



FlyCapture 2.6

C Language API Programming Reference

Revised December 18, 2013

Point Grey Research Inc.

12051 Riverside Way • Richmond, BC • Canada • V6W 1K7 • T (604) 242-9937 •
www.ptgrey.com

Software Warranty

Point Grey Research warrants to the Original Purchaser, for a period of one (1) year from date of purchase that:

1. The diskette on which the Software is furnished and the accompanying documentation are not defective;
2. The Software is properly recorded upon the diskettes enclosed;
3. The documentation is substantially complete and contains all the information Point Grey Research deems necessary to use the Software;
4. The Software functions substantially as described in the documentation.

Point Grey Research, Inc.'s entire liability and the Original Purchaser's exclusive remedy shall be the replacement of any diskette or documentation not meeting these warranties. On such an occasion, a copy of the paid receipt accompanied with the faulty diskette or documentation must be returned to Point Grey Research, Inc. or an authorized dealer.

Point Grey Research, Inc. expressly disclaims and excludes all other warranties, express, implied and statutory, including, but without limitation, warranty of merchantability and fitness for a particular application or purpose. In no event shall Point Grey Research, Inc. be liable to the Original Purchaser or any third party for direct, indirect, incidental, consequential, special or accidental damages, including without limitation damages for business interruption, loss of profits, revenue, data or bodily injury or death.

Software License Agreement

READ CAREFULLY: This is a legal agreement between you (an individual or a single entity) ("you") and Point Grey Research, Inc. ("PGR"). Before installing and using the FlyCapture® Software Development Kit and any updates to it that we may at our discretion provide to you (collectively, the "SDK"), you should read this agreement. If you do not agree with all of the terms of this agreement, do not install or use the SDK. PGR may change this agreement at any time and it is your responsibility to review the most updated version of it on PGR's website at <http://www.ptgrey.com/support/kb/data/PGR-FlyCap-SDK-LA.pdf>. By continuing to use the SDK following such changes, you agree to be bound by them.

1. **Grant of License:** Subject to the terms of this agreement, you are hereby granted a limited, terminable, non-transferable, non-exclusive license and right to use the SDK only in conjunction with: (a) those PGR cameras listed at <http://www.ptgrey.com/products/index.asp> (as such list may be amended by PGR at any time and from time to time) and owned by you; and (b) the images derived from such cameras.
2. **Free and Open Source Components:** Notwithstanding anything to the contrary herein, use, copying and distribution of components of the SDK licensed under free and open source license agreements are governed solely by the terms of those license agreements (which are contained in the electronic documentation for the SDK) and not this agreement.
3. **Restrictions:** Except as (and only if) explicitly permitted by Section 4 below, you will not, and will not permit any third parties to: (a) copy the SDK, other than a reasonable number of backup copies for your own use only, and such backup copies together with the original will be kept in your possession and control; (b) provide or disclose the SDK to any third party; (c) alter, modify, reverse engineer, decompile or disassemble the SDK, or attempt to do any of the foregoing; (d) grant sublicenses, leases, or any other rights in the SDK to any third party; or (e) remove, alter or obscure any proprietary rights notices (including any copyright and trademark notices) on and in the SDK.
4. **Additional OEM Rights:** If you are an original equipment manufacturer, then in addition to the rights set out in Section 1 above you are hereby granted a limited, terminable, non-transferable, non-exclusive license and right to use the SDK for the sole additional purpose of incorporating the libraries found in the SDK (collectively, the "**Libraries**") into new products developed by you, in whole or in part, using the SDK (collectively, the "**Derivative Products**") provided that you: (a) ensure that the components of any Derivative Product that derive functionality from any of the Libraries may only be used with PGR products, including the SDK, and images derived from such products; (b) may only redistribute drivers (.inf and .sys), dynamically linked libraries (.dlls and .so), executables (.exe) and documentation (.doc, .txt, .pdf and .chm) and only to the extent necessary to support your Derivative Products. For clarity, headers (.h), source (.c, .cpp, .cs and .vb) and statically linked libraries (.lib and .a) cannot be redistributed; (c) will prohibit any, and ensure that there is no, redistribution of any of the Libraries by any third party, including any end user customers; and (d) will include any PGR and third party proprietary rights legends or notices (including copyright and trademark notices), unaltered and unobscured, on all Derivative Products.
5. **Ownership:** PGR and third parties are the owners of and retain title to all proprietary and intellectual property rights (including all patent, copyright, trade secret and trademark

rights) in and to the SDK. You have no right, title or interest in the SDK, except as specifically set forth herein, and no rights in any trade-marks of PGR. All rights not explicitly granted herein are hereby reserved.

6. **Indemnification:** You assume the entire risk relating to, and will indemnify, hold harmless and defend PGR from and against any claims, actions, lawsuits, or proceedings, and any losses, liabilities, damages and expenses (including attorney's fees and expenses) that arise or result from your activities under this agreement, including the distribution or use of the SDK (including the Libraries) and/or the development, distribution or use of any Derivative Product (including any intellectual property infringement claims relating thereto).

7. **No Warranties:** Your use of the SDK is solely at your own risk. The SDK is provided "as is" and "as available" without warranty or condition of any kind, either express, implied or statutory, including implied warranties of merchantability, fitness for a particular application or purpose, title and non-infringement, and PGR hereby expressly disclaims all such warranties and conditions. Some jurisdictions do not allow the exclusion of implied warranties, so the above exclusion may not apply to you.

8. **Limitation of Liability:** Notwithstanding any other provision herein, to the maximum extent permitted by applicable law, PGR will not under any circumstances be liable for any direct, indirect, incidental, consequential, punitive or special losses or damages (including damages for bodily injury or death, business interruption, loss or corruption of business information or data, loss of opportunity, loss of privacy, cost of recovery, financial loss, loss of revenue or loss of profits for any reason whatsoever) whether or not PGR has been advised or should have been aware of the possibility of any such losses or damage arising, and in no event will the aggregate and cumulative liability of PGR for any matters arising under this agreement or otherwise exceed \$25.

9. **Changes and Support:** PGR may at its sole discretion elect to provide you with limited support services related to the SDK for such period of time as PGR at its sole discretion elects. PGR may change or cease to provide the SDK and the limited Support at any time and from time to time without notice to you and that PGR is not obligated to provide the SDK or any support. PGR may use any technical information, feedback or ideas you provide to PGR for PGR's business purposes, including product support and development.

10. **Termination:** PGR may terminate this agreement immediately on notice to you if you violate any of the terms of this agreement. PGR may also terminate this agreement for convenience provided that 30 days notice is delivered to you. Any notice given by PGR hereunder will be effective when sent by PGR to the email address you provided to PGR when you registered for a download account. On termination, for any reason, your rights hereunder will cease and you must immediately return all copies of the SDK to PGR and on termination for violation or default, you will be liable to PGR for all damages suffered as a result of the violation or default. Sections 6-8, 10 and 12 will survive any termination hereof.

11. **Export Laws:** This agreement involves products and/or technical data that may be controlled under laws and regulations of the United States and other countries, including the United States Export Administration Regulations, or any other applicable law, regulation, rule, guideline or order (collectively, "**Export Laws**"). You will comply with all Export Laws to ensure that the SDK is not exported, directly or indirectly, in contravention of the Export Laws.

You represent and warrant to PGR that you are not a person barred from receiving the SDK under any Export Laws.

12. **General:** This agreement is the entire agreement between you and PGR with respect to the subject matter of this agreement. If you are signing on behalf of an entity such as a corporation, you represent and warrant that you have the authority to bind such entity. This agreement and the rights granted hereunder are personal to you and you may not assign this agreement to a third party without the prior written consent of PGR. This agreement is governed exclusively by and will be enforced, construed, and interpreted exclusively in accordance with the laws of British Columbia ("BC") and the laws of Canada applicable in BC. The courts of the Province of BC will have exclusive jurisdiction over any dispute arising under this agreement. You agree that termination and/or monetary damages may not be a sufficient remedy if you breach this agreement and that PGR will be entitled, without waiving any other rights or remedies, to injunctive or equitable relief as may be deemed proper by a court of competent jurisdiction in the event of a breach. If PGR does not exercise any legal right or remedy in this agreement or otherwise, this will not be taken to be a formal waiver by PGR of its rights, which rights will remain available to PGR. If any provision of this agreement is construed to be illegal or invalid, the illegal or invalid provisions will be deemed stricken and deleted herefrom to the same extent and effect as if never incorporated herein, but all other provisions hereof will continue in full force and effect.

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	fc2AVIOption Struct Reference	5
3.1.1	Field Documentation	5
3.1.1.1	frameRate	5
3.1.1.2	reserved	5
3.2	fc2CameraInfo Struct Reference	6
3.2.1	Field Documentation	7
3.2.1.1	applicationIPAddress	7
3.2.1.2	applicationPort	7
3.2.1.3	bayerTileFormat	7
3.2.1.4	busNumber	7
3.2.1.5	ccpStatus	7
3.2.1.6	configROM	7
3.2.1.7	defaultGateway	7
3.2.1.8	driverName	7
3.2.1.9	driverType	7
3.2.1.10	firmwareBuildTime	7
3.2.1.11	firmwareVersion	7
3.2.1.12	gigEMajorVersion	7
3.2.1.13	gigEMinorVersion	8

3.2.1.14	iidcVer	8
3.2.1.15	interfaceType	8
3.2.1.16	ipAddress	8
3.2.1.17	isColorCamera	8
3.2.1.18	macAddress	8
3.2.1.19	maximumBusSpeed	8
3.2.1.20	modelName	8
3.2.1.21	nodeNumber	8
3.2.1.22	pcieBusSpeed	8
3.2.1.23	reserved	8
3.2.1.24	sensorInfo	8
3.2.1.25	sensorResolution	8
3.2.1.26	serialNumber	8
3.2.1.27	subnetMask	8
3.2.1.28	userDefinedName	8
3.2.1.29	vendorName	8
3.2.1.30	xmlURL1	8
3.2.1.31	xmlURL2	8
3.3	fc2Config Struct Reference	8
3.3.1	Field Documentation	9
3.3.1.1	asyncBusSpeed	9
3.3.1.2	bandwidthAllocation	9
3.3.1.3	grabMode	9
3.3.1.4	grabTimeout	9
3.3.1.5	isochBusSpeed	9
3.3.1.6	minNumImageNotifications	9
3.3.1.7	numBuffers	9
3.3.1.8	numImageNotifications	9
3.3.1.9	registerTimeout	9
3.3.1.10	registerTimeoutRetries	9
3.3.1.11	reserved	9
3.4	fc2ConfigROM Struct Reference	9
3.4.1	Field Documentation	10
3.4.1.1	chipIdHi	10

3.4.1.2	chipIdLo	10
3.4.1.3	nodeVendorId	10
3.4.1.4	pszKeyword	10
3.4.1.5	reserved	10
3.4.1.6	unitSpecId	10
3.4.1.7	unitSubSWVer	10
3.4.1.8	unitSWVer	10
3.4.1.9	vendorUniqueInfo_0	10
3.4.1.10	vendorUniqueInfo_1	10
3.4.1.11	vendorUniqueInfo_2	10
3.4.1.12	vendorUniqueInfo_3	10
3.5	fc2EmbeddedImageInfo Struct Reference	11
3.5.1	Field Documentation	11
3.5.1.1	brightness	11
3.5.1.2	exposure	12
3.5.1.3	frameCounter	12
3.5.1.4	gain	12
3.5.1.5	GPIOPinState	12
3.5.1.6	ROIPosition	12
3.5.1.7	shutter	12
3.5.1.8	strobePattern	12
3.5.1.9	timestamp	12
3.5.1.10	whiteBalance	12
3.6	fc2EmbeddedImageInfoProperty Struct Reference	12
3.6.1	Field Documentation	12
3.6.1.1	available	12
3.6.1.2	onOff	12
3.7	fc2Format7ImageSettings Struct Reference	12
3.7.1	Field Documentation	13
3.7.1.1	height	13
3.7.1.2	mode	13
3.7.1.3	offsetX	13
3.7.1.4	offsetY	13
3.7.1.5	pixelFormat	13

3.7.1.6	reserved	13
3.7.1.7	width	13
3.8	fc2Format7Info Struct Reference	13
3.8.1	Field Documentation	14
3.8.1.1	imageHStepSize	14
3.8.1.2	imageVStepSize	14
3.8.1.3	maxHeight	14
3.8.1.4	maxPacketSize	14
3.8.1.5	maxWidth	14
3.8.1.6	minPacketSize	14
3.8.1.7	mode	14
3.8.1.8	offsetHStepSize	14
3.8.1.9	offsetVStepSize	14
3.8.1.10	packetSize	14
3.8.1.11	percentage	14
3.8.1.12	pixelFormatBitField	14
3.8.1.13	reserved	14
3.8.1.14	vendorPixelFormatBitField	14
3.9	fc2Format7PacketInfo Struct Reference	14
3.9.1	Field Documentation	14
3.9.1.1	maxBytesPerPacket	15
3.9.1.2	recommendedBytesPerPacket	15
3.9.1.3	reserved	15
3.9.1.4	unitBytesPerPacket	15
3.10	fc2GigEConfig Struct Reference	15
3.10.1	Field Documentation	15
3.10.1.1	enablePacketResend	15
3.10.1.2	maxPacketsToResend	15
3.10.1.3	reserved	15
3.10.1.4	timeoutForPacketResend	15
3.11	fc2GigEImageSettings Struct Reference	16
3.11.1	Field Documentation	16
3.11.1.1	height	16
3.11.1.2	offsetX	16

3.11.1.3	offsetY	16
3.11.1.4	pixelFormat	16
3.11.1.5	reserved	16
3.11.1.6	width	16
3.12	fc2GigElImageSettingsInfo Struct Reference	16
3.12.1	Field Documentation	17
3.12.1.1	imageHStepSize	17
3.12.1.2	imageVStepSize	17
3.12.1.3	maxHeight	17
3.12.1.4	maxWidth	17
3.12.1.5	offsetHStepSize	17
3.12.1.6	offsetVStepSize	17
3.12.1.7	pixelFormatBitField	17
3.12.1.8	reserved	17
3.12.1.9	vendorPixelFormatBitField	17
3.13	fc2GigEProperty Struct Reference	17
3.13.1	Field Documentation	17
3.13.1.1	isReadable	17
3.13.1.2	isWritable	17
3.13.1.3	max	17
3.13.1.4	min	17
3.13.1.5	propType	18
3.13.1.6	reserved	18
3.13.1.7	value	18
3.14	fc2GigEStreamChannel Struct Reference	18
3.14.1	Field Documentation	18
3.14.1.1	destinationIpAddress	18
3.14.1.2	doNotFragment	19
3.14.1.3	hostPort	19
3.14.1.4	interPacketDelay	19
3.14.1.5	networkInterfaceIndex	19
3.14.1.6	packetSize	19
3.14.1.7	reserved	19
3.14.1.8	sourcePort	19

3.15	fc2H264Option Struct Reference	19
3.15.1	Field Documentation	19
3.15.1.1	bitrate	19
3.15.1.2	frameRate	19
3.15.1.3	height	19
3.15.1.4	reserved	19
3.15.1.5	width	19
3.16	fc2Image Struct Reference	20
3.16.1	Field Documentation	20
3.16.1.1	bayerFormat	20
3.16.1.2	cols	20
3.16.1.3	dataSize	20
3.16.1.4	format	20
3.16.1.5	imageImpl	20
3.16.1.6	pData	20
3.16.1.7	receivedDataSize	20
3.16.1.8	rows	20
3.16.1.9	stride	20
3.17	fc2ImageMetadata Struct Reference	20
3.17.1	Field Documentation	21
3.17.1.1	embeddedBrightness	21
3.17.1.2	embeddedExposure	21
3.17.1.3	embeddedFrameCounter	21
3.17.1.4	embeddedGain	21
3.17.1.5	embeddedGPIOPinState	21
3.17.1.6	embeddedROIPosition	21
3.17.1.7	embeddedShutter	21
3.17.1.8	embeddedStrobePattern	21
3.17.1.9	embeddedTimeStamp	21
3.17.1.10	embeddedWhiteBalance	21
3.17.1.11	reserved	21
3.18	fc2ImageContext Struct Reference	21
3.18.1	Field Documentation	22
3.18.1.1	pBusMgr	22

3.18.1.2	pCamera	22
3.19	fc2InternalGuiContext Struct Reference	22
3.19.1	Field Documentation	22
3.19.1.1	pCameraControlDlg	22
3.19.1.2	pCameraSelectionDlg	22
3.20	fc2InternallImageCallback Struct Reference	23
3.20.1	Field Documentation	23
3.20.1.1	pCallback	23
3.20.1.2	pCallbackData	23
3.21	fc2IPAddress Struct Reference	23
3.21.1	Field Documentation	23
3.21.1.1	octets	24
3.22	fc2JPEGOption Struct Reference	24
3.22.1	Field Documentation	24
3.22.1.1	progressive	24
3.22.1.2	quality	24
3.22.1.3	reserved	24
3.23	fc2JPG2Option Struct Reference	24
3.23.1	Field Documentation	24
3.23.1.1	quality	24
3.23.1.2	reserved	24
3.24	fc2LUTData Struct Reference	25
3.24.1	Field Documentation	25
3.24.1.1	enabled	25
3.24.1.2	inputBitDepth	25
3.24.1.3	numBanks	25
3.24.1.4	numChannels	25
3.24.1.5	numEntries	25
3.24.1.6	outputBitDepth	25
3.24.1.7	reserved	25
3.24.1.8	supported	25
3.25	fc2MACAddress Struct Reference	25
3.25.1	Field Documentation	25
3.25.1.1	octets	26

3.26	fc2MJPGOption Struct Reference	26
3.26.1	Field Documentation	26
3.26.1.1	frameRate	26
3.26.1.2	quality	26
3.26.1.3	reserved	26
3.27	fc2PGMOption Struct Reference	26
3.27.1	Field Documentation	26
3.27.1.1	binaryFile	26
3.27.1.2	reserved	26
3.28	fc2PGRGuid Struct Reference	27
3.28.1	Detailed Description	27
3.28.2	Field Documentation	27
3.28.2.1	value	27
3.29	fc2PNGOption Struct Reference	27
3.29.1	Field Documentation	27
3.29.1.1	compressionLevel	27
3.29.1.2	interlaced	27
3.29.1.3	reserved	27
3.30	fc2PPMOption Struct Reference	28
3.30.1	Field Documentation	28
3.30.1.1	binaryFile	28
3.30.1.2	reserved	28
3.31	fc2StrobeControl Struct Reference	28
3.31.1	Field Documentation	28
3.31.1.1	delay	28
3.31.1.2	duration	28
3.31.1.3	onOff	28
3.31.1.4	polarity	28
3.31.1.5	reserved	28
3.31.1.6	source	28
3.32	fc2StrobeInfo Struct Reference	29
3.32.1	Field Documentation	29
3.32.1.1	maxValue	29
3.32.1.2	minValue	29

3.32.1.3	onOffSupported	29
3.32.1.4	polaritySupported	29
3.32.1.5	present	29
3.32.1.6	readOutSupported	29
3.32.1.7	reserved	29
3.32.1.8	source	29
3.33	fc2SystemInfo Struct Reference	29
3.33.1	Field Documentation	30
3.33.1.1	byteOrder	30
3.33.1.2	cpuDescription	30
3.33.1.3	driverList	30
3.33.1.4	gpuDescription	30
3.33.1.5	libraryList	30
3.33.1.6	numCpuCores	30
3.33.1.7	osDescription	30
3.33.1.8	osType	30
3.33.1.9	reserved	30
3.33.1.10	screenHeight	30
3.33.1.11	screenWidth	30
3.33.1.12	sysMemSize	30
3.34	fc2TIFFOption Struct Reference	30
3.34.1	Field Documentation	31
3.34.1.1	compression	31
3.34.1.2	reserved	31
3.35	fc2TimeStamp Struct Reference	31
3.35.1	Field Documentation	31
3.35.1.1	cycleCount	31
3.35.1.2	cycleOffset	31
3.35.1.3	cycleSeconds	31
3.35.1.4	microSeconds	31
3.35.1.5	reserved	31
3.35.1.6	seconds	31
3.36	fc2TriggerDelay Struct Reference	31
3.36.1	Field Documentation	32

3.36.1.1	absControl	32
3.36.1.2	absValue	32
3.36.1.3	autoManualMode	32
3.36.1.4	onePush	32
3.36.1.5	onOff	32
3.36.1.6	present	32
3.36.1.7	reserved	32
3.36.1.8	type	32
3.36.1.9	valueA	32
3.36.1.10	valueB	32
3.37	fc2TriggerDelayInfo Struct Reference	32
3.37.1	Field Documentation	33
3.37.1.1	absMax	33
3.37.1.2	absMin	33
3.37.1.3	absValSupported	33
3.37.1.4	autoSupported	33
3.37.1.5	manualSupported	33
3.37.1.6	max	33
3.37.1.7	min	33
3.37.1.8	onePushSupported	33
3.37.1.9	onOffSupported	33
3.37.1.10	present	33
3.37.1.11	pUnitAbbr	33
3.37.1.12	pUnits	33
3.37.1.13	readOutSupported	33
3.37.1.14	reserved	33
3.37.1.15	type	33
3.38	fc2TriggerMode Struct Reference	34
3.38.1	Field Documentation	34
3.38.1.1	mode	34
3.38.1.2	onOff	34
3.38.1.3	parameter	34
3.38.1.4	polarity	34
3.38.1.5	reserved	34

