImageProcessing, [388](#)
- setClipDepth
 - ImageProcessing, [365](#)
 - ImageProcessing, [388](#)
- setClipLimit
 - ImageProcessing, [365](#)
 - ImageProcessing, [388](#)
- setClippingAlpha
 - ImageProcessing, [365](#)
 - ImageProcessing, [388](#)
- setClippingBeta
 - ImageProcessing, [365](#)
 - ImageProcessing, [388](#)
- setClippingType
 - ImageProcessing, [365](#)
 - ImageProcessing, [389](#)
- setClipWhite
 - ImageProcessing, [366](#)
 - ImageProcessing, [389](#)
- setColorMapping
 - ImageProcessing, [366](#)
 - ImageProcessing, [389](#)
- setConversionGainHigh
 - FliCredThree, [93](#)
 - FliCredTwo, [122](#)
- setConversionGainLow
 - FliCredThree, [93](#)
 - FliCredTwo, [122](#)
- setConversionGainMedium
 - FliCredThree, [93](#)
 - FliCredTwo, [122](#)
- setCoolingFirstPoint
 - FliCredTwoLite, [134](#)
- setCoolingMode
 - FliCredTwoLite, [134](#)
- setCoolingState
 - FliCredTwoLite, [134](#)
- setCoolingStepWidth
 - FliCredTwoLite, [134](#)
- setCoolingValue
 - FliOcam2K, [153](#)
- setCropping
 - FliCredOne, [71](#)
 - FliCredThree, [93](#)
 - FliCredTwo, [122](#)
- setCroppingColumns
 - FliCredOne, [71](#)
 - FliCredThree, [94](#)
 - FliCredTwo, [123](#)
- setCroppingRows
 - FliCredOne, [71](#)
 - FliCredThree, [94](#)
 - FliCredTwo, [123](#)
- setCroppingState
 - FliSdk, [196](#)
- setCustomSerial
 - FliSerialCamera, [206](#)
- setDarkOptimLevel
 - FliCredThree, [94](#)
 - FliCredTwo, [123](#)
- setDefaultCapacity
 - ImageRingBuffer, [407](#)
- setDenoisingH
 - ImageProcessing, [366](#)
 - ImageProcessing, [389](#)
- setDenoisingSearchWindowSize
 - ImageProcessing, [366](#)
 - ImageProcessing, [389](#)
- setDenoisingTemplateWindowSize
 - ImageProcessing, [366](#)
 - ImageProcessing, [390](#)
- setDimension
 - ImageProcessing, [366](#)
 - ImageProcessing, [390](#)
- setDoubleFeature
 - FliGenicamCamera, [145](#)
- setDragoBias
 - ImageProcessing, [367](#)
 - ImageProcessing, [390](#)
- setDragoGamma
 - ImageProcessing, [367](#)

- ImageProcessing, 390
- setDragoMultiplier
 - ImageProcessing, 367
 - ImageProcessing, 390
- setDragoSaturation
 - ImageProcessing, 367
 - ImageProcessing, 391
- setExcludeBorderOnOff
 - FliCred, 58
- setExtSynchroExposureExternal
 - FliCredThree, 94
 - FliCredTwo, 123
- setExtSynchroExposureInternal
 - FliCredThree, 94
 - FliCredTwo, 123
- setExtSynchroPolarityInverted
 - FliCredThree, 94
 - FliCredTwo, 123
- setExtSynchroPolarityStandard
 - FliCredThree, 95
 - FliCredTwo, 124
- setFactoryBadPixelMap
 - FliCredThree, 95
 - FliCredTwo, 124
- setFanModeAutomatic
 - FliCredTwo, 124
- setFanModeManual
 - FliCredTwo, 124
- setFanSpeed
 - FliCredTwo, 124
- setFilteringModeOnOff
 - FliCred, 58
- setFowlerOffset
 - FliRingBuffer, 163
 - FliSdk, 197
 - ImageRingBuffer, 407
- setFps
 - FliOcam2K, 153
 - FliSerialCamera, 206
- setFpsMax
 - FliOcam2K, 153
- setFrameMarkerSourceCC1
 - FliCredThree, 95
 - FliCredTwo, 124
- setFrameMarkerSourceCC2
 - FliCredThree, 95
 - FliCredTwo, 124
- setFrameMarkerSourceCC3
 - FliCredThree, 95
 - FliCredTwo, 125
- setFrameMarkerSourceCC4
 - FliCredThree, 95
 - FliCredTwo, 125
- setFrameMarkerSourceExternal
 - FliCredThree, 95
 - FliCredTwo, 125
- setGain
 - FliCredOne, 72
 - FliOcam2K, 153
- setGamma
 - ImageProcessing, 367
 - ImageProcessing, 391
- setGrabber
 - FliSdk, 197
- setHdrCalibrationC1
 - FliCredThree, 96
 - FliCredTwo, 125
- setHdrCalibrationC2
 - FliCredThree, 96
 - FliCredTwo, 125
- setHdrCalibrationOff
 - FliCredThree, 96
 - FliCredTwo, 125
- setImageDimension
 - FliSdk, 197
 - ImageRingBuffer, 407
- setImageDimensionImageRingBuffer
 - FliSdk, 198
- setImageDimensionImageRingBufferThermo
 - FliSdk, 198
- setImageDimensionThermo
 - ImageRingBuffer, 408
- setImagePatternConstant
 - FliCredThree, 96
 - FliCredTwo, 125
- setImagePatternOff
 - FliCredThree, 96
 - FliCredTwo, 125
- setImagePatternRamp
 - FliCredThree, 96
 - FliCredTwo, 126
- setImageTagState
 - ImageRingBuffer, 408
- setIntegerFeature
 - FliGenicamCamera, 146
- setIpAddress
 - FliCred, 58
- setIpAlternateDns
 - FliCred, 58
- setIpAutomatic
 - FliCred, 59
- setIpDns
 - FliCred, 59
- setIpGateway
 - FliCred, 59
- setIpManual
 - FliCred, 59
- setIpNetmask
 - FliCred, 59
- setIpRefresh
 - FliCred, 59
- setIsThermoThrRaw
 - ImageProcessing, 368
 - ImageProcessing, 391
- setKindOfBadPixelCorrection
 - FliCred, 59

- setMantiukGamma
 - ImageProcessing, 368
 - ImageProcessing, 391
- setMantiukMultiplier
 - ImageProcessing, 368
 - ImageProcessing, 391
- setMantiukSaturation
 - ImageProcessing, 368
 - ImageProcessing, 391
- setMantiukScale
 - ImageProcessing, 368
 - ImageProcessing, 392
- setMode
 - FliCredOne, 72
 - FliSdk, 198
- setNbFramesPerSwTrig
 - FliCredThree, 96
 - FliCredTwo, 126
- setNbImagesAccumulation
 - ImageProcessing, 368
 - ImageProcessing, 392
- setNbImagesPerBuffer
 - FliSdk, 199
- setNbLoop
 - ImageRingBuffer, 408
- setNbRead
 - ImageRingBuffer, 408
- setNbReadImro
 - ImageRingBuffer, 408
- setNbReadWoReset
 - FliCredOne, 72
 - FliCredTwo, 126
- setNbSampPix
 - ImageRingBuffer, 408
- setNloop
 - FliCredOne, 72
- setNsamplePixel
 - FliCredOne, 72
- setNumThreads
 - ImageProcessing, 369
 - ImageProcessing, 392
- setObserverList
 - ImageRingBuffer, 409
- setOcamFrameNumberOffset
 - FliSdk, 199
 - ImageRingBuffer, 409
- setPassword
 - FliCred, 60
- setPercentOfMean
 - ImageProcessing, 369
 - ImageProcessing, 392
- setPixelSign
 - ImageProcessing, 369
 - ImageProcessing, 392
- setPreset
 - FliCredThree, 97
 - FliCredTwo, 126