3.38.1.6	source	34
3.39	fc2TriggerModelInfo Struct Reference	34
3.39.1	Field Documentation	35
3.39.1.1	modeMask	35
3.39.1.2	onOffSupported	35
3.39.1.3	polaritySupported	35
3.39.1.4	present	35
3.39.1.5	readOutSupported	35
3.39.1.6	reserved	35
3.39.1.7	softwareTriggerSupported	35
3.39.1.8	sourceMask	35
3.39.1.9	valueReadable	35
3.40	fc2Version Struct Reference	35
3.40.1	Field Documentation	35
3.40.1.1	build	35
3.40.1.2	major	35
3.40.1.3	minor	35
3.40.1.4	type	35
4	File Documentation	37
4.1	FlyCapture2_C.h File Reference	37
4.1.1	Function Documentation	46
4.1.1.1	fc2AVIAppend	46
4.1.1.2	fc2AVIClose	46
4.1.1.3	fc2AVIOpen	47
4.1.1.4	fc2CalculateImageStatistics	47
4.1.1.5	fc2Connect	47
4.1.1.6	fc2ConvertImage	48
4.1.1.7	fc2ConvertImageTo	48
4.1.1.8	fc2CreateAVI	48
4.1.1.9	fc2CreateContext	49
4.1.1.10	fc2CreateGigEContext	49
4.1.1.11	fc2CreateImage	49
4.1.1.12	fc2CreateImageStatistics	50

4.1.1.13	fc2DestroyAVI	50
4.1.1.14	fc2DestroyContext	50
4.1.1.15	fc2DestroyImage	51
4.1.1.16	fc2DestroyImageStatistics	51
4.1.1.17	fc2DetermineBitsPerPixel	51
4.1.1.18	fc2Disconnect	51
4.1.1.19	fc2DiscoverGigECameras	52
4.1.1.20	fc2EnableLUT	52
4.1.1.21	fc2ErrorToDescription	52
4.1.1.22	fc2FireBusReset	53
4.1.1.23	fc2FireSoftwareTrigger	53
4.1.1.24	fc2FireSoftwareTriggerBroadcast	53
4.1.1.25	fc2ForceAllIPAddressesAutomatically	54
4.1.1.26	fc2ForceIPAddressAutomatically	54
4.1.1.27	fc2ForceIPAddressToCamera	54
4.1.1.28	fc2GetActiveLUTBank	55
4.1.1.29	fc2GetCameraFromIndex	55
4.1.1.30	fc2GetCameraFromSerialNumber	55
4.1.1.31	fc2GetCameraInfo	56
4.1.1.32	fc2GetCameraSerialNumberFromIndex	56
4.1.1.33	fc2GetChannelStatus	56
4.1.1.34	fc2GetConfiguration	57
4.1.1.35	fc2GetCycleTime	57
4.1.1.36	fc2GetDefaultColorProcessing	58
4.1.1.37	fc2GetDefaultOutputFormat	58
4.1.1.38	fc2GetDeviceFromIndex	58
4.1.1.39	fc2GetEmbeddedImageInfo	59
4.1.1.40	fc2GetFormat7Configuration	59
4.1.1.41	fc2GetFormat7Info	60
4.1.1.42	fc2GetGigEConfig	60
4.1.1.43	fc2GetGigEImageBinningSettings	60
4.1.1.44	fc2GetGigEImageSettings	60
4.1.1.45	fc2GetGigEImageSettingsInfo	60
4.1.1.46	fc2GetGigEImagingMode	60

4.1.1.47	fc2GetGigEProperty	60
4.1.1.48	fc2GetGigEStreamChannelInfo	61
4.1.1.49	fc2GetGPIOPinDirection	61
4.1.1.50	fc2GetImageData	61
4.1.1.51	fc2GetImageStatistics	62
4.1.1.52	fc2GetImageTimeStamp	62
4.1.1.53	fc2GetInterfaceTypeFromGuid	63
4.1.1.54	fc2GetLibraryVersion	63
4.1.1.55	fc2GetLUTBankInfo	63
4.1.1.56	fc2GetLUTChannel	64
4.1.1.57	fc2GetLUTInfo	64
4.1.1.58	fc2GetMemoryChannel	64
4.1.1.59	fc2GetMemoryChannelInfo	65
4.1.1.60	fc2GetNumOfCameras	65
4.1.1.61	fc2GetNumOfDevices	65
4.1.1.62	fc2GetNumStreamChannels	66
4.1.1.63	fc2GetProperty	66
4.1.1.64	fc2GetPropertyInfo	66
4.1.1.65	fc2GetRegisterString	66
4.1.1.66	fc2GetStrobe	67
4.1.1.67	fc2GetStrobeInfo	67
4.1.1.68	fc2GetSystemInfo	67
4.1.1.69	fc2GetTriggerDelay	68
4.1.1.70	fc2GetTriggerDelayInfo	68
4.1.1.71	fc2GetTriggerMode	68
4.1.1.72	fc2GetTriggerModeInfo	69
4.1.1.73	fc2GetVideoModeAndFrameRate	69
4.1.1.74	fc2GetVideoModeAndFrameRateInfo	69
4.1.1.75	fc2H264Open	70
4.1.1.76	fc2IsCameraControlable	70
4.1.1.77	fc2LaunchBrowser	70
4.1.1.78	fc2LaunchCommand	71
4.1.1.79	fc2LaunchCommandAsync	71
4.1.1.80	fc2LaunchHelp	71

4.1.1.81	fc2MJPgOpen	72
4.1.1.82	fc2QueryGigElmagingMode	72
4.1.1.83	fc2ReadGVCPMemory	72
4.1.1.84	fc2ReadGVCPRegister	72
4.1.1.85	fc2ReadGVCPRegisterBlock	73
4.1.1.86	fc2ReadRegister	73
4.1.1.87	fc2ReadRegisterBlock	73
4.1.1.88	fc2RegisterCallback	74
4.1.1.89	fc2RescanBus	74
4.1.1.90	fc2RestoreFromMemoryChannel	74
4.1.1.91	fc2RetrieveBuffer	75
4.1.1.92	fc2SavelImage	75
4.1.1.93	fc2SavelImageWithOptions	75
4.1.1.94	fc2SaveToMemoryChannel	76
4.1.1.95	fc2SetActiveLUTBank	76
4.1.1.96	fc2SetCallback	76
4.1.1.97	fc2SetChannelStatus	77
4.1.1.98	fc2SetConfiguration	77
4.1.1.99	fc2SetDefaultColorProcessing	77
4.1.1.100	fc2SetDefaultOutputFormat	78
4.1.1.101	fc2SetEmbeddedImageInfo	78
4.1.1.102	fc2SetFormat7Configuration	79
4.1.1.103	fc2SetFormat7ConfigurationPacket	79
4.1.1.104	fc2SetGigEConfig	79
4.1.1.105	fc2SetGigElmageBinningSettings	79
4.1.1.106	fc2SetGigElmageSettings	79
4.1.1.107	fc2SetGigElmagingMode	80
4.1.1.108	fc2SetGigEProperty	80
4.1.1.109	fc2SetGigEStreamChannelInfo	80
4.1.1.110	fc2SetGPIOPinDirection	80
4.1.1.111	fc2SetGPIOPinDirectionBroadcast	80
4.1.1.112	fc2SetImageData	81
4.1.1.113	fc2SetImageDimensions	81
4.1.1.114	fc2SetLUTChannel	82

4.1.1.115	fc2 SetProperty	82
4.1.1.116	fc2 SetPropertyBroadcast	82
4.1.1.117	fc2 SetStrobe	83
4.1.1.118	fc2 SetStrobeBroadcast	83
4.1.1.119	fc2 SetTriggerDelay	83
4.1.1.120	fc2 SetTriggerDelayBroadcast	84
4.1.1.121	fc2 SetTriggerMode	84
4.1.1.122	fc2 SetTriggerModeBroadcast	84
4.1.1.123	fc2 SetUserBuffers	85
4.1.1.124	fc2 SetVideoModeAndFrameRate	85
4.1.1.125	fc2 StartCapture	85
4.1.1.126	fc2 StartCaptureCallback	86
4.1.1.127	fc2 StartSyncCapture	86
4.1.1.128	fc2 StartSyncCaptureCallback	86
4.1.1.129	fc2 StopCapture	87
4.1.1.130	fc2 UnregisterCallback	87
4.1.1.131	fc2 ValidateFormat7Settings	87
4.1.1.132	fc2 WriteGVCPMemory	88
4.1.1.133	fc2 WriteGVCPRegister	88
4.1.1.134	fc2 WriteGVCPRegisterBlock	89
4.1.1.135	fc2 WriteGVCPRegisterBroadcast	89
4.1.1.136	fc2 WriteRegister	89
4.1.1.137	fc2 WriteRegisterBlock	90
4.1.1.138	fc2 WriteRegisterBroadcast	90
4.2	FlyCapture2Defs_C.h File Reference	90
4.2.1	Define Documentation	95
4.2.1.1	FALSE	95
4.2.1.2	FULL_32BIT_VALUE	95
4.2.1.3	MAX_STRING_LENGTH	95
4.2.1.4	TRUE	95
4.2.2	Typedef Documentation	95
4.2.2.1	BOOL	95
4.2.2.2	fc2AsyncCommandCallback	95
4.2.2.3	fc2AVIContext	95

4.2.2.4	fc2BusEventCallback	95
4.2.2.5	fc2CallbackHandle	95
4.2.2.6	fc2Context	95
4.2.2.7	fc2GuiContext	95
4.2.2.8	fc2ImageEventCallback	96
4.2.2.9	fc2ImageImpl	96
4.2.2.10	fc2ImageStatisticsContext	96
4.2.3	Enumeration Type Documentation	96
4.2.3.1	fc2BandwidthAllocation	96
4.2.3.2	fc2BayerTileFormat	96
4.2.3.3	fc2BusCallbackType	96
4.2.3.4	fc2BusSpeed	97
4.2.3.5	fc2ByteOrder	97
4.2.3.6	fc2ColorProcessingAlgorithm	97
4.2.3.7	fc2DriverType	98
4.2.3.8	fc2Error	98
4.2.3.9	fc2FrameRate	100
4.2.3.10	fc2GigEPropertyType	100
4.2.3.11	fc2GrabMode	100
4.2.3.12	fc2GrabTimeout	100
4.2.3.13	fc2ImageFileFormat	101
4.2.3.14	fc2InterfaceType	101
4.2.3.15	fc2Mode	101
4.2.3.16	fc2OSType	102
4.2.3.17	fc2PCleBusSpeed	103
4.2.3.18	fc2PixelFormat	103
4.2.3.19	fc2PropertyType	104
4.2.3.20	fc2StatisticsChannel	104
4.2.3.21	fc2TIFFCompressionMethod	105
4.2.3.22	fc2VideoMode	105
4.3	FlyCapture2GUI_C.h File Reference	106
4.3.1	Function Documentation	106
4.3.1.1	fc2CreateGUIContext	106
4.3.1.2	fc2DestroyGUIContext	106

4.3.1.3	fc2Disconnect	107
4.3.1.4	fc2GUIConnect	107
4.3.1.5	fc2Hide	107
4.3.1.6	fc2IsVisible	108
4.3.1.7	fc2Show	108
4.3.1.8	fc2ShowModal	108
4.4	FlyCapture2Internal_C.h File Reference	108
4.4.1	Function Documentation	109
4.4.1.1	IsContextValid	109
4.4.1.2	IsGuiContextValid	109
4.4.1.3	SyncCpplImageToStruct	109
4.5	FlyCapture2Platform_C.h File Reference	109
4.5.1	Define Documentation	109
4.5.1.1	FLYCAPTURE2_C_API	109
4.5.1.2	FLYCAPTURE2_C_CALL_CONVEN	109
4.6	MultiSyncLibrary_C.h File Reference	109
4.6.1	Function Documentation	110
4.6.1.1	syncCreateContext	110
4.6.1.2	syncDestroyContext	110
4.6.1.3	syncDisableCrossPCSSynchronization	111
4.6.1.4	syncEnableCrossPCSSynchronization	111
4.6.1.5	syncGetStatus	111
4.6.1.6	syncGetTimeSinceSynced	112
4.6.1.7	syncIsTimingBusConnected	112
4.6.1.8	syncQueryCrossPCSSynchronizationSetting	112
4.6.1.9	syncRescanMasterTimingBus	113
4.6.1.10	syncStart	113
4.6.1.11	syncStop	113
4.7	MultiSyncLibraryDefs_C.h File Reference	113
4.7.1	Define Documentation	114
4.7.1.1	FALSE	114
4.7.1.2	FULL_32BIT_VALUE	114
4.7.1.3	MAX_STRING_LENGTH	114
4.7.1.4	TRUE	114

4.7.2	Typedef Documentation	114
4.7.2.1	BOOL	114
4.7.2.2	syncContext	114
4.7.3	Enumeration Type Documentation	115
4.7.3.1	syncError	115
4.7.3.2	syncMessage	115
4.8	MultiSyncLibraryPlatform_C.h File Reference	115
4.8.1	Define Documentation	115
4.8.1.1	MULTISYNCLIBRARY_C_API	115
4.8.1.2	MULTISYNCLIBRARY_C_CALL_CONVEN	115

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

fc2AVIOption	5
fc2CameraInfo	6
fc2Config	8
fc2ConfigROM	9
fc2EmbeddedImageInfo	11
fc2EmbeddedImageInfoProperty	12
fc2Format7ImageSettings	12
fc2Format7Info	13
fc2Format7PacketInfo	14
fc2GigEConfig	15
fc2GigEImageSettings	16
fc2GigEImageSettingsInfo	16
fc2GigEProperty	17
fc2GigEStreamChannel	18
fc2H264Option	19
fc2Image	20
fc2ImageMetadata	20
fc2InternalContext	21
fc2InternalGuiContext	22
fc2InternalImageCallback	23
fc2IPAddress	23
fc2JPEGOption	24
fc2JPG2Option	24
fc2LUTData	25
fc2MACAddress	25
fc2MJPGOption	26
fc2PGMOption	26
fc2PGRGuid	
A GUID to the camera	27

fc2PNGOption	27
fc2PPMOption	28
fc2StrobeControl	28
fc2StrobeInfo	29
fc2SystemInfo	29
fc2TIFFOption	30
fc2TimeStamp	31
fc2TriggerDelay	31
fc2TriggerDelayInfo	32
fc2TriggerMode	34
fc2TriggerModeInfo	34
fc2Version	35

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

FlyCapture2_C.h	37
FlyCapture2Defs_C.h	90
FlyCapture2GUI_C.h	106
FlyCapture2Internal_C.h	108
FlyCapture2Platform_C.h	109
MultiSyncLibrary_C.h	109
MultiSyncLibraryDefs_C.h	113
MultiSyncLibraryPlatform_C.h	115

Chapter 3

Data Structure Documentation

3.1 fc2AVIOption Struct Reference

Data Fields

- float [frameRate](#)
- unsigned int [reserved](#) [256]

3.1.1 Field Documentation

3.1.1.1 float frameRate

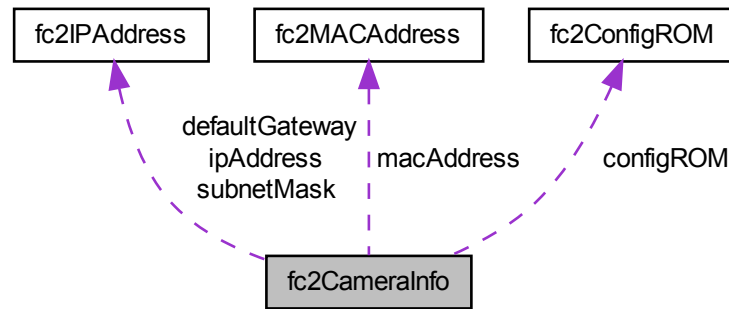
3.1.1.2 unsigned int reserved[256]

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

- [FlyCapture2Defs_C.h](#)

3.2 fc2CameraInfo Struct Reference

Collaboration diagram for fc2CameraInfo:



Data Fields

- unsigned int [serialNumber](#)
- [fc2InterfaceType](#) [interfaceType](#)
- [fc2DriverType](#) [driverType](#)
- [BOOL](#) [isColorCamera](#)
- char [modelName](#) [MAX_STRING_LENGTH]
- char [vendorName](#) [MAX_STRING_LENGTH]
- char [sensorInfo](#) [MAX_STRING_LENGTH]
- char [sensorResolution](#) [MAX_STRING_LENGTH]
- char [driverName](#) [MAX_STRING_LENGTH]
- char [firmwareVersion](#) [MAX_STRING_LENGTH]
- char [firmwareBuildTime](#) [MAX_STRING_LENGTH]
- [fc2BusSpeed](#) [maximumBusSpeed](#)
- [fc2PCleBusSpeed](#) [pcieBusSpeed](#)
- [fc2BayerTileFormat](#) [bayerTileFormat](#)
- unsigned short [busNumber](#)
- unsigned short [nodeNumber](#)
- unsigned int [iicVer](#)
- [fc2ConfigROM](#) [configROM](#)
- unsigned int [gigEMajorVersion](#)
- unsigned int [gigEMinorVersion](#)
- char [userDefinedName](#) [MAX_STRING_LENGTH]
- char [xmlURL1](#) [MAX_STRING_LENGTH]
- char [xmlURL2](#) [MAX_STRING_LENGTH]

- [fc2MACAddress](#) `macAddress`
- [fc2IPAddress](#) `ipAddress`
- [fc2IPAddress](#) `subnetMask`
- [fc2IPAddress](#) `defaultGateway`
- unsigned int [ccpStatus](#)
Status/Content of CCP register.
- unsigned int [applicationIPAddress](#)
Local Application IP Address.
- unsigned int [applicationPort](#)
Local Application port.
- unsigned int [reserved](#) [16]

3.2.1 Field Documentation

3.2.1.1 unsigned int `applicationIPAddress`

Local Application IP Address.

3.2.1.2 unsigned int `applicationPort`

Local Application port.

3.2.1.3 `fc2BayerTileFormat` `bayerTileFormat`

3.2.1.4 unsigned short `busNumber`

3.2.1.5 unsigned int `ccpStatus`

Status/Content of CCP register.

3.2.1.6 `fc2ConfigROM` `configROM`

3.2.1.7 `fc2IPAddress` `defaultGateway`

3.2.1.8 char `driverName`[MAX_STRING_LENGTH]

3.2.1.9 `fc2DriverType` `driverType`

3.2.1.10 char `firmwareBuildTime`[MAX_STRING_LENGTH]

3.2.1.11 char `firmwareVersion`[MAX_STRING_LENGTH]

3.2.1.12 unsigned int `gigEMajorVersion`

- 3.2.1.13 unsigned int `gigEMinorVersion`
- 3.2.1.14 unsigned int `iidcVer`
- 3.2.1.15 `fc2InterfaceType` `interfaceType`
- 3.2.1.16 `fc2IPAddress` `ipAddress`
- 3.2.1.17 `BOOL` `isColorCamera`
- 3.2.1.18 `fc2MACAddress` `macAddress`
- 3.2.1.19 `fc2BusSpeed` `maximumBusSpeed`
- 3.2.1.20 `char` `modelName`[MAX_STRING_LENGTH]
- 3.2.1.21 unsigned short `nodeNumber`
- 3.2.1.22 `fc2PCleBusSpeed` `pcieBusSpeed`
- 3.2.1.23 unsigned int `reserved`[16]
- 3.2.1.24 `char` `sensorInfo`[MAX_STRING_LENGTH]
- 3.2.1.25 `char` `sensorResolution`[MAX_STRING_LENGTH]
- 3.2.1.26 unsigned int `serialNumber`
- 3.2.1.27 `fc2IPAddress` `subnetMask`
- 3.2.1.28 `char` `userDefinedName`[MAX_STRING_LENGTH]
- 3.2.1.29 `char` `vendorName`[MAX_STRING_LENGTH]
- 3.2.1.30 `char` `xmiURL1`[MAX_STRING_LENGTH]
- 3.2.1.31 `char` `xmiURL2`[MAX_STRING_LENGTH]

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

- [FlyCapture2Defs_C.h](#)

3.3 `fc2Config` Struct Reference

Data Fields

- unsigned int `numBuffers`

- unsigned int [numImageNotifications](#)
- unsigned int [minNumImageNotifications](#)
- int [grabTimeout](#)
- [fc2GrabMode](#) [grabMode](#)
- [fc2BusSpeed](#) [isochBusSpeed](#)
- [fc2BusSpeed](#) [asyncBusSpeed](#)
- [fc2BandwidthAllocation](#) [bandwidthAllocation](#)
- unsigned int [registerTimeoutRetries](#)
- unsigned int [registerTimeout](#)
- unsigned int [reserved](#) [16]

3.3.1 Field Documentation

3.3.1.1 [fc2BusSpeed](#) [asyncBusSpeed](#)

3.3.1.2 [fc2BandwidthAllocation](#) [bandwidthAllocation](#)

3.3.1.3 [fc2GrabMode](#) [grabMode](#)

3.3.1.4 int [grabTimeout](#)

3.3.1.5 [fc2BusSpeed](#) [isochBusSpeed](#)

3.3.1.6 unsigned int [minNumImageNotifications](#)

3.3.1.7 unsigned int [numBuffers](#)

3.3.1.8 unsigned int [numImageNotifications](#)

3.3.1.9 unsigned int [registerTimeout](#)

3.3.1.10 unsigned int [registerTimeoutRetries](#)

3.3.1.11 unsigned int [reserved](#)[16]

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

- [FlyCapture2Defs_C.h](#)

3.4 fc2ConfigROM Struct Reference

Data Fields

- unsigned int [nodeVendorId](#)
- unsigned int [chipIdHi](#)

- unsigned int [chipIdLo](#)
- unsigned int [unitSpecId](#)
- unsigned int [unitSWVer](#)
- unsigned int [unitSubSWVer](#)
- unsigned int [vendorUniqueInfo_0](#)
- unsigned int [vendorUniqueInfo_1](#)
- unsigned int [vendorUniqueInfo_2](#)
- unsigned int [vendorUniqueInfo_3](#)
- char [pszKeyword](#) [MAX_STRING_LENGTH]
- unsigned int [reserved](#) [16]

3.4.1 Field Documentation

3.4.1.1 unsigned int [chipIdHi](#)

3.4.1.2 unsigned int [chipIdLo](#)

3.4.1.3 unsigned int [nodeVendorId](#)

3.4.1.4 char [pszKeyword](#)[MAX_STRING_LENGTH]

3.4.1.5 unsigned int [reserved](#)[16]

3.4.1.6 unsigned int [unitSpecId](#)

3.4.1.7 unsigned int [unitSubSWVer](#)

3.4.1.8 unsigned int [unitSWVer](#)

3.4.1.9 unsigned int [vendorUniqueInfo_0](#)

3.4.1.10 unsigned int [vendorUniqueInfo_1](#)

3.4.1.11 unsigned int [vendorUniqueInfo_2](#)

3.4.1.12 unsigned int [vendorUniqueInfo_3](#)

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

- [FlyCapture2Defs_C.h](#)

3.5 fc2EmbeddedImageInfo Struct Reference

Collaboration diagram for fc2EmbeddedImageInfo:



Data Fields

- [fc2EmbeddedImageInfoProperty timestamp](#)
- [fc2EmbeddedImageInfoProperty gain](#)
- [fc2EmbeddedImageInfoProperty shutter](#)
- [fc2EmbeddedImageInfoProperty brightness](#)
- [fc2EmbeddedImageInfoProperty exposure](#)
- [fc2EmbeddedImageInfoProperty whiteBalance](#)
- [fc2EmbeddedImageInfoProperty frameCounter](#)
- [fc2EmbeddedImageInfoProperty strobePattern](#)
- [fc2EmbeddedImageInfoProperty GPIOPinState](#)
- [fc2EmbeddedImageInfoProperty ROIPosition](#)

3.5.1 Field Documentation

3.5.1.1 fc2EmbeddedImageInfoProperty brightness

- 3.5.1.2 `fc2EmbeddedImageInfoProperty` exposure
- 3.5.1.3 `fc2EmbeddedImageInfoProperty` frameCounter
- 3.5.1.4 `fc2EmbeddedImageInfoProperty` gain
- 3.5.1.5 `fc2EmbeddedImageInfoProperty` GPIOPinState
- 3.5.1.6 `fc2EmbeddedImageInfoProperty` ROIPosition
- 3.5.1.7 `fc2EmbeddedImageInfoProperty` shutter
- 3.5.1.8 `fc2EmbeddedImageInfoProperty` strobePattern
- 3.5.1.9 `fc2EmbeddedImageInfoProperty` timestamp
- 3.5.1.10 `fc2EmbeddedImageInfoProperty` whiteBalance

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

- [FlyCapture2Defs_C.h](#)

3.6 `fc2EmbeddedImageInfoProperty` Struct Reference

Data Fields

- [BOOL](#) available
- [BOOL](#) onOff

3.6.1 Field Documentation

- 3.6.1.1 **BOOL** available
- 3.6.1.2 **BOOL** onOff

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

- [FlyCapture2Defs_C.h](#)

3.7 `fc2Format7ImageSettings` Struct Reference

Data Fields

- [fc2Mode](#) mode

- unsigned int [offsetX](#)
- unsigned int [offsetY](#)
- unsigned int [width](#)
- unsigned int [height](#)
- [fc2PixelFormat](#) [pixelFormat](#)
- unsigned int [reserved](#) [8]

3.7.1 Field Documentation

3.7.1.1 unsigned int [height](#)

3.7.1.2 [fc2Mode](#) [mode](#)

3.7.1.3 unsigned int [offsetX](#)

3.7.1.4 unsigned int [offsetY](#)

3.7.1.5 [fc2PixelFormat](#) [pixelFormat](#)

3.7.1.6 unsigned int [reserved](#)[8]

3.7.1.7 unsigned int [width](#)

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

- [FlyCapture2Defs_C.h](#)

3.8 fc2Format7Info Struct Reference

Data Fields

- [fc2Mode](#) [mode](#)
- unsigned int [maxWidth](#)
- unsigned int [maxHeight](#)
- unsigned int [offsetHStepSize](#)
- unsigned int [offsetVStepSize](#)
- unsigned int [imageHStepSize](#)
- unsigned int [imageVStepSize](#)
- unsigned int [pixelFormatBitField](#)
- unsigned int [vendorPixelFormatBitField](#)
- unsigned int [packetSize](#)
- unsigned int [minPacketSize](#)
- unsigned int [maxPacketSize](#)
- float [percentage](#)
- unsigned int [reserved](#) [16]

3.8.1 Field Documentation

3.8.1.1 unsigned int imageHStepSize

3.8.1.2 unsigned int imageVStepSize

3.8.1.3 unsigned int maxHeight

3.8.1.4 unsigned int maxPacketSize

3.8.1.5 unsigned int maxWidth

3.8.1.6 unsigned int minPacketSize

3.8.1.7 fc2Mode mode

3.8.1.8 unsigned int offsetHStepSize

3.8.1.9 unsigned int offsetVStepSize

3.8.1.10 unsigned int packetSize

3.8.1.11 float percentage

3.8.1.12 unsigned int pixelFormatBitField

3.8.1.13 unsigned int reserved[16]

3.8.1.14 unsigned int vendorPixelFormatBitField

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

- [FlyCapture2Defs_C.h](#)

3.9 fc2Format7PacketInfo Struct Reference

Data Fields

- unsigned int [recommendedBytesPerPacket](#)
- unsigned int [maxBytesPerPacket](#)
- unsigned int [unitBytesPerPacket](#)
- unsigned int [reserved](#) [8]

3.9.1 Field Documentation

3.9.1.1 unsigned int `maxBytesPerPacket`

3.9.1.2 unsigned int `recommendedBytesPerPacket`

3.9.1.3 unsigned int `reserved`[8]

3.9.1.4 unsigned int `unitBytesPerPacket`

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

- [FlyCapture2Defs_C.h](#)

3.10 fc2GigEConfig Struct Reference

Data Fields

- [BOOL](#) `enablePacketResend`
Turn on/off packet resend functionality.
- unsigned int [timeoutForPacketResend](#)
The number of milliseconds to wait for each requested packet.
- unsigned int [maxPacketsToResend](#)
The max number of packets that can be requested to be resend.
- unsigned int [reserved](#) [8]

3.10.1 Field Documentation

3.10.1.1 **BOOL** `enablePacketResend`

Turn on/off packet resend functionality.

3.10.1.2 unsigned int `maxPacketsToResend`

The max number of packets that can be requested to be resend.

3.10.1.3 unsigned int `reserved`[8]

3.10.1.4 unsigned int `timeoutForPacketResend`

The number of milliseconds to wait for each requested packet.

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

- [FlyCapture2Defs_C.h](#)

3.11 fc2GigEImageSettings Struct Reference

Data Fields

- unsigned int [offsetX](#)
- unsigned int [offsetY](#)
- unsigned int [width](#)
- unsigned int [height](#)
- [fc2PixelFormat](#) [pixelFormat](#)
- unsigned int [reserved](#) [8]

3.11.1 Field Documentation

3.11.1.1 unsigned int [height](#)

3.11.1.2 unsigned int [offsetX](#)

3.11.1.3 unsigned int [offsetY](#)

3.11.1.4 [fc2PixelFormat](#) [pixelFormat](#)

3.11.1.5 unsigned int [reserved](#)[8]

3.11.1.6 unsigned int [width](#)

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

- [FlyCapture2Defs_C.h](#)

3.12 fc2GigEImageSettingsInfo Struct Reference

Data Fields

- unsigned int [maxWidth](#)
- unsigned int [maxHeight](#)
- unsigned int [offsetHStepSize](#)
- unsigned int [offsetVStepSize](#)
- unsigned int [imageHStepSize](#)
- unsigned int [imageVStepSize](#)
- unsigned int [pixelFormatBitField](#)
- unsigned int [vendorPixelFormatBitField](#)
- unsigned int [reserved](#) [16]

3.12.1 Field Documentation

3.12.1.1 unsigned int imageHStepSize

3.12.1.2 unsigned int imageVStepSize

3.12.1.3 unsigned int maxHeight

3.12.1.4 unsigned int maxWidth

3.12.1.5 unsigned int offsetHStepSize

3.12.1.6 unsigned int offsetVStepSize

3.12.1.7 unsigned int pixelFormatBitField

3.12.1.8 unsigned int reserved[16]

3.12.1.9 unsigned int vendorPixelFormatBitField

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

- [FlyCapture2Defs_C.h](#)

3.13 fc2GigEProperty Struct Reference

Data Fields

- [fc2GigEPropertyType propType](#)
- [BOOL isReadable](#)
- [BOOL isWritable](#)
- unsigned int [min](#)
- unsigned int [max](#)
- unsigned int [value](#)
- unsigned int [reserved](#) [8]

3.13.1 Field Documentation

3.13.1.1 **BOOL isReadable**

3.13.1.2 **BOOL isWritable**

3.13.1.3 unsigned int max

3.13.1.4 unsigned int min

3.13.1.5 fc2GigEPropertyType propType

3.13.1.6 unsigned int reserved[8]

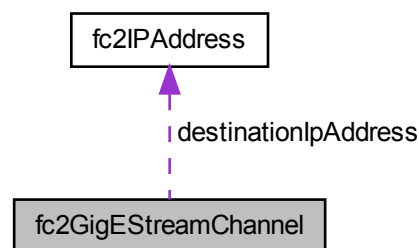
3.13.1.7 unsigned int value

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

- [FlyCapture2Defs_C.h](#)

3.14 fc2GigEStreamChannel Struct Reference

Collaboration diagram for fc2GigEStreamChannel:



Data Fields

- unsigned int [networkInterfaceIndex](#)
- unsigned int [hostPost](#)
- [BOOL](#) [doNotFragment](#)
- unsigned int [packetSize](#)
- unsigned int [interPacketDelay](#)
- [fc2IPAddress](#) [destinationIpAddress](#)
- unsigned int [sourcePort](#)
- unsigned int [reserved](#) [8]

3.14.1 Field Documentation

3.14.1.1 fc2IPAddress destinationIpAddress

3.14.1.2 **BOOL doNotFragment**

3.14.1.3 **unsigned int hostPost**

3.14.1.4 **unsigned int interPacketDelay**

3.14.1.5 **unsigned int networkInterfaceIndex**

3.14.1.6 **unsigned int packetSize**

3.14.1.7 **unsigned int reserved[8]**

3.14.1.8 **unsigned int sourcePort**

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

- [FlyCapture2Defs_C.h](#)

3.15 fc2H264Option Struct Reference

Data Fields

- float [frameRate](#)
- unsigned int [width](#)
- unsigned int [height](#)
- unsigned int [bitrate](#)
- unsigned int [reserved](#) [256]

3.15.1 Field Documentation

3.15.1.1 **unsigned int bitrate**

3.15.1.2 **float frameRate**

3.15.1.3 **unsigned int height**

3.15.1.4 **unsigned int reserved[256]**

3.15.1.5 **unsigned int width**

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

- [FlyCapture2Defs_C.h](#)

3.16 fc2Image Struct Reference

Data Fields

- unsigned int [rows](#)
- unsigned int [cols](#)
- unsigned int [stride](#)
- unsigned char * [pData](#)
- unsigned int [dataSize](#)
- unsigned int [receivedDataSize](#)
- [fc2PixelFormat](#) format
- [fc2BayerTileFormat](#) bayerFormat
- [fc2ImageImpl](#) imageImpl

3.16.1 Field Documentation

3.16.1.1 [fc2BayerTileFormat](#) bayerFormat

3.16.1.2 unsigned int cols

3.16.1.3 unsigned int dataSize

3.16.1.4 [fc2PixelFormat](#) format

3.16.1.5 [fc2ImageImpl](#) imageImpl

3.16.1.6 unsigned char* pData

3.16.1.7 unsigned int receivedDataSize

3.16.1.8 unsigned int rows

3.16.1.9 unsigned int stride

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

- [FlyCapture2Defs_C.h](#)

3.17 fc2ImageMetadata Struct Reference

Data Fields

- unsigned int [embeddedTimeStamp](#)
- unsigned int [embeddedGain](#)
- unsigned int [embeddedShutter](#)

- unsigned int [embeddedBrightness](#)
- unsigned int [embeddedExposure](#)
- unsigned int [embeddedWhiteBalance](#)
- unsigned int [embeddedFrameCounter](#)
- unsigned int [embeddedStrobePattern](#)
- unsigned int [embeddedGPIOPinState](#)
- unsigned int [embeddedROIPosition](#)
- unsigned int [reserved](#) [31]

3.17.1 Field Documentation

- 3.17.1.1 unsigned int [embeddedBrightness](#)
- 3.17.1.2 unsigned int [embeddedExposure](#)
- 3.17.1.3 unsigned int [embeddedFrameCounter](#)
- 3.17.1.4 unsigned int [embeddedGain](#)
- 3.17.1.5 unsigned int [embeddedGPIOPinState](#)
- 3.17.1.6 unsigned int [embeddedROIPosition](#)
- 3.17.1.7 unsigned int [embeddedShutter](#)
- 3.17.1.8 unsigned int [embeddedStrobePattern](#)
- 3.17.1.9 unsigned int [embeddedTimeStamp](#)
- 3.17.1.10 unsigned int [embeddedWhiteBalance](#)
- 3.17.1.11 unsigned int [reserved](#)[31]

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

- [FlyCapture2Defs_C.h](#)

3.18 fc2InternalContext Struct Reference

Data Fields

- FlyCapture2::BusManager * [pBusMgr](#)
- FlyCapture2::CameraBase * [pCamera](#)

3.18.1 Field Documentation

3.18.1.1 FlyCapture2::BusManager* pBusMgr

3.18.1.2 FlyCapture2::CameraBase* pCamera

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

- [FlyCapture2Internal_C.h](#)

3.19 fc2InternalGuiContext Struct Reference

Data Fields

- FlyCapture2::CameraSelectionDlg * [pCameraSelectionDlg](#)
- FlyCapture2::CameraControlDlg * [pCameraControlDlg](#)

3.19.1 Field Documentation

3.19.1.1 FlyCapture2::CameraControlDlg* pCameraControlDlg

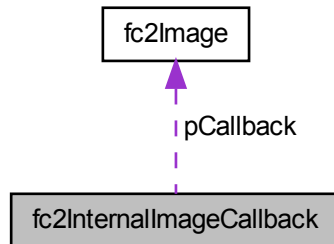
3.19.1.2 FlyCapture2::CameraSelectionDlg* pCameraSelectionDlg

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

- [FlyCapture2Internal_C.h](#)

3.20 fc2InternalImageCallback Struct Reference

Collaboration diagram for fc2InternalImageCallback:



Data Fields

- [fc2ImageEventCallback](#) pCallback
- void * [pCallbackData](#)

3.20.1 Field Documentation

3.20.1.1 [fc2ImageEventCallback](#) pCallback

3.20.1.2 void* [pCallbackData](#)

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

- [FlyCapture2Internal_C.h](#)

3.21 fc2IPAddress Struct Reference

Data Fields

- unsigned char [octets](#) [4]

3.21.1 Field Documentation

3.21.1.1 unsigned char octets[4]

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

- [FlyCapture2Defs_C.h](#)

3.22 fc2JPEGOption Struct Reference

Data Fields

- [BOOL progressive](#)
- unsigned int [quality](#)
- unsigned int [reserved](#) [16]

3.22.1 Field Documentation

3.22.1.1 BOOL progressive

3.22.1.2 unsigned int quality

3.22.1.3 unsigned int reserved[16]

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

- [FlyCapture2Defs_C.h](#)

3.23 fc2JPG2Option Struct Reference

Data Fields

- unsigned int [quality](#)
- unsigned int [reserved](#) [16]

3.23.1 Field Documentation

3.23.1.1 unsigned int quality

3.23.1.2 unsigned int reserved[16]

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

- [FlyCapture2Defs_C.h](#)

3.24 fc2LUTData Struct Reference

Data Fields

- [BOOL supported](#)
- [BOOL enabled](#)
- unsigned int [numBanks](#)
- unsigned int [numChannels](#)
- unsigned int [inputBitDepth](#)
- unsigned int [outputBitDepth](#)
- unsigned int [numEntries](#)
- unsigned int [reserved](#) [8]

3.24.1 Field Documentation

3.24.1.1 [BOOL enabled](#)

3.24.1.2 [unsigned int inputBitDepth](#)

3.24.1.3 [unsigned int numBanks](#)

3.24.1.4 [unsigned int numChannels](#)

3.24.1.5 [unsigned int numEntries](#)

3.24.1.6 [unsigned int outputBitDepth](#)

3.24.1.7 [unsigned int reserved\[8\]](#)

3.24.1.8 [BOOL supported](#)

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

- [FlyCapture2Defs_C.h](#)

3.25 fc2MACAddress Struct Reference

Data Fields

- unsigned char [octets](#) [6]

3.25.1 Field Documentation

3.25.1.1 unsigned char octets[6]

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

- [FlyCapture2Defs_C.h](#)

3.26 fc2MJPGOption Struct Reference

Data Fields

- float [frameRate](#)
- unsigned int [quality](#)
- unsigned int [reserved](#) [256]

3.26.1 Field Documentation

3.26.1.1 float frameRate

3.26.1.2 unsigned int quality

3.26.1.3 unsigned int reserved[256]

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

- [FlyCapture2Defs_C.h](#)

3.27 fc2PGMOption Struct Reference

Data Fields

- [BOOL](#) [binaryFile](#)
- unsigned int [reserved](#) [16]

3.27.1 Field Documentation

3.27.1.1 [BOOL](#) [binaryFile](#)

3.27.1.2 unsigned int [reserved](#)[16]

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

- [FlyCapture2Defs_C.h](#)

3.28 fc2PGRGuid Struct Reference

A GUID to the camera.

Data Fields

- unsigned int [value](#) [4]

3.28.1 Detailed Description

A GUID to the camera.

It is used to uniquely identify a camera.