- setPresetNumber
 - FliCredThree, 97
 - FliCredTwo, 126
- setRawData
 - FliGenicamCamera, 146
- setReinhardColorAdapt
 - ImageProcessing, 369
 - ImageProcessing, 392
- setReinhardGamma
 - ImageProcessing, 369
 - ImageProcessing, 393
- setReinhardIntensity
 - ImageProcessing, 369
 - ImageProcessing, 393
- setReinhardLightAdapt
 - ImageProcessing, 370
 - ImageProcessing, 393
- setResetWidth
 - FliCredOne, 72
- setRingBuffer
 - ImageProcessing, 393
- setRotationAngle
 - ImageProcessing, 370
 - ImageProcessing, 393
- setRotationAngleText
 - ImageProcessing, 370
 - ImageProcessing, 393
- setSensorSetpoint
 - FliCredTwoLite, 134
- setSensorTemp
 - FliCredTwo, 126
- setSharpenAlpha
 - ImageProcessing, 370
 - ImageProcessing, 394
- setSharpenBeta
 - ImageProcessing, 370
 - ImageProcessing, 394
- setSharpenGamma
 - ImageProcessing, 370
 - ImageProcessing, 394
- setSharpenKsize
 - ImageProcessing, 371
 - ImageProcessing, 394
- setSharpenSigmaX
 - ImageProcessing, 371
 - ImageProcessing, 394
- setSharpenSigmaY
 - ImageProcessing, 371
 - ImageProcessing, 395
- setShutterBlanking
 - FliOcam2S, 156
- setShutterBurst
 - FliOcam2S, 156
- setShutterEnd
 - FliOcam2S, 156
- setShutterExternal
 - FliOcam2S, 157
- setShutterInternal
 - FliOcam2S, 157

- setShutterPulseCount
 - FliOcam2S, 157
- setShutterPulsePosition
 - FliOcam2S, 157
- setShutterPulseWidth
 - FliOcam2S, 157
- setShutterSingle
 - FliOcam2S, 157
- setShutterStep
 - FliOcam2S, 157
- setShutterSweepMode
 - FliOcam2S, 158
- setSizeInFrames
 - FliRingBuffer, 163
 - ImageRingBuffer, 409
- setSizeInFramesThermo
 - FliRingBuffer, 163
 - ImageRingBuffer, 409
- setSizeInMo
 - FliRingBuffer, 164
 - ImageRingBuffer, 410
- setSlowMode
 - FliCred, 60
- setSnakeParam
 - FliCredThree, 97
 - FliCredTwo, 126
- setStdDevAndMeanSelection
 - ImageProcessing, 371
 - ImageProcessing, 395
- setStringFeature
 - FliGenicamCamera, 146
- setSyncDelay
 - FliCredThree, 97
 - FliCredTwo, 127
- setSynchronizationCmos
 - FliCredTwo, 127
- setSynchronizationFullCmos
 - FliCredTwo, 127
- setSynchronizationLvds
 - FliCredTwo, 127
- setSyncSignalSourceCC1
 - FliCredThree, 97
 - FliCredTwo, 127
- setSyncSignalSourceCC2
 - FliCredThree, 97
 - FliCredTwo, 127
- setSyncSignalSourceCC3
 - FliCredThree, 97
 - FliCredTwo, 127
- setSyncSignalSourceCC4
 - FliCredThree, 98
 - FliCredTwo, 128
- setSyncSignalSourceExternal
 - FliCredThree, 98
 - FliCredTwo, 128
- setThermoCalibrationData
 - ImageProcessing, 371
 - ImageProcessing, 395
- setThermoUnit
 - ImageProcessing, 372
 - ImageProcessing, 395
- setThresholdingLevelsValues
 - FliCred, 60
- setThresholdingOnOff
 - FliCred, 60
- setTint
 - FliCredThree, 98
 - FliCredTwo, 128
- setTlSyDel
 - FliCredThree, 98
 - FliCredTwo, 128
- setToneMappingDrago