3.28.2 Field Documentation

3.28.2.1 unsigned int [value](#)[4]

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

- [FlyCapture2Defs_C.h](#)

3.29 fc2PNGOption Struct Reference

Data Fields

- [BOOL](#) [interlaced](#)
- unsigned int [compressionLevel](#)
- unsigned int [reserved](#) [16]

3.29.1 Field Documentation

3.29.1.1 unsigned int [compressionLevel](#)

3.29.1.2 [BOOL](#) [interlaced](#)

3.29.1.3 unsigned int [reserved](#)[16]

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

- [FlyCapture2Defs_C.h](#)

3.30 fc2PPMOption Struct Reference

Data Fields

- [BOOL](#) `binaryFile`
- unsigned int [reserved](#) [16]

3.30.1 Field Documentation

3.30.1.1 [BOOL](#) `binaryFile`

3.30.1.2 unsigned int [reserved](#)[16]

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

- [FlyCapture2Defs_C.h](#)

3.31 fc2StrobeControl Struct Reference

Data Fields

- unsigned int [source](#)
- [BOOL](#) `onOff`
- unsigned int [polarity](#)
- float [delay](#)
- float [duration](#)
- unsigned int [reserved](#) [8]

3.31.1 Field Documentation

3.31.1.1 float [delay](#)

3.31.1.2 float [duration](#)

3.31.1.3 [BOOL](#) `onOff`

3.31.1.4 unsigned int [polarity](#)

3.31.1.5 unsigned int [reserved](#)[8]

3.31.1.6 unsigned int [source](#)

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

- [FlyCapture2Defs_C.h](#)

3.32 fc2StrobeInfo Struct Reference

Data Fields

- unsigned int [source](#)
- **BOOL** [present](#)
- **BOOL** [readOutSupported](#)
- **BOOL** [onOffSupported](#)
- **BOOL** [polaritySupported](#)
- float [minValue](#)
- float [maxValue](#)
- unsigned int [reserved](#) [8]

3.32.1 Field Documentation

3.32.1.1 float [maxValue](#)

3.32.1.2 float [minValue](#)

3.32.1.3 **BOOL** [onOffSupported](#)

3.32.1.4 **BOOL** [polaritySupported](#)

3.32.1.5 **BOOL** [present](#)

3.32.1.6 **BOOL** [readOutSupported](#)

3.32.1.7 unsigned int [reserved](#)[8]

3.32.1.8 unsigned int [source](#)

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

- [FlyCapture2Defs_C.h](#)

3.33 fc2SystemInfo Struct Reference

Data Fields

- [fc2OSType](#) [osType](#)
- char [osDescription](#) [MAX_STRING_LENGTH]
- [fc2ByteOrder](#) [byteOrder](#)
- size_t [sysMemSize](#)
- char [cpuDescription](#) [MAX_STRING_LENGTH]
- size_t [numCpuCores](#)

- char [driverList](#) [MAX_STRING_LENGTH]
- char [libraryList](#) [MAX_STRING_LENGTH]
- char [gpuDescription](#) [MAX_STRING_LENGTH]
- size_t [screenWidth](#)
- size_t [screenHeight](#)
- unsigned int [reserved](#) [16]

3.33.1 Field Documentation

3.33.1.1 **fc2ByteOrder** `byteOrder`

3.33.1.2 **char** `cpuDescription`[MAX_STRING_LENGTH]

3.33.1.3 **char** `driverList`[MAX_STRING_LENGTH]

3.33.1.4 **char** `gpuDescription`[MAX_STRING_LENGTH]

3.33.1.5 **char** `libraryList`[MAX_STRING_LENGTH]

3.33.1.6 **size_t** `numCpuCores`

3.33.1.7 **char** `osDescription`[MAX_STRING_LENGTH]

3.33.1.8 **fc2OSType** `osType`

3.33.1.9 **unsigned int** `reserved`[16]

3.33.1.10 **size_t** `screenHeight`

3.33.1.11 **size_t** `screenWidth`

3.33.1.12 **size_t** `sysMemSize`

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

- [FlyCapture2Defs_C.h](#)

3.34 fc2TIFFOption Struct Reference

Data Fields

- [fc2TIFFCompressionMethod](#) `compression`
- unsigned int [reserved](#) [16]

3.34.1 Field Documentation

3.34.1.1 fc2TIFFCompressionMethod compression

3.34.1.2 unsigned int reserved[16]

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

- [FlyCapture2Defs_C.h](#)

3.35 fc2TimeStamp Struct Reference

Data Fields

- long long [seconds](#)
- unsigned int [microSeconds](#)
- unsigned int [cycleSeconds](#)
- unsigned int [cycleCount](#)
- unsigned int [cycleOffset](#)
- unsigned int [reserved](#) [8]

3.35.1 Field Documentation

3.35.1.1 unsigned int cycleCount

3.35.1.2 unsigned int cycleOffset

3.35.1.3 unsigned int cycleSeconds

3.35.1.4 unsigned int microSeconds

3.35.1.5 unsigned int reserved[8]

3.35.1.6 long long seconds

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

- [FlyCapture2Defs_C.h](#)

3.36 fc2TriggerDelay Struct Reference

Data Fields

- [fc2PropertyType](#) type

- [BOOL present](#)
- [BOOL absControl](#)
- [BOOL onePush](#)
- [BOOL onOff](#)
- [BOOL autoManualMode](#)
- unsigned int [valueA](#)
- unsigned int [valueB](#)
- float [absValue](#)
- unsigned int [reserved](#) [8]

3.36.1 Field Documentation

3.36.1.1 BOOL absControl

3.36.1.2 float absValue

3.36.1.3 BOOL autoManualMode

3.36.1.4 BOOL onePush

3.36.1.5 BOOL onOff

3.36.1.6 BOOL present

3.36.1.7 unsigned int reserved[8]

3.36.1.8 fc2PropertyType type

3.36.1.9 unsigned int valueA

3.36.1.10 unsigned int valueB

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

- [FlyCapture2Defs_C.h](#)

3.37 fc2TriggerDelayInfo Struct Reference

Data Fields

- [fc2PropertyType type](#)
- [BOOL present](#)
- [BOOL autoSupported](#)
- [BOOL manualSupported](#)
- [BOOL onOffSupported](#)

- [BOOL onePushSupported](#)
- [BOOL absValSupported](#)
- [BOOL readOutSupported](#)
- unsigned int [min](#)
- unsigned int [max](#)
- float [absMin](#)
- float [absMax](#)
- char [pUnits](#) [MAX_STRING_LENGTH]
- char [pUnitAbbr](#) [MAX_STRING_LENGTH]
- unsigned int [reserved](#) [8]

3.37.1 Field Documentation

3.37.1.1 float [absMax](#)

3.37.1.2 float [absMin](#)

3.37.1.3 **BOOL** [absValSupported](#)

3.37.1.4 **BOOL** [autoSupported](#)

3.37.1.5 **BOOL** [manualSupported](#)

3.37.1.6 unsigned int [max](#)

3.37.1.7 unsigned int [min](#)

3.37.1.8 **BOOL** [onePushSupported](#)

3.37.1.9 **BOOL** [onOffSupported](#)

3.37.1.10 **BOOL** [present](#)

3.37.1.11 char [pUnitAbbr](#)[MAX_STRING_LENGTH]

3.37.1.12 char [pUnits](#)[MAX_STRING_LENGTH]

3.37.1.13 **BOOL** [readOutSupported](#)

3.37.1.14 unsigned int [reserved](#)[8]

3.37.1.15 **fc2PropertyType** type

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

- [FlyCapture2Defs_C.h](#)

3.38 fc2TriggerMode Struct Reference

Data Fields

- [BOOL onOff](#)
- unsigned int [polarity](#)
- unsigned int [source](#)
- unsigned int [mode](#)
- unsigned int [parameter](#)
- unsigned int [reserved](#) [8]

3.38.1 Field Documentation

3.38.1.1 unsigned int **mode**

3.38.1.2 **BOOL onOff**

3.38.1.3 unsigned int **parameter**

3.38.1.4 unsigned int **polarity**

3.38.1.5 unsigned int **reserved**[8]

3.38.1.6 unsigned int **source**

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

- [FlyCapture2Defs_C.h](#)

3.39 fc2TriggerModelInfo Struct Reference

Data Fields

- [BOOL present](#)
- [BOOL readOutSupported](#)
- [BOOL onOffSupported](#)
- [BOOL polaritySupported](#)
- [BOOL valueReadable](#)
- unsigned int [sourceMask](#)
- [BOOL softwareTriggerSupported](#)
- unsigned int [modeMask](#)
- unsigned int [reserved](#) [8]

3.39.1 Field Documentation

3.39.1.1 unsigned int modeMask

3.39.1.2 BOOL onOffSupported

3.39.1.3 BOOL polaritySupported

3.39.1.4 BOOL present

3.39.1.5 BOOL readOutSupported

3.39.1.6 unsigned int reserved[8]

3.39.1.7 BOOL softwareTriggerSupported

3.39.1.8 unsigned int sourceMask

3.39.1.9 BOOL valueReadable

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

- [FlyCapture2Defs_C.h](#)

3.40 fc2Version Struct Reference

Data Fields

- unsigned int [major](#)
- unsigned int [minor](#)
- unsigned int [type](#)
- unsigned int [build](#)

3.40.1 Field Documentation

3.40.1.1 unsigned int build

3.40.1.2 unsigned int major

3.40.1.3 unsigned int minor

3.40.1.4 unsigned int type

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

- [FlyCapture2Defs_C.h](#)

Chapter 4

File Documentation

4.1 FlyCapture2_C.h File Reference

Functions

- FLYCAPTURE2_C_API [fc2Error](#) [fc2CreateContext](#) ([fc2Context](#) *pContext)
Create a FC2 context for IIDC camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2CreateGigEContext](#) ([fc2Context](#) *pContext)
Create a FC2 context for a GigE Vision camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2DestroyContext](#) ([fc2Context](#) context)
Destroy the FC2 context.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2FireBusReset](#) ([fc2Context](#) context, [fc2PGRGuid](#) *pGuid)
Fire a bus reset.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetNumOfCameras](#) ([fc2Context](#) context, unsigned int *pNumCameras)
Gets the number of cameras attached to the PC.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2IsCameraControlable](#) ([fc2Context](#) context, [fc2PGRGuid](#) *pGuid, [BOOL](#) *pControlable)
Query whether a GigE camera is controlable.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetCameraFromIndex](#) ([fc2Context](#) context, unsigned int index, [fc2PGRGuid](#) *pGuid)
Gets the PGRGuid for a camera on the PC.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetCameraFromSerialNumber](#) ([fc2Context](#) context, unsigned int serialNumber, [fc2PGRGuid](#) *pGuid)
Gets the PGRGuid for a camera on the PC.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetCameraSerialNumberFromIndex](#) ([fc2Context](#) context, unsigned int index, unsigned int *pSerialNumber)
Gets the serial number of the camera with the specified index.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetInterfaceTypeFromGuid](#) ([fc2Context](#) context, [fc2PGRGuid](#) *pGuid, [fc2InterfaceType](#) *pInterfaceType)
Gets the interface type associated with a PGRGuid.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetNumOfDevices](#) ([fc2Context](#) context, unsigned int *pNumDevices)
Gets the number of devices.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetDeviceFromIndex](#) ([fc2Context](#) context, unsigned int index, [fc2PGRGuid](#) *pGuid)
Gets the PGRGuid for a device.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2RegisterCallback](#) ([fc2Context](#) context, [fc2BusEventCallback](#) enumCallback, [fc2BusCallbackType](#) callbackType, void *pParameter, [fc2CallbackHandle](#) *pCallbackHandle)
Register a callback function that will be called when the specified callback event occurs.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2UnregisterCallback](#) ([fc2Context](#) context, [fc2CallbackHandle](#) callbackHandle)
Unregister a callback function.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2RescanBus](#) ([fc2Context](#) context)
Force a rescan of the buses.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2ForceIPAddressToCamera](#) ([fc2Context](#) context, [fc2MACAddress](#) macAddress, [fc2IPAddress](#) ipAddress, [fc2IPAddress](#) subnetMask, [fc2IPAddress](#) defaultGateway)
Force the camera with the specific MAC address to the specified IP address, subnet mask and default gateway.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2ForceAllIPAddressesAutomatically](#) ()
Force all cameras on the network to be assigned sequential IP addresses on the same subnet as the network adapters that they are connected to.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2ForceIPAddressAutomatically](#) (unsigned int serialNumber)
Force a cameras on the network to be assigned sequential IP addresses on the same subnet as the network adapters that it is connected to.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2DiscoverGigECameras](#) ([fc2Context](#) context, [fc2CameraInfo](#) *gigECameras, unsigned int *arraySize)
Discover all cameras connected to the network even if they reside on a different subnet.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2WriteRegister](#) ([fc2Context](#) context, unsigned int address, unsigned int value)
Write to the specified register on the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2WriteRegisterBroadcast](#) ([fc2Context](#) context, unsigned int address, unsigned int value)
Write to the specified register on the camera with broadcast.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2ReadRegister](#) ([fc2Context](#) context, unsigned int address, unsigned int *pValue)
Read the specified register from the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2WriteRegisterBlock](#) ([fc2Context](#) context, unsigned short addressHigh, unsigned int addressLow, const unsigned int *pBuffer, unsigned int length)

Write to the specified register block on the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2ReadRegisterBlock](#) ([fc2Context](#) context, unsigned short addressHigh, unsigned int addressLow, unsigned int *pBuffer, unsigned int length)

Write to the specified register block on the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2Connect](#) ([fc2Context](#) context, [fc2PGRGuid](#) *guid)

Connects the camera object to the camera specified by the GUID.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2Disconnect](#) ([fc2Context](#) context)

Disconnects the fc2Context from the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetCallback](#) ([fc2Context](#) context, [fc2ImageEventCallback](#) pCallbackFn, void *pCallbackData)

Sets the callback data to be used on completion of image transfer.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2StartCapture](#) ([fc2Context](#) context)

Starts isochronous image capture.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2StartCaptureCallback](#) ([fc2Context](#) context, [fc2ImageEventCallback](#) pCallbackFn, void *pCallbackData)

Starts isochronous image capture.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2StartSyncCapture](#) (unsigned int numCameras, [fc2Context](#) *pContexts)

Starts synchronized isochronous image capture on multiple cameras.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2StartSyncCaptureCallback](#) (unsigned int numCameras, [fc2Context](#) *pContexts, [fc2ImageEventCallback](#) *pCallbackFns, void **pCallbackDataArray)

Starts synchronized isochronous image capture on multiple cameras.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2RetrieveBuffer](#) ([fc2Context](#) context, [fc2Image](#) *pImage)

Retrieves the the next image object containing the next image.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2StopCapture](#) ([fc2Context](#) context)

Stops isochronous image transfer and cleans up all associated resources.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetUserBuffers](#) ([fc2Context](#) context, unsigned char *const ppMemBuffers, int size, int nNumBuffers)

Specify user allocated buffers to use as image data buffers.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetConfiguration](#) ([fc2Context](#) context, [fc2Config](#) *config)

Get the configuration associated with the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetConfiguration](#) ([fc2Context](#) context, [fc2Config](#) *config)

Set the configuration associated with the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetCameraInfo](#) ([fc2Context](#) context, [fc2CameraInfo](#) *pCameraInfo)

Retrieves information from the camera such as serial number, model name and other camera information.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetPropertyInfo](#) ([fc2Context](#) context, [fc2PropertyInfo](#) *propInfo)

Retrieves information about the specified camera property.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetProperty](#) ([fc2Context](#) context, [fc2Property](#) *prop)

Reads the settings for the specified property from the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetProperty](#) ([fc2Context](#) context, [fc2Property](#) *prop)

Writes the settings for the specified property to the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetPropertyBroadcast](#) ([fc2Context](#) context, [fc2Property](#) *prop)

Writes the settings for the specified property to the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetGPIOPinDirection](#) ([fc2Context](#) context, unsigned int pin, unsigned int *pDirection)

Get the GPIO pin direction for the specified pin.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetGPIOPinDirection](#) ([fc2Context](#) context, unsigned int pin, unsigned int direction)

Set the GPIO pin direction for the specified pin.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetGPIOPinDirectionBroadcast](#) ([fc2Context](#) context, unsigned int pin, unsigned int direction)

Set the GPIO pin direction for the specified pin.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetTriggerModelInfo](#) ([fc2Context](#) context, [fc2TriggerModelInfo](#) *triggerModelInfo)

Retrieve trigger information from the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetTriggerMode](#) ([fc2Context](#) context, [fc2TriggerMode](#) *triggerMode)

Retrieve current trigger settings from the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetTriggerMode](#) ([fc2Context](#) context, [fc2TriggerMode](#) *triggerMode)

Set the specified trigger settings to the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetTriggerModeBroadcast](#) ([fc2Context](#) context, [fc2TriggerMode](#) *triggerMode)

Set the specified trigger settings to the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2FireSoftwareTrigger](#) ([fc2Context](#) context)
- FLYCAPTURE2_C_API [fc2Error](#) [fc2FireSoftwareTriggerBroadcast](#) ([fc2Context](#) context)

Fire the software trigger according to the DCAM specifications.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetTriggerDelayInfo](#) ([fc2Context](#) context, [fc2TriggerDelayInfo](#) *triggerDelayInfo)

Retrieve trigger delay information from the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetTriggerDelay](#) ([fc2Context](#) context, [fc2TriggerDelay](#) *triggerDelay)

Retrieve current trigger delay settings from the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetTriggerDelay](#) ([fc2Context](#) context, [fc2TriggerDelay](#) *triggerDelay)

Set the specified trigger delay settings to the camera.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetTriggerDelayBroadcast](#) ([fc2Context](#) context, [fc2TriggerDelay](#) *triggerDelay)
Set the specified trigger delay settings to the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetStrobeInfo](#) ([fc2Context](#) context, [fc2StrobeInfo](#) *strobeInfo)
Retrieve strobe information from the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetStrobe](#) ([fc2Context](#) context, [fc2StrobeControl](#) *strobeControl)
Retrieve current strobe settings from the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetStrobe](#) ([fc2Context](#) context, [fc2StrobeControl](#) *strobeControl)
Set current strobe settings to the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetStrobeBroadcast](#) ([fc2Context](#) context, [fc2StrobeControl](#) *strobeControl)
Set current strobe settings to the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetVideoModeAndFrameRateInfo](#) ([fc2Context](#) context, [fc2VideoMode](#) videoMode, [fc2FrameRate](#) frameRate, [BOOL](#) *pSupported)
Query the camera to determine if the specified video mode and frame rate is supported.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetVideoModeAndFrameRate](#) ([fc2Context](#) context, [fc2VideoMode](#) *videoMode, [fc2FrameRate](#) *frameRate)
Get the current video mode and frame rate from the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetVideoModeAndFrameRate](#) ([fc2Context](#) context, [fc2VideoMode](#) videoMode, [fc2FrameRate](#) frameRate)
Set the specified video mode and frame rate to the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetFormat7Info](#) ([fc2Context](#) context, [fc2Format7Info](#) *info, [BOOL](#) *pSupported)
Retrieve the availability of Format7 custom image mode and the camera capabilities for the specified Format7 mode.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2ValidateFormat7Settings](#) ([fc2Context](#) context, [fc2Format7ImageSettings](#) *imageSettings, [BOOL](#) *settingsAreValid, [fc2Format7PacketInfo](#) *packetInfo)
Validates Format7ImageSettings structure and returns valid packet size information if the image settings are valid.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetFormat7Configuration](#) ([fc2Context](#) context, [fc2Format7ImageSettings](#) *imageSettings, unsigned int *packetSize, float *percentage)
Get the current Format7 configuration from the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetFormat7ConfigurationPacket](#) ([fc2Context](#) context, [fc2Format7ImageSettings](#) *imageSettings, unsigned int packetSize)
Set the current Format7 configuration to the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetFormat7Configuration](#) ([fc2Context](#) context, [fc2Format7ImageSettings](#) *imageSettings, float percentSpeed)
Set the current Format7 configuration to the camera.

- FLYCAPTURE2_C_API [fc2Error fc2WriteGVCPRegister](#) ([fc2Context](#) context, unsigned int address, unsigned int value)
Write a GVCP register.
- FLYCAPTURE2_C_API [fc2Error fc2WriteGVCPRegisterBroadcast](#) ([fc2Context](#) context, unsigned int address, unsigned int value)
Write a GVCP register with broadcast.
- FLYCAPTURE2_C_API [fc2Error fc2ReadGVCPRegister](#) ([fc2Context](#) context, unsigned int address, unsigned int *pValue)
Read a GVCP register.
- FLYCAPTURE2_C_API [fc2Error fc2WriteGVCPRegisterBlock](#) ([fc2Context](#) context, unsigned int address, const unsigned int *pBuffer, unsigned int length)
Write a GVCP register block.
- FLYCAPTURE2_C_API [fc2Error fc2ReadGVCPRegisterBlock](#) ([fc2Context](#) context, unsigned int address, unsigned int *pBuffer, unsigned int length)
Read a GVCP register block.
- FLYCAPTURE2_C_API [fc2Error fc2WriteGVCPMemory](#) ([fc2Context](#) context, unsigned int address, const unsigned char *pBuffer, unsigned int length)
Write a GVCP memory block.
- FLYCAPTURE2_C_API [fc2Error fc2ReadGVCPMemory](#) ([fc2Context](#) context, unsigned int address, unsigned char *pBuffer, unsigned int length)
Read a GVCP memory block.
- FLYCAPTURE2_C_API [fc2Error fc2GetGigEProperty](#) ([fc2Context](#) context, [fc2GigEProperty](#) *pGigEProp)
Get the specified GigEProperty.
- FLYCAPTURE2_C_API [fc2Error fc2SetGigEProperty](#) ([fc2Context](#) context, const [fc2GigEProperty](#) *pGigEProp)
Set the specified GigEProperty.
- FLYCAPTURE2_C_API [fc2Error fc2QueryGigEImagingMode](#) ([fc2Context](#) context, [fc2Mode](#) mode, [BOOL](#) *isSupported)
- FLYCAPTURE2_C_API [fc2Error fc2GetGigEImagingMode](#) ([fc2Context](#) context, [fc2Mode](#) *mode)
- FLYCAPTURE2_C_API [fc2Error fc2SetGigEImagingMode](#) ([fc2Context](#) context, [fc2Mode](#) mode)
- FLYCAPTURE2_C_API [fc2Error fc2GetGigEImageSettingsInfo](#) ([fc2Context](#) context, [fc2GigEImageSettingsInfo](#) *pInfo)
- FLYCAPTURE2_C_API [fc2Error fc2GetGigEImageSettings](#) ([fc2Context](#) context, [fc2GigEImageSettings](#) *pImageSettings)
- FLYCAPTURE2_C_API [fc2Error fc2SetGigEImageSettings](#) ([fc2Context](#) context, const [fc2GigEImageSettings](#) *pImageSettings)
- FLYCAPTURE2_C_API [fc2Error fc2GetGigEConfig](#) ([fc2Context](#) context, [fc2GigEConfig](#) *pConfig)
- FLYCAPTURE2_C_API [fc2Error fc2SetGigEConfig](#) ([fc2Context](#) context, const [fc2GigEConfig](#) *pConfig)
- FLYCAPTURE2_C_API [fc2Error fc2GetGigEImageBinningSettings](#) ([fc2Context](#) context, unsigned int *horzBinningValue, unsigned int *vertBinningValue)

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetGigEImageBinningSettings](#) ([fc2Context](#) context, unsigned int horzBinningValue, unsigned int vertBinningValue)
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetNumStreamChannels](#) ([fc2Context](#) context, unsigned int *numChannels)
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetGigEStreamChannelInfo](#) ([fc2Context](#) context, unsigned int channel, [fc2GigEStreamChannel](#) *pChannel)
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetGigEStreamChannelInfo](#) ([fc2Context](#) context, unsigned int channel, [fc2GigEStreamChannel](#) *pChannel)
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetLUTInfo](#) ([fc2Context](#) context, [fc2LUT-Data](#) *pData)
Query if LUT support is available on the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetLUTBankInfo](#) ([fc2Context](#) context, unsigned int bank, [BOOL](#) *pReadSupported, [BOOL](#) *pWriteSupported)
Query the read/write status of a single LUT bank.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetActiveLUTBank](#) ([fc2Context](#) context, unsigned int *pActiveBank)
Get the LUT bank that is currently being used.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetActiveLUTBank](#) ([fc2Context](#) context, unsigned int activeBank)
Set the LUT bank that will be used.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2EnableLUT](#) ([fc2Context](#) context, [BOOL](#) on)
Enable or disable LUT functionality on the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetLUTChannel](#) ([fc2Context](#) context, unsigned int bank, unsigned int channel, unsigned int sizeEntries, unsigned int *pEntries)
Get the LUT channel settings from the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetLUTChannel](#) ([fc2Context](#) context, unsigned int bank, unsigned int channel, unsigned int sizeEntries, unsigned int *pEntries)
Set the LUT channel settings to the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetMemoryChannel](#) ([fc2Context](#) context, unsigned int *pCurrentChannel)
Retrieve the current memory channel from the camera.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SaveToMemoryChannel](#) ([fc2Context](#) context, unsigned int channel)
Save the current settings to the specified current memory channel.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2RestoreFromMemoryChannel](#) ([fc2Context](#) context, unsigned int channel)
Restore the specified current memory channel.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetMemoryChannelInfo](#) ([fc2Context](#) context, unsigned int *pNumChannels)
Query the camera for memory channel support.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetEmbeddedImageInfo](#) ([fc2Context](#) context, [fc2EmbeddedImageInfo](#) *pInfo)
Get the current status of the embedded image information register, as well as the availability of each embedded property.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetEmbeddedImageInfo](#) ([fc2Context](#) context, [fc2EmbeddedImageInfo](#) *pInfo)
Sets the on/off values of the embedded image information structure to the camera.
- FLYCAPTURE2_C_API const char * [fc2GetRegisterString](#) (unsigned int registerVal)
Returns a text representation of the register value.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2CreateImage](#) ([fc2Image](#) *pImage)
Create a [fc2Image](#).
- FLYCAPTURE2_C_API [fc2Error](#) [fc2DestroyImage](#) ([fc2Image](#) *image)
Destroy the [fc2Image](#).
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetDefaultColorProcessing](#) ([fc2ColorProcessingAlgorithm](#) defaultMethod)
Set the default color processing algorithm.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetDefaultColorProcessing](#) ([fc2ColorProcessingAlgorithm](#) *pDefaultMethod)
Get the default color processing algorithm.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetDefaultOutputFormat](#) ([fc2PixelFormat](#) format)
Set the default output pixel format.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetDefaultOutputFormat](#) ([fc2PixelFormat](#) *pFormat)
Get the default output pixel format.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2DetermineBitsPerPixel](#) ([fc2PixelFormat](#) format, unsigned int *pBitsPerPixel)
Calculate the bits per pixel for the specified pixel format.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SaveImage](#) ([fc2Image](#) *pImage, const char *pFilename, [fc2ImageFileFormat](#) format)
Save the image to the specified file name with the file format specified.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SaveImageWithOptions](#) ([fc2Image](#) *pImage, const char *pFilename, [fc2ImageFileFormat](#) format, void *pOption)
Save the image to the specified file name with the file format specified.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2ConvertImage](#) ([fc2Image](#) *pImageIn, [fc2Image](#) *pImageOut)
- FLYCAPTURE2_C_API [fc2Error](#) [fc2ConvertImageTo](#) ([fc2PixelFormat](#) format, [fc2Image](#) *pImageIn, [fc2Image](#) *pImageOut)
Converts the current image buffer to the specified output format and stores the result in the specified image.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetImageData](#) ([fc2Image](#) *pImage, unsigned char **ppData)
Get a pointer to the data associated with the image.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetImageData](#) ([fc2Image](#) *pImage, const unsigned char *pData, unsigned int dataSize)
Set the data of the Image object.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetImageDimensions](#) ([fc2Image](#) *pImage, unsigned int rows, unsigned int cols, unsigned int stride, [fc2PixelFormat](#) pixelFormat, [fc2BayerTileFormat](#) bayerFormat)

Sets the dimensions of the image object.

- FLYCAPTURE2_C_API [fc2TimeStamp](#) [fc2GetImageTimeStamp](#) ([fc2Image](#) *pImage)

Get the timestamp data associated with the image.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2CalculateImageStatistics](#) ([fc2Image](#) *pImage, [fc2ImageStatisticsContext](#) *pImageStatisticsContext)

Calculate statistics associated with the image.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2CreateImageStatistics](#) ([fc2ImageStatisticsContext](#) *pImageStatisticsContext)

Create a statistics context.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2DestroyImageStatistics](#) ([fc2ImageStatisticsContext](#) imageStatisticsContext)

Destroy a statistics context.

- FLYCAPTURE2_C_API const [fc2Error](#) [fc2GetChannelStatus](#) ([fc2ImageStatisticsContext](#) imageStatisticsContext, [fc2StatisticsChannel](#) channel, [BOOL](#) *pEnabled)

Get the status of a statistics channel.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2SetChannelStatus](#) ([fc2ImageStatisticsContext](#) imageStatisticsContext, [fc2StatisticsChannel](#) channel, [BOOL](#) enabled)

Set the status of a statistics channel.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetImageStatistics](#) ([fc2ImageStatisticsContext](#) imageStatisticsContext, [fc2StatisticsChannel](#) channel, unsigned int *pRangeMin, unsigned int *pRangeMax, unsigned int *pPixelValueMin, unsigned int *pPixelValueMax, unsigned int *pNumPixelValues, float *pPixelValueMean, int **ppHistogram)

Get all statistics for the image.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2CreateAVI](#) ([fc2AVIContext](#) *pAVIContext)

Create a AVI context.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2AVIOpen](#) ([fc2AVIContext](#) AVIContext, const char *pFileName, [fc2AVIOption](#) *pOption)

Open an AVI file in preparation for writing Images to disk.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2MJPEGOpen](#) ([fc2AVIContext](#) AVIContext, const char *pFileName, [fc2MJPEGOption](#) *pOption)

Open an MJPEG file in preparation for writing Images to disk.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2H264Open](#) ([fc2AVIContext](#) AVIContext, const char *pFileName, [fc2H264Option](#) *pOption)

Open an H.264 file in preparation for writing Images to disk.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2AVIAppend](#) ([fc2AVIContext](#) AVIContext, [fc2Image](#) *pImage)

Append an image to the AVI file.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2AVIClose](#) ([fc2AVIContext](#) AVIContext)

Close the AVI file.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2DestroyAVI](#) ([fc2AVIContext](#) AVIContext)

Destroy a AVI context.

- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetSystemInfo](#) ([fc2SystemInfo](#) *pSystem-Info)
Get system information.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetLibraryVersion](#) ([fc2Version](#) *pVersion)
Get library version.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2LaunchBrowser](#) (const char *pAddress)
Launch a URL in the system default browser.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2LaunchHelp](#) (const char *pFileName)
Open a CHM file in the system default CHM viewer.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2LaunchCommand](#) (const char *p-Command)
Execute a command in the terminal.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2LaunchCommandAsync](#) (const char *p-Command, [fc2AsyncCommandCallback](#) pCallback, void *pUserData)
Execute a command in the terminal.
- FLYCAPTURE2_C_API const char * [fc2ErrorToDescription](#) ([fc2Error](#) error)
Get a string representation of an error.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2GetCycleTime](#) ([fc2Context](#) context, [fc2Time-Stamp](#) *pTimeStamp)
Get cycle time from camera.

4.1.1 Function Documentation

4.1.1.1 FLYCAPTURE2_C_API [fc2Error](#) [fc2AVIAppend](#) ([fc2AVIContext](#) *AVIContext*, [fc2Image](#) * *pImage*)

Append an image to the AVI file.

Parameters

<i>AVIContext</i>	The AVI context to use.
<i>pImage</i>	The image to append.

Returns

A [fc2Error](#) indicating the success or failure of the function.

4.1.1.2 FLYCAPTURE2_C_API [fc2Error](#) [fc2AVIClose](#) ([fc2AVIContext](#) *AVIContext*)

Close the AVI file.

Parameters

<i>AVIContext</i>	The AVI context to use.
-------------------	-------------------------

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.3 FLYCAPTURE2_C_API `fc2Error fc2AVIOpen (fc2AVIContext AVIContext, const char * pFileName, fc2AVIOption * pOption)`

Open an AVI file in preparation for writing Images to disk.

The size of AVI files is limited to 2GB. The filenames are automatically generated using the filename specified.

Parameters

<i>AVIContext</i>	The AVI context to use.
<i>pFileName</i>	The filename of the AVI file.
<i>pOption</i>	Options to apply to the AVI file.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.4 FLYCAPTURE2_C_API `fc2Error fc2CalculateImageStatistics (fc2Image * plImage, fc2ImageStatisticsContext * plImageStatisticsContext)`

Calculate statistics associated with the image.

In order to collect statistics for a particular channel, the enabled flag for the channel must be set to true. Statistics can only be collected for images in Mono8, Mono16, RGB, RGBU, BGR and BGRU.

Parameters

<i>plImage</i>	The fc2Image to be used.
<i>plImage-Statistics-Context</i>	The <code>fc2ImageStatisticsContext</code> to hold the statistics.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.5 FLYCAPTURE2_C_API `fc2Error fc2Connect (fc2Context context, fc2PGRGuid * guid)`

Connects the camera object to the camera specified by the GUID.

Parameters

<i>context</i>	The fc2Context to be used.
<i>guid</i>	The unique identifier for a specific camera on the PC.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.6 FLYCAPTURE2_C_API fc2Error fc2ConvertImage (fc2Image * *plmageIn*, fc2Image * *plmageOut*)

Parameters

<i>plmageIn</i>	
<i>plmageOut</i>	

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.7 FLYCAPTURE2_C_API fc2Error fc2ConvertImageTo (fc2PixelFormat *format*, fc2Image * *plmageIn*, fc2Image * *plmageOut*)

Converts the current image buffer to the specified output format and stores the result in the specified image.

The destination image does not need to be configured in any way before the call is made.

Parameters

<i>format</i>	Output format of the converted image.
<i>plmageIn</i>	Input image.
<i>plmageOut</i>	Output image.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.8 FLYCAPTURE2_C_API fc2Error fc2CreateAVI (fc2AVIContext * *pAVIContext*)

Create a AVI context.

Parameters

<i>pAVIContext</i>	A AVI context.
--------------------	----------------

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.9 FLYCAPTURE2_C_API fc2Error fc2CreateContext (fc2Context * *pContext*)

Create a FC2 context for IIDC camera.

This call must be made before any other calls that use a context will succeed.

Parameters

<i>pContext</i>	A pointer to the <code>fc2Context</code> to be created.
-----------------	---

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.10 FLYCAPTURE2_C_API fc2Error fc2CreateGigEContext (fc2Context * *pContext*)

Create a FC2 context for a GigE Vision camera.

This call must be made before any other calls that use a context will succeed.

Parameters

<i>pContext</i>	A pointer to the <code>fc2Context</code> to be created.
-----------------	---

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.11 FLYCAPTURE2_C_API fc2Error fc2CreateImage (fc2Image * *pImage*)

Create a [fc2Image](#).

If externally allocated memory is to be used for the converted image, simply assigning the `pData` member of the [fc2Image](#) structure is insufficient. [fc2SetImageData\(\)](#) should be called in order to populate the [fc2Image](#) structure correctly.

Parameters

<i>pImage</i>	Pointer to image to be created.
---------------	---------------------------------

See also

[fc2SetImageData\(\)](#)

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.12 FLYCAPTURE2_C_API `fc2Error` `fc2CreateImageStatistics` (`fc2ImageStatisticsContext` * *plImageStatisticsContext*)

Create a statistics context.

Parameters

<i>plImage-Statistics-Context</i>	A statistics context.
-----------------------------------	-----------------------

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.13 FLYCAPTURE2_C_API `fc2Error` `fc2DestroyAVI` (`fc2AVIContext` *AVIContext*)

Destroy a AVI context.

Parameters

<i>AVIContext</i>	A AVI context.
-------------------	----------------

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.14 FLYCAPTURE2_C_API `fc2Error` `fc2DestroyContext` (`fc2Context` *context*)

Destroy the FC2 context.

This must be called when the user is finished with the context in order to prevent memory leaks.

Parameters

<i>context</i>	The context to be destroyed.
----------------	------------------------------

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.15 FLYCAPTURE2_C_API fc2Error fc2DestroyImage (fc2Image * *image*)

Destroy the [fc2Image](#).

Parameters

<i>image</i>	Pointer to image to be destroyed.
--------------	-----------------------------------

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.16 FLYCAPTURE2_C_API fc2Error fc2DestroyImageStatistics (fc2ImageStatisticsContext *imageStatisticsContext*)

Destroy a statistics context.

Parameters

<i>image-Statistics-Context</i>	A statistics context.
---------------------------------	-----------------------

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.17 FLYCAPTURE2_C_API fc2Error fc2DetermineBitsPerPixel (fc2PixelFormat *format*, unsigned int * *pBitsPerPixel*)

Calculate the bits per pixel for the specified pixel format.

Parameters

<i>format</i>	The pixel format.
<i>pBitsPerPixel</i>	The bits per pixel.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.18 FLYCAPTURE2_C_API fc2Error fc2Disconnect (fc2Context *context*)

Disconnects the fc2Context from the camera.

Parameters

<i>context</i>	The fc2Context to be used.
----------------	----------------------------

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.19 FLYCAPTURE2_C_API fc2Error fc2DiscoverGigECameras (fc2Context context, fc2CameraInfo * gigECameras, unsigned int * arraySize)

Discover all cameras connected to the network even if they reside on a different subnet.

This is useful in situations where a GigE camera is using Persistent IP and the application's subnet is different from the device subnet. After discovering the camera, it is easy to use ForceIPAddressToCamera() to set a different IP configuration.

Parameters

<i>context</i>	The fc2Context to be used.
<i>gigE-Cameras</i>	Pointer to an array of CameraInfo structures.
<i>arraySize</i>	Size of the array. Number of discovered cameras is returned in the same value.

Returns

An Error indicating the success or failure of the function. If the error is PGRERROR_BUFFER_TOO_SMALL then arraySize will contain the minimum size needed for gigECameras array.

4.1.1.20 FLYCAPTURE2_C_API fc2Error fc2EnableLUT (fc2Context context, BOOL on)

Enable or disable LUT functionality on the camera.

Parameters

<i>context</i>	The fc2Context to be used.
<i>on</i>	Whether to enable or disable LUT.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.21 FLYCAPTURE2_C_API const char* fc2ErrorToDescription (fc2Error error)

Get a string representation of an error.

Parameters

<i>error</i>	Error to be parsed.
--------------	---------------------

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.22 FLYCAPTURE2_C_API `fc2Error` `fc2FireBusReset` (`fc2Context` *context*, `fc2PGRGuid` * *pGuid*)

Fire a bus reset.

The actual bus reset is only fired for the specified 1394 bus, but it will effectively cause a global bus reset for the library.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>pGuid</i>	PGRGuid of the camera or the device to cause bus reset.

Returns

An `Error` indicating the success or failure of the function.

4.1.1.23 FLYCAPTURE2_C_API `fc2Error` `fc2FireSoftwareTrigger` (`fc2Context` *context*)

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
----------------	---

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.24 FLYCAPTURE2_C_API `fc2Error` `fc2FireSoftwareTriggerBroadcast` (`fc2Context` *context*)

Fire the software trigger according to the DCAM specifications.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
----------------	---

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.25 FLYCAPTURE2_C_API fc2Error fc2ForceAllIPAddressesAutomatically ()

Force all cameras on the network to be assigned sequential IP addresses on the same subnet as the network adapters that they are connected to.

This is useful in situations where a GigE Vision cameras are using Persistent IP addresses and the application's subnet is different from the devices.

Returns

An Error indicating the success or failure of the function.

4.1.1.26 FLYCAPTURE2_C_API fc2Error fc2ForceIPAddressAutomatically (unsigned int serialNumber)

Force a cameras on the network to be assigned sequential IP addresses on the same subnet as the network adapters that it is connected to.

This is useful in situations where a GigE Vision cameras is using Persistent IP addresses and the application's subnet is different from the device.

Returns

An Error indicating the success or failure of the function.

4.1.1.27 FLYCAPTURE2_C_API fc2Error fc2ForceIPAddressToCamera (fc2Context context, fc2MACAddress macAddress, fc2IPAddress ipAddress, fc2IPAddress subnetMask, fc2IPAddress defaultGateway)

Force the camera with the specific MAC address to the specified IP address, subnet mask and default gateway.

This is useful in situations where a GigE Vision camera is using Persistent IP and the application's subnet is different from the device subnet.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>macAddress</i>	MAC address of the camera.
<i>ipAddress</i>	IP address to set on the camera.
<i>subnetMask</i>	Subnet mask to set on the camera.
<i>default-Gateway</i>	Default gateway to set on the camera.

Returns

An Error indicating the success or failure of the function.

4.1.1.28 FLYCAPTURE2_C_API fc2Error fc2GetActiveLUTBank (fc2Context context, unsigned int * pActiveBank)

Get the LUT bank that is currently being used.

For cameras with PGR LUT, the active bank is always 0.

Parameters

<i>context</i>	The fc2Context to be used.
<i>pActiveBank</i>	The currently active bank.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.29 FLYCAPTURE2_C_API fc2Error fc2GetCameraFromIndex (fc2Context context, unsigned int index, fc2PGRGuid * pGuid)

Gets the PGRGuid for a camera on the PC.

It uniquely identifies the camera specified by the index and is used to identify the camera during a [fc2Connect\(\)](#) call.

Parameters

<i>context</i>	The fc2Context to be used.
<i>index</i>	Zero based index of camera.
<i>pGuid</i>	Unique PGRGuid for the camera.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.30 FLYCAPTURE2_C_API fc2Error fc2GetCameraFromSerialNumber (fc2Context context, unsigned int serialNumber, fc2PGRGuid * pGuid)

Gets the PGRGuid for a camera on the PC.

It uniquely identifies the camera specified by the serial number and is used to identify the camera during a [fc2Connect\(\)](#) call.

Parameters

<i>context</i>	The fc2Context to be used.
<i>serial-Number</i>	Serial number of camera.
<i>pGuid</i>	Unique PGRGuid for the camera.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.31 FLYCAPTURE2_C_API fc2Error fc2GetCameraInfo (fc2Context *context*, fc2CameraInfo * *pCameraInfo*)

Retrieves information from the camera such as serial number, model name and other camera information.

Parameters

<i>context</i>	The fc2Context to be used.
<i>pCameraInfo</i>	Pointer to the camera information structure to be filled.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.32 FLYCAPTURE2_C_API fc2Error fc2GetCameraSerialNumberFromIndex (fc2Context *context*, unsigned int *index*, unsigned int * *pSerialNumber*)

Gets the serial number of the camera with the specified index.

Parameters

<i>context</i>	The fc2Context to be used.
<i>index</i>	Zero based index of desired camera.
<i>pSerial-Number</i>	Serial number of camera.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.33 FLYCAPTURE2_C_API const fc2Error fc2GetChannelStatus (fc2ImageStatisticsContext *imageStatisticsContext*, fc2StatisticsChannel *channel*, BOOL * *pEnabled*)

Get the status of a statistics channel.

Parameters

<i>image-Statistics-Context</i>	A statistics context.
<i>channel</i>	The statistics channel.
<i>pEnabled</i>	Whether the channel is enabled.

See also

SetChannelStatus()

Returns

An Error indicating the success or failure of the function.

4.1.1.34 FLYCAPTURE2_C_API fc2Error fc2GetConfiguration (fc2Context *context*, fc2Config * *config*)

Get the configuration associated with the camera.

Parameters

<i>context</i>	The fc2Context to be used.
<i>config</i>	Pointer to the configuration structure to be filled.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.35 FLYCAPTURE2_C_API fc2Error fc2GetCycleTime (fc2Context *context*, fc2TimeStamp * *pTimeStamp*)

Get cycle time from camera.

Parameters

<i>TimeStamp</i>	struct.
------------------	---------

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.36 FLYCAPTURE2_C_API `fc2Error fc2GetDefaultColorProcessing (fc2ColorProcessingAlgorithm * pDefaultMethod)`

Get the default color processing algorithm.

Parameters

<i>pDefault-Method</i>	The default color processing algorithm.
------------------------	---

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.37 FLYCAPTURE2_C_API `fc2Error fc2GetDefaultOutputFormat (fc2PixelFormat * pFormat)`

Get the default output pixel format.

Parameters

<i>pFormat</i>	The default pixel format.
----------------	---------------------------

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.38 FLYCAPTURE2_C_API `fc2Error fc2GetDeviceFromIndex (fc2Context context, unsigned int index, fc2PGRGuid * pGuid)`

Gets the PGRGuid for a device.

It uniquely identifies the device specified by the index.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>index</i>	Zero based index of device.
<i>pGuid</i>	Unique PGRGuid for the device.

See also

[fc2GetNumOfDevices\(\)](#)

Returns

An Error indicating the success or failure of the function.