 - ImageProcessing, 372
 - ImageProcessing, 395
- setToneMappingMantiuk
 - ImageProcessing, 372
 - ImageProcessing, 396
- setToneMappingNormal
 - ImageProcessing, 372
 - ImageProcessing, 396
- setToneMappingReinhard
 - ImageProcessing, 372
 - ImageProcessing, 396
- setTriggerSourceExternal
 - FliCredThree, 98
 - FliCredTwo, 128
- setTriggerSourceSoftware
 - FliCredThree, 98
 - FliCredTwo, 128
- setTuningGeneralUse
 - FliCredThree, 98
 - FliCredTwo, 128
- setTuningLongExposure
 - FliCredThree, 99
 - FliCredTwo, 129
- setTuningShortExposure
 - FliCredThree, 99
 - FliCredTwo, 129
- setupFixedCCsFrameGrabber
 - FliSdk, 199
- setUserBadPixelMap
 - FliCredThree, 99
 - FliCredTwo, 129
- setUserConvolutionMatrix
 - FliCred, 61
- setUserConvolutionMatrixIndex_V2
 - FliCred, 61
- setVoltageVref
 - FliCredThree, 99
 - FliCredTwo, 129
- setWorkMode
 - FliOcam2K, 153
- sfncCamera
 - FliSdk, 200
- shutDown
 - FliCred, 61

- Simulator
 - FliCblueOneEnum, [13](#)
- SimulatorGreyHorizontalRamp
 - FliCblueOneEnum, [13](#)
- SimulatorGreyHorizontalRampMoving
 - FliCblueOneEnum, [13](#)
- sleep
 - FliSerialCamera, [206](#)
- Soft
 - lImageProcessing, [354](#)
- SoftwareSignalPulse
 - FliSfncCamera, [338](#)
- SoftwareSignalSelector
 - FliSfncCamera, [338](#)
- softwareTrig
 - FliCredThree, [99](#)
 - FliCredTwo, [129](#)
- SourceCount
 - FliSfncCamera, [338](#)
- SourceIDValue
 - FliSfncCamera, [339](#)
- SourceSelector
 - FliSfncCamera, [339](#)
- Sparse
 - FliCblueOne, [41](#)
- SparseHeight
 - FliCblueOne, [42](#)
- SparseMode
 - FliCblueOne, [42](#)
- SparseModeEnum
 - FliCblueOneEnum, [12](#)
- SparseModeStringToValue
 - FliCblueOneEnum, [16](#)
- SparseOffsetX
 - FliCblueOne, [42](#)
- SparseOffsetY
 - FliCblueOne, [42](#)
- SparseSelector
 - FliCblueOne, [42](#)
- SparseSelectorEnum
 - FliCblueOneEnum, [12](#)
- SparseSelectorStringToValue
 - FliCblueOneEnum, [16](#)
- SparseWidth
 - FliCblueOne, [42](#)
- start
 - FliSdk, [200](#)
- startEthernetGrabber
 - FliCredThree, [99](#)
 - FliCredTwo, [129](#)
- startHttpServer
 - FliCredThree, [99](#)
 - FliCredTwo, [129](#)
- startVacuumRegen
 - FliCredOne, [72](#)
- StaticAlternateDomainNameServer
 - FliCblueOne, [43](#)
- StaticDefaultGateway
 - FliCblueOne, [43](#)
- StaticDomainNameServer
 - FliCblueOne, [43](#)
- StaticIPAddress
 - FliCblueOne, [43](#)
- StaticSubnetMask
 - FliCblueOne, [43](#)
- stop
 - FliSdk, [200](#)
- stopEthernetGrabber
 - FliCredThree, [100](#)
 - FliCredTwo, [130](#)
- stopHttpServer
 - FliCredThree, [100](#)
 - FliCredTwo, [130](#)
- Stream0
 - FliCblueSfncEnum, [23](#)
- StreamDataPacketCrc
 - FliCblueSfncEnum, [20](#)
- Sum
 - FliCblueTwoEnum, [29](#)
- TEC1
 - FliCblueOneEnum, [11](#)
- TestEventGenerate