4.1.1.39 FLYCAPTURE2_C_API fc2Error fc2GetEmbeddedImageInfo (fc2Context *context*, fc2EmbeddedImageInfo * *pInfo*)

Get the current status of the embedded image information register, as well as the availability of each embedded property.

Parameters

<i>context</i>	The fc2Context to be used.
<i>pInfo</i>	Structure to be filled.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.40 FLYCAPTURE2_C_API fc2Error fc2GetFormat7Configuration (fc2Context *context*, fc2Format7ImageSettings * *imageSettings*, unsigned int * *packetSize*, float * *percentage*)

Get the current Format7 configuration from the camera.

This call will only succeed if the camera is already in Format7.

Parameters

<i>context</i>	The fc2Context to be used.
<i>image-Settings</i>	Current image settings.
<i>packetSize</i>	Current packet size.
<i>percentage</i>	Current packet size as a percentage.

Returns

A `fc2Error` indicating the success or failure of the function.

**4.1.1.41 FLYCAPTURE2_C_API `fc2Error` `fc2GetFormat7Info` (`fc2Context` *context*,
`fc2Format7Info` * *info*, `BOOL` * *pSupported*)**

Retrieve the availability of Format7 custom image mode and the camera capabilities for the specified Format7 mode.

The mode must be specified in the `Format7Info` structure in order for the function to succeed.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>info</i>	Structure to be filled with the capabilities of the specified mode and the current state in the specified mode.
<i>pSupported</i>	Whether the specified mode is supported.

Returns

A `fc2Error` indicating the success or failure of the function.

**4.1.1.42 FLYCAPTURE2_C_API `fc2Error` `fc2GetGigEConfig` (`fc2Context` *context*,
`fc2GigEConfig` * *pConfig*)**

**4.1.1.43 FLYCAPTURE2_C_API `fc2Error` `fc2GetGigElmageBinningSettings` (`fc2Context` *context*,
`unsigned int` * *horzBinningValue*, `unsigned int` * *vertBinningValue*)**

**4.1.1.44 FLYCAPTURE2_C_API `fc2Error` `fc2GetGigElmageSettings` (`fc2Context` *context*,
`fc2GigElmageSettings` * *pImageSettings*)**

**4.1.1.45 FLYCAPTURE2_C_API `fc2Error` `fc2GetGigElmageSettingsInfo` (`fc2Context` *context*,
`fc2GigElmageSettingsInfo` * *pInfo*)**

**4.1.1.46 FLYCAPTURE2_C_API `fc2Error` `fc2GetGigElmagingMode` (`fc2Context` *context*,
`fc2Mode` * *mode*)**

**4.1.1.47 FLYCAPTURE2_C_API `fc2Error` `fc2GetGigEProperty` (`fc2Context` *context*,
`fc2GigEProperty` * *pGigEProp*)**

Get the specified `GigEProperty`.

The `GigEPropertyType` field must be set in order for this function to succeed.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>pGigEProp</i>	The <code>GigE</code> property to get.

Returns

An Error indicating the success or failure of the function.

4.1.1.48 FLYCAPTURE2_C_API **fc2Error** **fc2GetGigEStreamChannelInfo** (**fc2Context** *context*,
unsigned int *channel*, **fc2GigEStreamChannel** * *pChannel*)

4.1.1.49 FLYCAPTURE2_C_API **fc2Error** **fc2GetGPIOPinDirection** (**fc2Context** *context*,
unsigned int *pin*, unsigned int * *pDirection*)

Get the GPIO pin direction for the specified pin.

This is not a required call when using the trigger or strobe functions as the pin direction is set automatically internally.

Parameters

<i>context</i>	The fc2Context to be used.
<i>pin</i>	Pin to get the direction for.
<i>pDirection</i>	Direction of the pin. 0 for input, 1 for output.

Returns

A **fc2Error** indicating the success or failure of the function.

4.1.1.50 FLYCAPTURE2_C_API **fc2Error** **fc2GetImageData** (**fc2Image** * *pImage*, unsigned
char ** *ppData*)

Get a pointer to the data associated with the image.

This function is considered unsafe. The pointer returned could be invalidated if the buffer is resized or released. The pointer may also be invalidated if the Image object is passed to [fc2RetrieveBuffer\(\)](#).

Parameters

<i>pImage</i>	The fc2Image to be used.
<i>ppData</i>	A pointer to the image data.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.51 FLYCAPTURE2_C_API `fc2Error fc2GetImageStatistics (fc2ImageStatisticsContext imageStatisticsContext, fc2StatisticsChannel channel, unsigned int * pRangeMin, unsigned int * pRangeMax, unsigned int * pPixelValueMin, unsigned int * pPixelValueMax, unsigned int * pNumPixelValues, float * pPixelValueMean, int ** ppHistogram)`

Get all statistics for the image.

Parameters

<i>imageStatisticsContext</i>	The statistics context.
<i>channel</i>	The statistics channel.
<i>pRangeMin</i>	The minimum possible value.
<i>pRangeMax</i>	The maximum possible value.
<i>pPixelValueMin</i>	The minimum pixel value.
<i>pPixelValueMax</i>	The maximum pixel value.
<i>pNumPixelValues</i>	The number of unique pixel values.
<i>pPixelValueMean</i>	The mean of the image.
<i>ppHistogram</i>	Pointer to an array containing the histogram.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.52 FLYCAPTURE2_C_API `fc2TimeStamp fc2GetImageTimeStamp (fc2Image * pImage)`

Get the timestamp data associated with the image.

Parameters

<i>pImage</i>	The fc2Image to be used.
---------------	--

Returns

Timestamp data associated with the image.

4.1.1.53 FLYCAPTURE2_C_API fc2Error fc2GetInterfaceTypeFromGuid (fc2Context context, fc2PGRGuid * pGuid, fc2InterfaceType * pInterfaceType)

Gets the interface type associated with a PGRGuid.

This is useful in situations where there is a need to enumerate all cameras for a particular interface.

Parameters

<i>context</i>	The fc2Context to be used.
<i>pGuid</i>	The PGRGuid to get the interface for.
<i>pInterface-Type</i>	The interface type of the PGRGuid.

Returns**4.1.1.54 FLYCAPTURE2_C_API fc2Error fc2GetLibraryVersion (fc2Version * pVersion)**

Get library version.

Parameters

<i>pVersion</i>	Structure to receive the library version.
-----------------	---

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.55 FLYCAPTURE2_C_API fc2Error fc2GetLUTBankInfo (fc2Context context, unsigned int bank, BOOL * pReadSupported, BOOL * pWriteSupported)

Query the read/write status of a single LUT bank.

Parameters

<i>context</i>	The fc2Context to be used.
<i>bank</i>	The bank to query.
<i>pRead-Supported</i>	Whether reading from the bank is supported.
<i>pWrite-Supported</i>	Whether writing to the bank is supported.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.56 FLYCAPTURE2_C API `fc2Error fc2GetLUTChannel (fc2Context context, unsigned int bank, unsigned int channel, unsigned int sizeEntries, unsigned int * pEntries)`

Get the LUT channel settings from the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>bank</i>	Bank to retrieve.
<i>channel</i>	Channel to retrieve.
<i>sizeEntries</i>	Number of entries in LUT table to read.
<i>pEntries</i>	Array to store LUT entries.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.57 FLYCAPTURE2_C API `fc2Error fc2GetLUTInfo (fc2Context context, fc2LUTData * pData)`

Query if LUT support is available on the camera.

Note that some cameras may report support for the LUT and return an `inputBitDepth` of 0. In these cases use `log2(numEntries)` for the `inputBitDepth`.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>pData</i>	The LUT structure to be filled.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.58 FLYCAPTURE2_C API `fc2Error fc2GetMemoryChannel (fc2Context context, unsigned int * pCurrentChannel)`

Retrieve the current memory channel from the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>pCurrentChannel</i>	Current memory channel.

Returns

A `fc2Error` indicating the success or failure of the function.

**4.1.1.59 FLYCAPTURE2_C_API `fc2Error` `fc2GetMemoryChannelInfo` (`fc2Context` *context*,
unsigned int * *pNumChannels*)**

Query the camera for memory channel support.

If the number of channels is 0, then memory channel support is not available.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>pNum-Channels</i>	Number of memory channels supported.

Returns

A `fc2Error` indicating the success or failure of the function.

**4.1.1.60 FLYCAPTURE2_C_API `fc2Error` `fc2GetNumOfCameras` (`fc2Context` *context*,
unsigned int * *pNumCameras*)**

Gets the number of cameras attached to the PC.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>pNum-Cameras</i>	Number of cameras detected.

Returns

A `fc2Error` indicating the success or failure of the function.

**4.1.1.61 FLYCAPTURE2_C_API `fc2Error` `fc2GetNumOfDevices` (`fc2Context` *context*,
unsigned int * *pNumDevices*)**

Gets the number of devices.

This may include hubs, host controllers and other hardware devices (including cameras).

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>pNum-Devices</i>	The number of devices found.

Returns

An Error indicating the success or failure of the function.

4.1.1.62 FLYCAPTURE2_C_API fc2Error fc2GetNumStreamChannels (fc2Context *context*, unsigned int * *numChannels*)

4.1.1.63 FLYCAPTURE2_C_API fc2Error fc2GetProperty (fc2Context *context*, fc2Property * *prop*)

Reads the settings for the specified property from the camera.

The property type must be specified in the fc2Property structure passed into the function in order for the function to succeed. If auto is on, the integer and abs values returned may not be consistent with each other.

Parameters

<i>context</i>	The fc2Context to be used.
<i>prop</i>	Pointer to the Property structure to be filled.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.64 FLYCAPTURE2_C_API fc2Error fc2GetPropertyInfo (fc2Context *context*, fc2PropertyInfo * *propInfo*)

Retrieves information about the specified camera property.

The property type must be specified in the fc2PropertyInfo structure passed into the function in order for the function to succeed.

Parameters

<i>context</i>	The fc2Context to be used.
<i>propInfo</i>	Pointer to the PropertyInfo structure to be filled.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.65 FLYCAPTURE2_C_API const char* fc2GetRegisterString (unsigned int *registerVal*)

Returns a text representation of the register value.

Parameters

<i>registerVal</i>	The register value to query.
--------------------	------------------------------

Returns

A `fc2Error` indicating the success or failure of the function.

**4.1.1.66 FLYCAPTURE2_C_API `fc2Error` `fc2GetStrobe` (`fc2Context` *context*,
`fc2StrobeControl` * *strobeControl*)**

Retrieve current strobe settings from the camera.

The strobe pin must be specified in the structure before being passed in to the function.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>strobe-Control</i>	Structure to receive strobe settings.

Returns

A `fc2Error` indicating the success or failure of the function.

**4.1.1.67 FLYCAPTURE2_C_API `fc2Error` `fc2GetStrobeInfo` (`fc2Context` *context*,
`fc2StrobeInfo` * *strobeInfo*)**

Retrieve strobe information from the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>strobeInfo</i>	Structure to receive strobe information.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.68 FLYCAPTURE2_C_API `fc2Error` `fc2GetSystemInfo` (`fc2SystemInfo` * *pSystemInfo*)

Get system information.

Parameters

<i>pSystemInfo</i>	Structure to receive system information.
--------------------	--

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.69 FLYCAPTURE2_C_API `fc2Error` `fc2GetTriggerDelay` (`fc2Context` *context*, `fc2TriggerDelay` * *triggerDelay*)

Retrieve current trigger delay settings from the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>triggerDelay</i>	Structure to receive trigger delay settings.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.70 FLYCAPTURE2_C_API `fc2Error` `fc2GetTriggerDelayInfo` (`fc2Context` *context*, `fc2TriggerDelayInfo` * *triggerDelayInfo*)

Retrieve trigger delay information from the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>triggerDelay-Info</i>	Structure to receive trigger delay information.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.71 FLYCAPTURE2_C_API `fc2Error` `fc2GetTriggerMode` (`fc2Context` *context*, `fc2TriggerMode` * *triggerMode*)

Retrieve current trigger settings from the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>triggerMode</i>	Structure to receive trigger mode settings.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.72 FLYCAPTURE2_C_API fc2Error fc2GetTriggerModelInfo (fc2Context context, fc2TriggerModelInfo * triggerModelInfo)

Retrieve trigger information from the camera.

Parameters

<i>context</i>	The fc2Context to be used.
<i>triggerModelInfo</i>	Structure to receive trigger information.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.73 FLYCAPTURE2_C_API fc2Error fc2GetVideoModeAndFrameRate (fc2Context context, fc2VideoMode * videoMode, fc2FrameRate * frameRate)

Get the current video mode and frame rate from the camera.

If the camera is in Format7, the video mode will be VIDEOMODE_FORMAT7 and the frame rate will be FRAMERATE_FORMAT7.

Parameters

<i>context</i>	The fc2Context to be used.
<i>videoMode</i>	Current video mode.
<i>frameRate</i>	Current frame rate.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.74 FLYCAPTURE2_C_API fc2Error fc2GetVideoModeAndFrameRateInfo (fc2Context context, fc2VideoMode videoMode, fc2FrameRate frameRate, BOOL * pSupported)

Query the camera to determine if the specified video mode and frame rate is supported.

Parameters

<i>context</i>	The fc2Context to be used.
<i>videoMode</i>	Video mode to check.
<i>frameRate</i>	Frame rate to check.
<i>pSupported</i>	Whether the video mode and frame rate is supported.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.75 FLYCAPTURE2_C_API fc2Error fc2H264Open (fc2AVIContext *AVIContext*, const char * *pFileName*, fc2H264Option * *pOption*)

Open an H.264 file in preparation for writing Images to disk.

The size of AVI files is limited to 2GB. The filenames are automatically generated using the filename specified.

Parameters

<i>AVIContext</i>	The AVI context to use.
<i>pFileName</i>	The filename of the AVI file.
<i>pOption</i>	Options to apply to the AVI file.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.76 FLYCAPTURE2_C_API fc2Error fc2IsCameraControlable (fc2Context *context*, fc2PGRGuid * *pGuid*, BOOL * *pControlable*)

Query whether a GigE camera is controlable.

Parameters

<i>context</i>	The fc2Context to be used.
<i>pGuid</i>	Unique PGRGuid for the camera.
<i>pControlable</i>	True indicates camera is controllable

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.77 FLYCAPTURE2_C_API fc2Error fc2LaunchBrowser (const char * *pAddress*)

Launch a URL in the system default browser.

Parameters

<i>pAddress</i>	URL to open in browser.
-----------------	-------------------------

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.78 FLYCAPTURE2_C_API `fc2Error` `fc2LaunchCommand` (`const char *` *pCommand*)

Execute a command in the terminal.

This is a blocking call that will return when the command completes.

Parameters

<i>pCommand</i>	Command to execute.
-----------------	---------------------

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.79 FLYCAPTURE2_C_API `fc2Error` `fc2LaunchCommandAsync` (`const char *` *pCommand*, `fc2AsyncCommandCallback` *pCallback*, `void *` *pUserData*)

Execute a command in the terminal.

This is a non-blocking call that will return immediately. The return value of the command can be retrieved in the callback.

Parameters

<i>pCommand</i>	Command to execute.
<i>pCallback</i>	Callback to fire when command is complete.
<i>pUserData</i>	Data pointer to pass to callback.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.80 FLYCAPTURE2_C_API `fc2Error` `fc2LaunchHelp` (`const char *` *pFileName*)

Open a CHM file in the system default CHM viewer.

Parameters

<i>pFileName</i>	Filename of CHM file to open.
------------------	-------------------------------

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.81 FLYCAPTURE2_C_API fc2Error fc2MJPEGOpen (fc2AVIContext *AVIContext*, const char * *pFileName*, fc2MJPEGOption * *pOption*)

Open an MJPEG file in preparation for writing Images to disk.

The size of AVI files is limited to 2GB. The filenames are automatically generated using the filename specified.

Parameters

<i>AVIContext</i>	The AVI context to use.
<i>pFileName</i>	The filename of the AVI file.
<i>pOption</i>	Options to apply to the AVI file.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.82 FLYCAPTURE2_C_API fc2Error fc2QueryGigElmagingMode (fc2Context *context*, fc2Mode *mode*, BOOL * *isSupported*)

4.1.1.83 FLYCAPTURE2_C_API fc2Error fc2ReadGVCPMemory (fc2Context *context*, unsigned int *address*, unsigned char * *pBuffer*, unsigned int *length*)

Read a GVCP memory block.

Parameters

<i>context</i>	The fc2Context to be used.
<i>address</i>	GVCP address to be read from.
<i>pBuffer</i>	Array containing data to be written.
<i>length</i>	Size of array, in quadlets.

Returns

An Error indicating the success or failure of the function.

4.1.1.84 FLYCAPTURE2_C_API fc2Error fc2ReadGVCPRegister (fc2Context *context*, unsigned int *address*, unsigned int * *pValue*)

Read a GVCP register.

Parameters

<i>context</i>	The fc2Context to be used.
<i>address</i>	GVCP address to be read from.
<i>pValue</i>	The value that is read.

Returns

An Error indicating the success or failure of the function.

4.1.1.85 FLYCAPTURE2_C_API fc2Error fc2ReadGVCPRegisterBlock (fc2Context context, unsigned int address, unsigned int * pBuffer, unsigned int length)

Read a GVCP register block.

Parameters

<i>context</i>	The fc2Context to be used.
<i>address</i>	GVCP address to be read from.
<i>pBuffer</i>	Array containing data to be written.
<i>length</i>	Size of array, in quadlets.

Returns

An Error indicating the success or failure of the function.

4.1.1.86 FLYCAPTURE2_C_API fc2Error fc2ReadRegister (fc2Context context, unsigned int address, unsigned int * pValue)

Read the specified register from the camera.

Parameters

<i>context</i>	The fc2Context to be used.
<i>address</i>	DCAM address to be read from.
<i>pValue</i>	The value that is read.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.87 FLYCAPTURE2_C_API fc2Error fc2ReadRegisterBlock (fc2Context context, unsigned short addressHigh, unsigned int addressLow, unsigned int * pBuffer, unsigned int length)

Write to the specified register block on the camera.

Parameters

<i>context</i>	The fc2Context to be used.
<i>addressHigh</i>	Top 16 bits of the 48 bit absolute address to read from.
<i>addressLow</i>	Bottom 32 bits of the 48 bits absolute address to read from.
<i>pBuffer</i>	Array to store read data.
<i>length</i>	Size of array, in quadlets.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.88 FLYCAPTURE2_C API `fc2Error fc2RegisterCallback (fc2Context context, fc2BusEventCallback enumCallback, fc2BusCallbackType callbackType, void * pParameter, fc2CallbackHandle * pCallbackHandle)`

Register a callback function that will be called when the specified callback event occurs.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>enum-Callback</i>	Pointer to function that will receive the callback.
<i>callbackType</i>	Type of callback to register for.
<i>pParameter</i>	Callback parameter to be passed to callback.
<i>pCallback-Handle</i>	Unique callback handle used for unregistering callback.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.89 FLYCAPTURE2_C API `fc2Error fc2RescanBus (fc2Context context)`

Force a rescan of the buses.

This does not trigger a bus reset. However, any current connections to a Camera object will be invalidated.

Returns

An Error indicating the success or failure of the function.

4.1.1.90 FLYCAPTURE2_C API `fc2Error fc2RestoreFromMemoryChannel (fc2Context context, unsigned int channel)`

Restore the specified current memory channel.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>channel</i>	Memory channel to restore from.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.91 FLYCAPTURE2_C_API `fc2Error` `fc2RetrieveBuffer` (`fc2Context` *context*, `fc2Image` * *pImage*)

Retrieves the the next image object containing the next image.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>pImage</i>	Pointer to fc2Image to store image data.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.92 FLYCAPTURE2_C_API `fc2Error` `fc2SaveImage` (`fc2Image` * *pImage*, `const char` * *pFilename*, `fc2ImageFileFormat` *format*)

Save the image to the specified file name with the file format specified.

Parameters

<i>pImage</i>	The fc2Image to be used.
<i>pFilename</i>	Filename to save image with.
<i>format</i>	File format to save in.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.93 FLYCAPTURE2_C_API `fc2Error` `fc2SaveImageWithOptions` (`fc2Image` * *pImage*, `const char` * *pFilename*, `fc2ImageFileFormat` *format*, `void` * *pOption*)

Save the image to the specified file name with the file format specified.

Parameters

<i>pImage</i>	The fc2Image to be used.
<i>pFilename</i>	Filename to save image with.
<i>format</i>	File format to save in.
<i>pOption</i>	Options for saving image.

Returns

A `fc2Error` indicating the success or failure of the function.

**4.1.1.94 FLYCAPTURE2_C_API `fc2Error` `fc2SaveToMemoryChannel` (`fc2Context` *context*,
unsigned int *channel*)**

Save the current settings to the specified current memory channel.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>channel</i>	Memory channel to save to.

Returns

A `fc2Error` indicating the success or failure of the function.

**4.1.1.95 FLYCAPTURE2_C_API `fc2Error` `fc2SetActiveLUTBank` (`fc2Context` *context*,
unsigned int *activeBank*)**

Set the LUT bank that will be used.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>activeBank</i>	The bank to be set as active.

Returns

A `fc2Error` indicating the success or failure of the function.

**4.1.1.96 FLYCAPTURE2_C_API `fc2Error` `fc2SetCallback` (`fc2Context` *context*,
`fc2ImageEventCallback` *pCallbackFn*, void * *pCallbackData*)**

Sets the callback data to be used on completion of image transfer.