 - FliSfncCamera, [339](#)
- TestPattern
 - FliCblueOne, [43](#)
 - FliSfncCamera, [339](#)
- TestPatternEnum
 - FliCblueOneEnum, [13](#)
- TestPatternGeneratorSelector
 - FliCblueOne, [44](#)
 - FliSfncCamera, [339](#)
- TestPatternGeneratorSelectorEnum
 - FliCblueOneEnum, [13](#)
- TestPatternGeneratorSelectorStringToValue
 - FliCblueOneEnum, [17](#)
- TestPatternStringToValue
 - FliCblueOneEnum, [17](#)
- TestPayloadFormatMode
 - FliSfncCamera, [339](#)
- TestPendingAck
 - FliSfncCamera, [340](#)
- ThermoUnit
 - lImageProcessing, [354](#)
- Timed
 - FliCblueSfncEnum, [22](#)
- TimerDelay
 - FliSfncCamera, [340](#)
- TimerDuration
 - FliSfncCamera, [340](#)
- TimerReset
 - FliSfncCamera, [340](#)
- TimerSelector
 - FliSfncCamera, [340](#)
- TimerStatus
 - FliSfncCamera, [340](#)
- TimerTriggerActivation

- FliSfncCamera, [341](#)
- TimerTriggerArmDelay
 - FliSfncCamera, [341](#)
- TimerTriggerSource
 - FliSfncCamera, [341](#)
- TimerValue
 - FliSfncCamera, [341](#)
- Timestamp
 - FliSfncCamera, [341](#)
- TimestampLatch
 - FliSfncCamera, [341](#)
- TimestampLatchValue
 - FliSfncCamera, [342](#)
- TimestampReset
 - FliSfncCamera, [342](#)
- TLParamsLocked
 - FliSfncCamera, [342](#)
- TLParamsLockedSelector
 - FliSfncCamera, [342](#)
- TLParamsLockedState
 - FliSfncCamera, [342](#)
- TransferAbort
 - FliSfncCamera, [342](#)
- TransferBlockCount
 - FliSfncCamera, [343](#)
- TransferBurstCount
 - FliSfncCamera, [343](#)
- TransferComponentSelector
 - FliSfncCamera, [343](#)
- TransferControlMode
 - FliSfncCamera, [343](#)
- TransferOperationMode
 - FliSfncCamera, [343](#)
- TransferPause
 - FliSfncCamera, [343](#)
- TransferQueueCurrentBlockCount
 - FliSfncCamera, [344](#)
- TransferQueueMaxBlockCount
 - FliSfncCamera, [344](#)
- TransferQueueMode
 - FliSfncCamera, [344](#)
- TransferResume
 - FliSfncCamera, [344](#)
- TransferSelector
 - FliSfncCamera, [344](#)
- TransferStart
 - FliSfncCamera, [344](#)
- TransferStatus
 - FliSfncCamera, [345](#)
- TransferStatusSelector
 - FliSfncCamera, [345](#)
- TransferStop
 - FliSfncCamera, [345](#)
- TransferStreamChannel
 - FliSfncCamera, [345](#)
- TransferTriggerActivation
 - FliSfncCamera, [345](#)
- TransferTriggerMode
 - FliSfncCamera, [345](#)
- TransferTriggerSelector
 - FliSfncCamera, [346](#)
- TransferTriggerSource
 - FliSfncCamera, [346](#)
- trigger_nb_frames_hg_to_mg
 - FliCredThree, [77](#)
- trigger_nb_frames_lg_to_mg
 - FliCredThree, [77](#)
- trigger_nb_frames_mg_to_hg
 - FliCredThree, [77](#)
- trigger_nb_frames_mg_to_lg
 - FliCredThree, [77](#)
- trigger_ratio_pixels_hg_to_mg
 - FliCredThree, [77](#)
- trigger_ratio_pixels_lg_to_mg
 - FliCredThree, [77](#)
- trigger_ratio_pixels_mg_to_hg
 - FliCredThree, [77](#)
- trigger_ratio_pixels_mg_to_lg