To clear the current stored callback data, pass in NULL for both callback arguments.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>pCallbackFn</i>	A function to be called when a new image is received.
<i>pCallback-Data</i>	A pointer to data that can be passed to the callback function.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.97 FLYCAPTURE2_C_API `fc2Error` `fc2SetChannelStatus` (`fc2ImageStatistics-Context` *imageStatisticsContext*, `fc2StatisticsChannel` *channel*, `BOOL` *enabled*)

Set the status of a statistics channel.

Parameters

<i>image-Statistics-Context</i>	A statistics context.
<i>channel</i>	The statistics channel.
<i>enabled</i>	Whether the channel should be enabled.

See also

`GetChannelStatus()`

Returns

An Error indicating the success or failure of the function.

4.1.1.98 FLYCAPTURE2_C_API `fc2Error` `fc2SetConfiguration` (`fc2Context` *context*, `fc2Config *` *config*)

Set the configuration associated with the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>config</i>	Pointer to the configuration structure to be used.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.99 FLYCAPTURE2_C_API `fc2Error` `fc2SetDefaultColorProcessing` (`fc2ColorProcessingAlgorithm` *defaultMethod*)

Set the default color processing algorithm.

This method will be used for any image with the DEFAULT algorithm set. The method used is determined at the time of the `Convert()` call, therefore the most recent execution

of this function will take precedence. The default setting is shared within the current process.

Parameters

<i>default-Method</i>	The color processing algorithm to set.
-----------------------	--

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.100 FLYCAPTURE2_C_API `fc2Error fc2SetDefaultOutputFormat (fc2PixelFormat format)`

Set the default output pixel format.

This format will be used for any call to `Convert()` that does not specify an output format. The format used will be determined at the time of the `Convert()` call, therefore the most recent execution of this function will take precedence. The default is shared within the current process.

Parameters

<i>format</i>	The output pixel format to set.
---------------	---------------------------------

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.101 FLYCAPTURE2_C_API `fc2Error fc2SetEmbeddedImageInfo (fc2Context context, fc2EmbeddedImageInfo * pInfo)`

Sets the on/off values of the embedded image information structure to the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>pInfo</i>	Structure to be used.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.102 FLYCAPTURE2_C_API `fc2Error` `fc2SetFormat7Configuration` (`fc2Context` *context*, `fc2Format7ImageSettings` * *imageSettings*, `float` *percentSpeed*)

Set the current Format7 configuration to the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>image-Settings</i>	Image settings to be written to the camera.
<i>percent-Speed</i>	Packet size as a percentage to be written to the camera.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.103 FLYCAPTURE2_C_API `fc2Error` `fc2SetFormat7ConfigurationPacket` (`fc2Context` *context*, `fc2Format7ImageSettings` * *imageSettings*, `unsigned int` *packetSize*)

Set the current Format7 configuration to the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>image-Settings</i>	Image settings to be written to the camera.
<i>packetSize</i>	Packet size to be written to the camera.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.104 FLYCAPTURE2_C_API `fc2Error` `fc2SetGigEConfig` (`fc2Context` *context*, `const` `fc2GigEConfig` * *pConfig*)

4.1.1.105 FLYCAPTURE2_C_API `fc2Error` `fc2SetGigEImageBinningSettings` (`fc2Context` *context*, `unsigned int` *horzBinningValue*, `unsigned int` *vertBinningValue*)

4.1.1.106 FLYCAPTURE2_C_API `fc2Error` `fc2SetGigEImageSettings` (`fc2Context` *context*, `const` `fc2GigEImageSettings` * *pImageSettings*)

4.1.1.107 **FLYCAPTURE2_C_API fc2Error fc2SetGigElmagingMode (fc2Context *context*, fc2Mode *mode*)**

4.1.1.108 **FLYCAPTURE2_C_API fc2Error fc2SetGigEProperty (fc2Context *context*, const fc2GigEProperty * *pGigEProp*)**

Set the specified GigEProperty.

The GigEPropertyType field must be set in order for this function to succeed.

Parameters

<i>context</i>	The fc2Context to be used.
<i>pGigEProp</i>	The GigE property to set.

Returns

An Error indicating the success or failure of the function.

4.1.1.109 **FLYCAPTURE2_C_API fc2Error fc2SetGigEStreamChannelInfo (fc2Context *context*, unsigned int *channel*, fc2GigEStreamChannel * *pChannel*)**

4.1.1.110 **FLYCAPTURE2_C_API fc2Error fc2SetGPIOPinDirection (fc2Context *context*, unsigned int *pin*, unsigned int *direction*)**

Set the GPIO pin direction for the specified pin.

This is useful if there is a need to set the pin into an input pin (i.e. to read the voltage) off the pin without setting it as a trigger source. This is not a required call when using the trigger or strobe functions as the pin direction is set automatically internally.

Parameters

<i>context</i>	The fc2Context to be used.
<i>pin</i>	Pin to get the direction for.
<i>direction</i>	Direction of the pin. 0 for input, 1 for output.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.111 **FLYCAPTURE2_C_API fc2Error fc2SetGPIOPinDirectionBroadcast (fc2Context *context*, unsigned int *pin*, unsigned int *direction*)**

Set the GPIO pin direction for the specified pin.

This is useful if there is a need to set the pin into an input pin (i.e. to read the voltage) off the pin without setting it as a trigger source. This is not a required call when using the trigger or strobe functions as the pin direction is set automatically internally.

Parameters

<i>context</i>	The fc2Context to be used.
<i>pin</i>	Pin to get the direction for.
<i>direction</i>	Direction of the pin. 0 for input, 1 for output.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.112 FLYCAPTURE2_C_API fc2Error fc2SetImageData (fc2Image * *plImage*, const unsigned char * *pData*, unsigned int *dataSize*)

Set the data of the Image object.

Ownership of the image buffer is not transferred to the Image object. It is the user's responsibility to delete the buffer when it is no longer in use.

Parameters

<i>plImage</i>	The fc2Image to be used.
<i>pData</i>	Pointer to the image buffer.
<i>dataSize</i>	Size of the image buffer.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.113 FLYCAPTURE2_C_API fc2Error fc2SetImageDimensions (fc2Image * *plImage*, unsigned int *rows*, unsigned int *cols*, unsigned int *stride*, fc2PixelFormat *pixelFormat*, fc2BayerTileFormat *bayerFormat*)

Sets the dimensions of the image object.

Parameters

<i>plImage</i>	The fc2Image to be used.
<i>rows</i>	Number of rows to set.
<i>cols</i>	Number of cols to set.
<i>stride</i>	Stride to set.
<i>pixelFormat</i>	Pixel format to set.
<i>bayerFormat</i>	Bayer tile format to set.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.114 **FLYCAPTURE2_C_API** **fc2Error** **fc2SetLUTChannel** (**fc2Context** *context*, unsigned int *bank*, unsigned int *channel*, unsigned int *sizeEntries*, unsigned int * *pEntries*)

Set the LUT channel settings to the camera.

Parameters

<i>context</i>	The fc2Context to be used.
<i>bank</i>	Bank to set.
<i>channel</i>	Channel to set.
<i>sizeEntries</i>	Number of entries in LUT table to write. This must be the same size as numEntries returned by GetLutInfo().
<i>pEntries</i>	Array containing LUT entries to write.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.115 **FLYCAPTURE2_C_API** **fc2Error** **fc2SetProperty** (**fc2Context** *context*, **fc2Property** * *prop*)

Writes the settings for the specified property to the camera.

The property type must be specified in the Property structure passed into the function in order for the function to succeed. The absControl flag controls whether the absolute or integer value is written to the camera.

Parameters

<i>context</i>	The fc2Context to be used.
<i>prop</i>	Pointer to the Property structure to be used.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.116 **FLYCAPTURE2_C_API** **fc2Error** **fc2SetPropertyBroadcast** (**fc2Context** *context*, **fc2Property** * *prop*)

Writes the settings for the specified property to the camera.

The property type must be specified in the Property structure passed into the function in order for the function to succeed. The absControl flag controls whether the absolute or integer value is written to the camera.

Parameters

<i>context</i>	The fc2Context to be used.
<i>prop</i>	Pointer to the Property structure to be used.

Returns

A `fc2Error` indicating the success or failure of the function.

**4.1.1.117 FLYCAPTURE2_C_API `fc2Error` `fc2SetStrobe` (`fc2Context` *context*,
`fc2StrobeControl` * *strobeControl*)**

Set current strobe settings to the camera.

The strobe pin must be specified in the structure before being passed in to the function.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>strobe-Control</i>	Structure providing strobe settings.

Returns

A `fc2Error` indicating the success or failure of the function.

**4.1.1.118 FLYCAPTURE2_C_API `fc2Error` `fc2SetStrobeBroadcast` (`fc2Context` *context*,
`fc2StrobeControl` * *strobeControl*)**

Set current strobe settings to the camera.

The strobe pin must be specified in the structure before being passed in to the function.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>strobe-Control</i>	Structure providing strobe settings.

Returns

A `fc2Error` indicating the success or failure of the function.

**4.1.1.119 FLYCAPTURE2_C_API `fc2Error` `fc2SetTriggerDelay` (`fc2Context` *context*,
`fc2TriggerDelay` * *triggerDelay*)**

Set the specified trigger delay settings to the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>triggerDelay</i>	Structure providing trigger delay settings.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.120 FLYCAPTURE2_C_API `fc2Error` `fc2SetTriggerDelayBroadcast` (`fc2Context` *context*, `fc2TriggerDelay` * *triggerDelay*)

Set the specified trigger delay settings to the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>triggerDelay</i>	Structure providing trigger delay settings.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.121 FLYCAPTURE2_C_API `fc2Error` `fc2SetTriggerMode` (`fc2Context` *context*, `fc2TriggerMode` * *triggerMode*)

Set the specified trigger settings to the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>triggerMode</i>	Structure providing trigger mode settings.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.122 FLYCAPTURE2_C_API `fc2Error` `fc2SetTriggerModeBroadcast` (`fc2Context` *context*, `fc2TriggerMode` * *triggerMode*)

Set the specified trigger settings to the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>triggerMode</i>	Structure providing trigger mode settings.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.123 FLYCAPTURE2_C_API **fc2Error** **fc2SetUserBuffers** (**fc2Context** *context*, unsigned char *const *ppMemBuffers*, int *size*, int *nNumBuffers*)

Specify user allocated buffers to use as image data buffers.

Parameters

<i>context</i>	The fc2Context to be used.
<i>ppMem- Buffers</i>	Pointer to memory buffers to be written to. The size of the data should be equal to (size * numBuffers) or larger.
<i>size</i>	The size of each buffer (in bytes).
<i>nNum- Buffers</i>	Number of buffers in the array.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.124 FLYCAPTURE2_C_API **fc2Error** **fc2SetVideoModeAndFrameRate** (**fc2Context** *context*, **fc2VideoMode** *videoMode*, **fc2FrameRate** *frameRate*)

Set the specified video mode and frame rate to the camera.

It is not possible to set the camera to VIDEOMODE_FORMAT7 or FRAMERATE_FORMAT7. Use the Format7 functions to set the camera into Format7.

Parameters

<i>context</i>	The fc2Context to be used.
<i>videoMode</i>	Video mode to set to camera.
<i>frameRate</i>	Frame rate to set to camera.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.125 FLYCAPTURE2_C_API **fc2Error** **fc2StartCapture** (**fc2Context** *context*)

Starts isochronous image capture.

It will use either the current video mode or the most recently set video mode of the camera.

Parameters

<i>context</i>	The fc2Context to be used.
----------------	----------------------------

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.126 FLYCAPTURE2_C_API `fc2Error` `fc2StartCaptureCallback` (`fc2Context` *context*, `fc2ImageEventCallback` *pCallbackFn*, `void *` *pCallbackData*)

Starts isochronous image capture.

It will use either the current video mode or the most recently set video mode of the camera. The callback function is called when a new image is received from the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>pCallbackFn</i>	A function to be called when a new image is received.
<i>pCallbackData</i>	A pointer to data that can be passed to the callback function. A NULL pointer is acceptable.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.127 FLYCAPTURE2_C_API `fc2Error` `fc2StartSyncCapture` (`unsigned int` *numCameras*, `fc2Context *` *pContexts*)

Starts synchronized isochronous image capture on multiple cameras.

Parameters

<i>numCameras</i>	Number of <code>fc2Contexts</code> in the <code>ppCameras</code> array.
<i>pContexts</i>	Array of <code>fc2Contexts</code> .

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.128 FLYCAPTURE2_C_API `fc2Error` `fc2StartSyncCaptureCallback` (`unsigned int` *numCameras*, `fc2Context *` *pContexts*, `fc2ImageEventCallback *` *pCallbackFns*, `void **` *pCallbackDataArray*)

Starts synchronized isochronous image capture on multiple cameras.

Parameters

<i>num-Cameras</i>	Number of fc2Contexts in the ppCameras array.
<i>pContexts</i>	Array of fc2Contexts.
<i>pCallback-Fns</i>	Array of callback functions for each camera.
<i>pCallback-DataArray</i>	Array of callback data pointers.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.129 FLYCAPTURE2_C_API fc2Error fc2StopCapture (fc2Context *context*)

Stops isochronous image transfer and cleans up all associated resources.

Parameters

<i>context</i>	The fc2Context to be used.
----------------	----------------------------

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.130 FLYCAPTURE2_C_API fc2Error fc2UnregisterCallback (fc2Context *context*,
fc2CallbackHandle *callbackHandle*)

Unregister a callback function.

Parameters

<i>context</i>	The fc2Context to be used.
<i>callback-Handle</i>	Unique callback handle.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.131 FLYCAPTURE2_C_API fc2Error fc2ValidateFormat7Settings (fc2Context *context*,
fc2Format7ImageSettings * *imageSettings*, BOOL * *settingsAreValid*,
fc2Format7PacketInfo * *packetInfo*)

Validates Format7ImageSettings structure and returns valid packet size information if the image settings are valid.

The current image settings are cached while validation is taking place. The cached settings are restored when validation is complete.

Parameters

<i>context</i>	The fc2Context to be used.
<i>image-Settings</i>	Structure containing the image settings.
<i>settingsAre-Valid</i>	Whether the settings are valid.
<i>packetInfo</i>	Packet size information that can be used to determine a valid packet size.

Returns

A fc2Error indicating the success or failure of the function.

4.1.1.132 FLYCAPTURE2_C_API fc2Error fc2WriteGVCPMemory (fc2Context context, unsigned int address, const unsigned char * pBuffer, unsigned int length)

Write a GVCP memory block.

Parameters

<i>context</i>	The fc2Context to be used.
<i>address</i>	GVCP address to be write to.
<i>pBuffer</i>	Array containing data to be written.
<i>length</i>	Size of array, in quadlets.

Returns

An Error indicating the success or failure of the function.

4.1.1.133 FLYCAPTURE2_C_API fc2Error fc2WriteGVCPRegister (fc2Context context, unsigned int address, unsigned int value)

Write a GVCP register.

Parameters

<i>context</i>	The fc2Context to be used.
<i>address</i>	GVCP address to be written to.
<i>value</i>	The value to be written.

Returns

An Error indicating the success or failure of the function.

4.1.1.134 FLYCAPTURE2_C_API fc2Error fc2WriteGVCPRegisterBlock (fc2Context *context*, unsigned int *address*, const unsigned int * *pBuffer*, unsigned int *length*)

Write a GVCP register block.

Parameters

<i>context</i>	The fc2Context to be used.
<i>address</i>	GVCP address to be write to.
<i>pBuffer</i>	Array containing data to be written.
<i>length</i>	Size of array, in quadlets.

Returns

An Error indicating the success or failure of the function.

4.1.1.135 FLYCAPTURE2_C_API fc2Error fc2WriteGVCPRegisterBroadcast (fc2Context *context*, unsigned int *address*, unsigned int *value*)

Write a GVCP register with broadcast.

Parameters

<i>context</i>	The fc2Context to be used.
<i>address</i>	GVCP address to be written to.
<i>value</i>	The value to be written.

Returns

An Error indicating the success or failure of the function.

4.1.1.136 FLYCAPTURE2_C_API fc2Error fc2WriteRegister (fc2Context *context*, unsigned int *address*, unsigned int *value*)

Write to the specified register on the camera.

Parameters

<i>context</i>	The fc2Context to be used.
<i>address</i>	DCAM address to be written to.
<i>value</i>	The value to be written.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.137 FLYCAPTURE2.C API `fc2Error fc2WriteRegisterBlock (fc2Context context, unsigned short addressHigh, unsigned int addressLow, const unsigned int * pBuffer, unsigned int length)`

Write to the specified register block on the camera.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>addressHigh</i>	Top 16 bits of the 48 bit absolute address to write to.
<i>addressLow</i>	Bottom 32 bits of the 48 bits absolute address to write to.
<i>pBuffer</i>	Array containing data to be written.
<i>length</i>	Size of array, in quadlets.

Returns

A `fc2Error` indicating the success or failure of the function.

4.1.1.138 FLYCAPTURE2.C API `fc2Error fc2WriteRegisterBroadcast (fc2Context context, unsigned int address, unsigned int value)`

Write to the specified register on the camera with broadcast.

Parameters

<i>context</i>	The <code>fc2Context</code> to be used.
<i>address</i>	DCAM address to be written to.
<i>value</i>	The value to be written.

Returns

A `fc2Error` indicating the success or failure of the function.

4.2 FlyCapture2Defs_C.h File Reference

Data Structures

- struct [fc2PGRGuid](#)
A GUID to the camera.
- struct [fc2Image](#)
- struct [fc2SystemInfo](#)

- struct [fc2Version](#)
- struct [fc2Config](#)
- struct [fc2TriggerDelayInfo](#)
- struct [fc2TriggerDelay](#)
- struct [fc2TriggerModeInfo](#)
- struct [fc2TriggerMode](#)
- struct [fc2StrobeInfo](#)
- struct [fc2StrobeControl](#)
- struct [fc2Format7ImageSettings](#)
- struct [fc2Format7Info](#)
- struct [fc2Format7PacketInfo](#)
- struct [fc2IPAddress](#)
- struct [fc2MACAddress](#)
- struct [fc2GigEProperty](#)
- struct [fc2GigEStreamChannel](#)
- struct [fc2GigEConfig](#)
- struct [fc2GigEImageSettingsInfo](#)
- struct [fc2GigEImageSettings](#)
- struct [fc2TimeStamp](#)
- struct [fc2ConfigROM](#)
- struct [fc2CameraInfo](#)
- struct [fc2EmbeddedImageInfoProperty](#)
- struct [fc2EmbeddedImageInfo](#)
- struct [fc2ImageMetadata](#)
- struct [fc2LUTData](#)
- struct [fc2PNGOption](#)
- struct [fc2PPMOption](#)
- struct [fc2PGMOption](#)
- struct [fc2TIFFOption](#)
- struct [fc2JPEGOption](#)
- struct [fc2JPG2Option](#)
- struct [fc2AVIOption](#)
- struct [fc2MJPGOption](#)
- struct [fc2H264Option](#)

Defines

- #define [FALSE](#) 0
- #define [TRUE](#) 1
- #define [FULL_32BIT_VALUE](#) 0x7FFFFFFF
- #define [MAX_STRING_LENGTH](#) 512

Typedefs

- typedef int [BOOL](#)
- typedef void * [fc2Context](#)
A context to the FlyCapture2 C library.
- typedef void * [fc2GuiContext](#)
A context to the FlyCapture2 C GUI library.
- typedef void * [fc2ImageImpl](#)
An internal pointer used in the [fc2Image](#) structure.
- typedef void * [fc2AVIContext](#)
A context referring to the AVI recorder object.
- typedef void * [fc2ImageStatisticsContext](#)
A context referring to the ImageStatistics object.
- typedef void * [fc2CallbackHandle](#)
- typedef void(* [fc2BusEventCallback](#))(void *pParameter, unsigned int serial-Number)
- typedef void(* [fc2ImageEventCallback](#))(fc2Image *image, void *pCallbackData)
- typedef void(* [fc2AsyncCommandCallback](#))(fc2Error retError, void *pUserData)