 - FliCredThree, [77](#)
- TriggerActivation
 - FliSfncCamera, [346](#)
- TriggerDelay
 - FliSfncCamera, [346](#)
- TriggerDivider
 - FliSfncCamera, [346](#)
- TriggerMode
 - FliSfncCamera, [346](#)
- TriggerMultiplier
 - FliSfncCamera, [347](#)
- TriggerOverlap
 - FliSfncCamera, [347](#)
- TriggerSelector
 - FliSfncCamera, [347](#)
- TriggerSoftware
 - FliSfncCamera, [347](#)
- TriggerSource
 - FliSfncCamera, [347](#)
- undefined
 - FliCredOne, [64](#)
- update
 - FliSdk, [200](#)
- updateAutoExposureParam
 - ImageProcessing, [372](#)
 - ImageProcessing, [396](#)
- upgradeFirmware
 - FliCred, [62](#)
- useDeprecatedFunction
 - IRawImageReceivedObserver, [412](#)
- UserOutputSelector
 - FliSfncCamera, [347](#)
- UserOutputValue
 - FliSfncCamera, [348](#)
- UserOutputValueAll
 - FliSfncCamera, [348](#)
- UserOutputValueAllMask
 - FliSfncCamera, [348](#)

- UserSet0
 - FliCblueOneEnum, [14](#)
- UserSet1
 - FliCblueOneEnum, [14](#)
- UserSet2
 - FliCblueOneEnum, [14](#)
- UserSet3
 - FliCblueOneEnum, [14](#)
- UserSet4
 - FliCblueOneEnum, [14](#)
- UserSet5
 - FliCblueOneEnum, [14](#)
- UserSet6
 - FliCblueOneEnum, [14](#)
- UserSet7
 - FliCblueOneEnum, [14](#)
- UserSet8
 - FliCblueOneEnum, [14](#)
- UserSet9
 - FliCblueOneEnum, [14](#)
- UserSetDefault
 - FliCblueOne, [44](#)
 - FliSfncCamera, [348](#)
- UserSetDefaultEnum
 - FliCblueOneEnum, [13](#)
- UserSetDefaultStringToValue
 - FliCblueOneEnum, [17](#)
- UserSetDescription
 - FliSfncCamera, [348](#)
- UserSetFeatureEnable
 - FliSfncCamera, [348](#)
- UserSetFeatureSelector
 - FliSfncCamera, [349](#)
- UserSetLoad
 - FliSfncCamera, [349](#)
- UserSetSave
 - FliSfncCamera, [349](#)
- UserSetSelector
 - FliCblueOne, [44](#)
 - FliSfncCamera, [349](#)
- UserSetSelectorEnum
 - FliCblueOneEnum, [14](#)
- UserSetSelectorStringToValue
 - FliCblueOneEnum, [17](#)
- version
 - FliSdk, [200](#)
- White
 - FliCblueOneEnum, [13](#)
- WhiteClip
 - FliSfncCamera, [349](#)
- WhiteClipSelector
 - FliSfncCamera, [349](#)
- Width
 - FliSfncCamera, [350](#)
- width
 - Ocam2Conf, [414](#)
- WidthMax
 - FliSfncCamera, [350](#)
- wmode
 - Ocam2Conf, [414](#)
- writeSerial
 - FliSerialCamera, [206](#)
- xSendBadPixelFile
 - FliCredThree, [100](#)
 - FliCredTwo, [130](#)
- xSendBiasFile
 - FliCredThree, [100](#)
 - FliCredTwo, [130](#)
- xSendBiasHdrC1File
 - FliCredThree, [100](#)
 - FliCredTwo, [130](#)
- xSendBiasHdrC2File
 - FliCredThree, [100](#)
 - FliCredTwo, [130](#)
- xSendFlatFile
 - FliCredThree, [100](#)
 - FliCredTwo, [130](#)
- xSendFlatHdrC1File
 - FliCredThree, [101](#)
 - FliCredTwo, [131](#)
- xSendFlatHdrC2File
 - FliCredThree, [101](#)
 - FliCredTwo, [131](#)
- xSendLicenseFile
 - FliCredThree, [101](#)
 - FliCredTwo, [131](#)