Enumerations

- enum [fc2Error](#) { [FC2_ERROR_UNDEFINED](#) = -1, [FC2_ERROR_OK](#), [FC2_ERROR_FAILED](#), [FC2_ERROR_NOT_IMPLEMENTED](#), [FC2_ERROR_FAILED_BUS_MASTER_CONNECTION](#), [FC2_ERROR_NOT_CONNECTED](#), [FC2_ERROR_INIT_FAILED](#), [FC2_ERROR_NOT_INITIALIZED](#), [FC2_ERROR_INVALID_PARAMETER](#), [FC2_ERROR_INVALID_SETTINGS](#), [FC2_ERROR_INVALID_BUS_MANAGER](#), [FC2_ERROR_MEMORY_ALLOCATION_FAILED](#), [FC2_ERROR_LOW_LEVEL_FAILURE](#), [FC2_ERROR_NOT_FOUND](#), [FC2_ERROR_FAILED_GUID](#), [FC2_ERROR_INVALID_PACKET_SIZE](#), [FC2_ERROR_INVALID_MODE](#), [FC2_ERROR_NOT_IN_FORMAT7](#), [FC2_ERROR_NOT_SUPPORTED](#), [FC2_ERROR_TIMEOUT](#), [FC2_ERROR_BUS_MASTER_FAILED](#), [FC2_ERROR_INVALID_GENERATION](#), [FC2_ERROR_LUT_FAILED](#), [FC2_ERROR_IIDC_FAILED](#), [FC2_ERROR_STROBE_FAILED](#), [FC2_ERROR_TRIGGER_FAILED](#), [FC2_ERROR_PROPERTY_FAILED](#), [FC2_ERROR_PROPERTY_NOT_PRESENT](#), [FC2_ERROR_REGISTER_FAILED](#), [FC2_ERROR_READ_REGISTER_FAILED](#), [FC2_ERROR_WRITE_REGISTER_FAILED](#), [FC2_ERROR_ISOCH_FAILED](#), [FC2_ERROR_ISOCH_ALREADY_STARTED](#), [FC2_ERROR_ISOCH_NOT_STARTED](#), [FC2_ERROR_ISOCH_START_FAILED](#), [FC2_ERROR_ISOCH_RETRIEVE_BUFFER_FAILED](#), [FC2_ERROR_ISOCH_STOP_FAILED](#), [FC2_ERROR_ISOCH_SYNC_FAILED](#), [FC2_ERROR_ISOCH_BANDWIDTH_EXCEEDED](#), [FC2_ERROR_IMAGE_CONVERSION_FAILED](#), [FC2_ERROR_IMAGE_LIBRARY_FAILURE](#), [FC2_ERROR_BUFFER_TOO_SMALL](#), [FC2_ERROR_IMAGE_CONSISTENCY_ERROR](#), [FC2_ERROR_FORCE_32BITS](#) = [FULL_32BIT_VALUE](#) }
- enum [fc2BusCallbackType](#) { [FC2_BUS_RESET](#), [FC2_ARRIVAL](#), [FC2_REMOVAL](#), [FC2_CALLBACK_TYPE_FORCE_32BITS](#) = [FULL_32BIT_VALUE](#) }

- enum `fc2GrabMode` { `FC2_DROP_FRAMES`, `FC2_BUFFER_FRAMES`, `FC2_UNSPECIFIED_GRAB_MODE`, `FC2_GRAB_MODE_FORCE_32BITS` = `FULL_32BIT_VALUE` }
 - enum `fc2GrabTimeout` { `FC2_TIMEOUT_NONE` = 0, `FC2_TIMEOUT_INFINITE` = -1, `FC2_TIMEOUT_UNSPECIFIED` = -2, `FC2_GRAB_TIMEOUT_FORCE_32BITS` = `FULL_32BIT_VALUE` }
 - enum `fc2BandwidthAllocation` { `FC2_BANDWIDTH_ALLOCATION_OFF` = 0, `FC2_BANDWIDTH_ALLOCATION_ON` = 1, `FC2_BANDWIDTH_ALLOCATION_UNSUPPORTED` = 2, `FC2_BANDWIDTH_ALLOCATION_UNSPECIFIED` = 3, `FC2_BANDWIDTH_ALLOCATION_FORCE_32BITS` = `FULL_32BIT_VALUE` }
 - enum `fc2InterfaceType` { `FC2_INTERFACE_IEEE1394`, `FC2_INTERFACE_USB_2`, `FC2_INTERFACE_USB_3`, `FC2_INTERFACE_GIGE`, `FC2_INTERFACE_UNKNOWN`, `FC2_INTERFACE_TYPE_FORCE_32BITS` = `FULL_32BIT_VALUE` }
 - enum `fc2DriverType` { `FC2_DRIVER_1394_CAM`, `FC2_DRIVER_1394_PRO`, `FC2_DRIVER_1394_JUJU`, `FC2_DRIVER_1394_VIDEO1394`, `FC2_DRIVER_1394_RAW1394`, `FC2_DRIVER_USB_NONE`, `FC2_DRIVER_USB_CAM`, `FC2_DRIVER_USB3_PRO`, `FC2_DRIVER_GIGE_NONE`, `FC2_DRIVER_GIGE_FILTER`, `FC2_DRIVER_GIGE_PRO`, `FC2_DRIVER_UNKNOWN` = -1, `FC2_DRIVER_FORCE_32BITS` = `FULL_32BIT_VALUE` }
- Types of low level drivers that flycapture uses.*
- enum `fc2PropertyType` { `FC2_BRIGHTNESS`, `FC2_AUTO_EXPOSURE`, `FC2_SHARPNESS`, `FC2_WHITE_BALANCE`, `FC2_HUE`, `FC2_SATURATION`, `FC2_GAMMA`, `FC2_IRIS`, `FC2_FOCUS`, `FC2_ZOOM`, `FC2_PAN`, `FC2_TILT`, `FC2_SHUTTER`, `FC2_GAIN`, `FC2_TRIGGER_MODE`, `FC2_TRIGGER_DELAY`, `FC2_FRAME_RATE`, `FC2_TEMPERATURE`, `FC2_UNSPECIFIED_PROPERTY_TYPE`, `FC2_PROPERTY_TYPE_FORCE_32BITS` = `FULL_32BIT_VALUE` }
 - enum `fc2FrameRate` { `FC2_FRAMERATE_1_875`, `FC2_FRAMERATE_3_75`, `FC2_FRAMERATE_7_5`, `FC2_FRAMERATE_15`, `FC2_FRAMERATE_30`, `FC2_FRAMERATE_60`, `FC2_FRAMERATE_120`, `FC2_FRAMERATE_240`, `FC2_FRAMERATE_FORMAT7`, `FC2_NUM_FRAMERATES`, `FC2_FRAMERATE_FORCE_32BITS` = `FULL_32BIT_VALUE` }
 - enum `fc2VideoMode` { `FC2_VIDEOMODE_160x120YUV444`, `FC2_VIDEOMODE_320x240YUV422`, `FC2_VIDEOMODE_640x480YUV411`, `FC2_VIDEOMODE_640x480YUV422`, `FC2_VIDEOMODE_640x480RGB`, `FC2_VIDEOMODE_640x480Y8`, `FC2_VIDEOMODE_640x480Y16`, `FC2_VIDEOMODE_800x600YUV422`, `FC2_VIDEOMODE_800x600RGB`, `FC2_VIDEOMODE_800x600Y8`, `FC2_VIDEOMODE_800x600Y16`, `FC2_VIDEOMODE_1024x768YUV422`, `FC2_VIDEOMODE_1024x768RGB`, `FC2_VIDEOMODE_1024x768Y8`, `FC2_VIDEOMODE_1024x768Y16`, `FC2_VIDEOMODE_1280x960YUV422`, `FC2_VIDEOMODE_1280x960RGB`, `FC2_VIDEOMODE_1280x960Y8`, `FC2_VIDEOMODE_1280x960Y16`, `FC2_VIDEOMODE_1600x1200YUV422`, `FC2_VIDEOMODE_1600x1200RGB`, `FC2_VIDEOMODE_1600x1200Y8`, `FC2_VIDEOMODE_1600x1200Y16`, `FC2_VIDEOMODE_FORMAT7`, `FC2_NUM_VIDEOMODES`, `FC2_VIDEOMODE_FORCE_32BITS` = `FULL_32BIT_VALUE` }
 - enum `fc2Mode` { `FC2_MODE_0` = 0, `FC2_MODE_1`, `FC2_MODE_2`, `FC2_MODE_3`, `FC2_MODE_4`, `FC2_MODE_5`, `FC2_MODE_6`, `FC2_MODE_7`,

```

FC2_MODE_8, FC2_MODE_9, FC2_MODE_10, FC2_MODE_11, FC2_MODE_12, FC2_MODE_13, FC2_MODE_14, FC2_MODE_15, FC2_MODE_16,
FC2_MODE_17, FC2_MODE_18, FC2_MODE_19, FC2_MODE_20, FC2_MODE_21, FC2_MODE_22, FC2_MODE_23, FC2_MODE_24, FC2_MODE_25,
FC2_MODE_26, FC2_MODE_27, FC2_MODE_28, FC2_MODE_29, FC2_MODE_30, FC2_MODE_31, FC2_NUM_MODES, FC2_MODE_FORCE_32BITS
= FULL_32BIT_VALUE }
• enum fc2PixelFormat { FC2_PIXEL_FORMAT_MONO8 = 0x80000000, FC2_PIXEL_FORMAT_411YUV8 = 0x40000000, FC2_PIXEL_FORMAT_422YUV8 = 0x20000000, FC2_PIXEL_FORMAT_444YUV8 = 0x10000000, FC2_PIXEL_FORMAT_RGB8 = 0x08000000, FC2_PIXEL_FORMAT_MONO16 = 0x04000000, FC2_PIXEL_FORMAT_RGB16 = 0x02000000, FC2_PIXEL_FORMAT_S_MONO16 = 0x01000000, FC2_PIXEL_FORMAT_S_RGB16 = 0x00800000, FC2_PIXEL_FORMAT_RAW8 = 0x00400000, FC2_PIXEL_FORMAT_RAW16 = 0x00200000, FC2_PIXEL_FORMAT_MONO12 = 0x00100000, FC2_PIXEL_FORMAT_RAW12 = 0x00080000, FC2_PIXEL_FORMAT_BGR = 0x80000008, FC2_PIXEL_FORMAT_BGRU = 0x40000008, FC2_PIXEL_FORMAT_RGB = FC2_PIXEL_FORMAT_RGB8, FC2_PIXEL_FORMAT_RGBU = 0x40000002, FC2_PIXEL_FORMAT_BGR16 = 0x02000001, FC2_PIXEL_FORMAT_BGRU16 = 0x02000002, FC2_PIXEL_FORMAT_422YUV8_JPEG = 0x40000001, FC2_NUM_PIXEL_FORMATS = 20, FC2_UNSPECIFIED_PIXEL_FORMAT = 0 }
• enum fc2BusSpeed { FC2_BUSSPEED_S100, FC2_BUSSPEED_S200, FC2_BUSSPEED_S400, FC2_BUSSPEED_S480, FC2_BUSSPEED_S800, FC2_BUSSPEED_S1600, FC2_BUSSPEED_S3200, FC2_BUSSPEED_S5000, FC2_BUSSPEED_10BASE_T, FC2_BUSSPEED_100BASE_T, FC2_BUSSPEED_1000BASE_T, FC2_BUSSPEED_10000BASE_T, FC2_BUSSPEED_S_FASTEST, FC2_BUSSPEED_ANY, FC2_BUSSPEED_SPEED_UNKNOWN = -1, FC2_BUSSPEED_FORCE_32BITS = FULL_32BIT_VALUE }
• enum fc2PCIEBusSpeed { FC2_PCIE_BUSSPEED_2_5, FC2_PCIE_BUSSPEED_5_0, FC2_PCIE_BUSSPEED_UNKNOWN = -1, FC2_PCIE_BUSSPEED_FORCE_32BITS = FULL_32BIT_VALUE }
• enum fc2ColorProcessingAlgorithm { FC2_DEFAULT, FC2_NO_COLOR_PROCESSING, FC2_NEAREST_NEIGHBOR_FAST, FC2_EDGE_SENSING, FC2_HQ_LINEAR, FC2_RIGOROUS, FC2_IPP, FC2_DIRECTIONAL, FC2_COLOR_PROCESSING_ALGORITHM_FORCE_32BITS = FULL_32BIT_VALUE }
• enum fc2BayerTileFormat { FC2_BT_NONE, FC2_BT_RGGB, FC2_BT_GRBG, FC2_BT_GBRG, FC2_BT_BGGR, FC2_BT_FORCE_32BITS = FULL_32BIT_VALUE }
• enum fc2ImageFileFormat { FC2_FROM_FILE_EXT = -1, FC2_PGM, FC2_PPM, FC2_BMP, FC2_JPEG, FC2_JPEG2000, FC2_TIFF, FC2_PNG, FC2_RAW, FC2_IMAGE_FILE_FORMAT_FORCE_32BITS = FULL_32BIT_VALUE }
• enum fc2GigEPropertyType { FC2_HEARTBEAT, FC2_HEARTBEAT_TIMEOUT }
• enum fc2StatisticsChannel { FC2_STATISTICS_GREY, FC2_STATISTICS_RED, FC2_STATISTICS_GREEN, FC2_STATISTICS_BLUE, FC2_STATISTICS_HUE, FC2_STATISTICS_SATURATION, FC2_STATISTICS_LIGHTNESS, FC2_STATISTICS_FORCE_32BITS = FULL_32BIT_VALUE }

```

- enum `fc2OSType` { `FC2_WINDOWS_X86`, `FC2_WINDOWS_X64`, `FC2_LINUX_X86`, `FC2_LINUX_X64`, `FC2_MAC`, `FC2_UNKNOWN_OS`, `FC2_OSTYPE_FORCE_32BITS` = `FULL_32BIT_VALUE` }
- enum `fc2ByteOrder` { `FC2_BYTE_ORDER_LITTLE_ENDIAN`, `FC2_BYTE_ORDER_BIG_ENDIAN`, `FC2_BYTE_ORDER_FORCE_32BITS` = `FULL_32BIT_VALUE` }
- enum `fc2TIFFCompressionMethod` { `FC2_TIFF_NONE` = 1, `FC2_TIFF_PACKBITS`, `FC2_TIFF_DEFLATE`, `FC2_TIFF_ADOBE_DEFLATE`, `FC2_TIFF_CCITT3`, `FC2_TIFF_CCITT4`, `FC2_TIFF_LZW`, `FC2_TIFF_JPEG` }

4.2.1 Define Documentation

4.2.1.1 `#define FALSE 0`

4.2.1.2 `#define FULL_32BIT_VALUE 0x7FFFFFFF`

4.2.1.3 `#define MAX_STRING_LENGTH 512`

4.2.1.4 `#define TRUE 1`

4.2.2 Typedef Documentation

4.2.2.1 `typedef int BOOL`

4.2.2.2 `typedef void(* fc2AsyncCommandCallback)(fc2Error retError, void *pUserData)`

4.2.2.3 `typedef void* fc2AVIContext`

A context referring to the AVI recorder object.

4.2.2.4 `typedef void(* fc2BusEventCallback)(void *pParameter, unsigned int serialNumber)`

4.2.2.5 `typedef void* fc2CallbackHandle`

4.2.2.6 `typedef void* fc2Context`

A context to the FlyCapture2 C library.

It must be created before performing any calls to the library.

4.2.2.7 `typedef void* fc2GuiContext`

A context to the FlyCapture2 C GUI library.

It must be created before performing any calls to the library.

4.2.2.8 `typedef void(* fc2ImageEventCallback)(fc2Image *image, void *pCallbackData)`

4.2.2.9 `typedef void* fc2ImageImpl`

An internal pointer used in the [fc2Image](#) structure.

4.2.2.10 `typedef void* fc2ImageStatisticsContext`

A context referring to the ImageStatistics object.

4.2.3 Enumeration Type Documentation

4.2.3.1 `enum fc2BandwidthAllocation`

Enumerator:

FC2_BANDWIDTH_ALLOCATION_OFF
FC2_BANDWIDTH_ALLOCATION_ON
FC2_BANDWIDTH_ALLOCATION_UNSUPPORTED
FC2_BANDWIDTH_ALLOCATION_UNSPECIFIED
FC2_BANDWIDTH_ALLOCATION_FORCE_32BITS

4.2.3.2 `enum fc2BayerTileFormat`

Enumerator:

FC2_BT_NONE No bayer tile format.
FC2_BT_RGGB Red-Green-Green-Blue.
FC2_BT_GRBG Green-Red-Blue-Green.
FC2_BT_GBRG Green-Blue-Red-Green.
FC2_BT_BGGR Blue-Green-Green-Red.
FC2_BT_FORCE_32BITS

4.2.3.3 `enum fc2BusCallbackType`

Enumerator:

FC2_BUS_RESET
FC2_ARRIVAL
FC2_REMOVAL
FC2_CALLBACK_TYPE_FORCE_32BITS

4.2.3.4 enum fc2BusSpeed

Enumerator:

FC2_BUSSPEED_S100 100Mbps/sec.
FC2_BUSSPEED_S200 200Mbps/sec.
FC2_BUSSPEED_S400 400Mbps/sec.
FC2_BUSSPEED_S480 480Mbps/sec. Only for USB2 cameras.
FC2_BUSSPEED_S800 800Mbps/sec.
FC2_BUSSPEED_S1600 1600Mbps/sec.
FC2_BUSSPEED_S3200 3200Mbps/sec.
FC2_BUSSPEED_S5000 5000Mbps/sec. Only for USB3 cameras.
FC2_BUSSPEED_10BASE_T 10Base-T. Only for GigE cameras.
FC2_BUSSPEED_100BASE_T 100Base-T. Only for GigE cameras.
FC2_BUSSPEED_1000BASE_T 1000Base-T (Gigabit Ethernet). Only for GigE cameras.
FC2_BUSSPEED_10000BASE_T 10000Base-T. Only for GigE cameras.
FC2_BUSSPEED_S_FASTEST The fastest speed available.
FC2_BUSSPEED_ANY Any speed that is available.
FC2_BUSSPEED_SPEED_UNKNOWN Unknown bus speed.
FC2_BUSSPEED_FORCE_32BITS

4.2.3.5 enum fc2ByteOrder

Enumerator:

FC2_BYTE_ORDER_LITTLE_ENDIAN
FC2_BYTE_ORDER_BIG_ENDIAN
FC2_BYTE_ORDER_FORCE_32BITS

4.2.3.6 enum fc2ColorProcessingAlgorithm

Enumerator:

FC2_DEFAULT
FC2_NO_COLOR_PROCESSING
FC2_NEAREST_NEIGHBOR_FAST
FC2_EDGE_SENSING
FC2_HQ_LINEAR
FC2_RIGOROUS
FC2_IPP
FC2_DIRECTIONAL
FC2_COLOR_PROCESSING_ALGORITHM_FORCE_32BITS

4.2.3.7 enum fc2DriverType

Types of low level drivers that flycapture uses.

Enumerator:

FC2_DRIVER_1394_CAM PGRCam.sys.
FC2_DRIVER_1394_PRO PGR1394.sys.
FC2_DRIVER_1394_JUJU firewire_core.
FC2_DRIVER_1394_VIDEO1394 video1394.
FC2_DRIVER_1394_RAW1394 raw1394.
FC2_DRIVER_USB_NONE No usb driver used just BSD stack. (Linux only)
FC2_DRIVER_USB_CAM PGRUsbCam.sys.
FC2_DRIVER_USB3_PRO PGRXHCl.sys.
FC2_DRIVER_GIGE_NONE no gige drivers used,MS/BSD stack.
FC2_DRIVER_GIGE_FILTER PGRGigE.sys.
FC2_DRIVER_GIGE_PRO PGRGigEPro.sys.
FC2_DRIVER_UNKNOWN Unknown driver type.
FC2_DRIVER_FORCE_32BITS

4.2.3.8 enum fc2Error

Enumerator:

FC2_ERROR_UNDEFINED Undefined.
FC2_ERROR_OK Function returned with no errors.
FC2_ERROR_FAILED General failure.
FC2_ERROR_NOT_IMPLEMENTED Function has not been implemented.
FC2_ERROR_FAILED_BUS_MASTER_CONNECTION Could not connect to -
Bus Master.
FC2_ERROR_NOT_CONNECTED Camera has not been connected.
FC2_ERROR_INIT_FAILED Initialization failed.
FC2_ERROR_NOT_INTIALIZED Camera has not been initialized.
FC2_ERROR_INVALID_PARAMETER Invalid parameter passed to function.
FC2_ERROR_INVALID_SETTINGS Setting set to camera is invalid.
FC2_ERROR_INVALID_BUS_MANAGER Invalid Bus Manager object.
FC2_ERROR_MEMORY_ALLOCATION_FAILED Could not allocate memory.
FC2_ERROR_LOW_LEVEL_FAILURE Low level error.
FC2_ERROR_NOT_FOUND Device not found.
FC2_ERROR_FAILED_GUID GUID failure.
FC2_ERROR_INVALID_PACKET_SIZE Packet size set to camera is invalid.

FC2_ERROR_INVALID_MODE Invalid mode has been passed to function.

FC2_ERROR_NOT_IN_FORMAT7 Error due to not being in Format7.

FC2_ERROR_NOT_SUPPORTED This feature is unsupported.

FC2_ERROR_TIMEOUT Timeout error.

FC2_ERROR_BUS_MASTER_FAILED Bus Master Failure.

FC2_ERROR_INVALID_GENERATION Generation Count Mismatch.

FC2_ERROR_LUT_FAILED Look Up Table failure.

FC2_ERROR_IIDC_FAILED IIDC failure.

FC2_ERROR_STROBE_FAILED Strobe failure.

FC2_ERROR_TRIGGER_FAILED Trigger failure.

FC2_ERROR_PROPERTY_FAILED Property failure.

FC2_ERROR_PROPERTY_NOT_PRESENT Property is not present.

FC2_ERROR_REGISTER_FAILED Register access failed.

FC2_ERROR_READ_REGISTER_FAILED Register read failed.

FC2_ERROR_WRITE_REGISTER_FAILED Register write failed.

FC2_ERROR_ISOCH_FAILED Isochronous failure.

FC2_ERROR_ISOCH_ALREADY_STARTED Isochronous transfer has already been started.

FC2_ERROR_ISOCH_NOT_STARTED Isochronous transfer has not been started.

FC2_ERROR_ISOCH_START_FAILED Isochronous start failed.

FC2_ERROR_ISOCH_RETRIEVE_BUFFER_FAILED Isochronous retrieve buffer failed.

FC2_ERROR_ISOCH_STOP_FAILED Isochronous stop failed.

FC2_ERROR_ISOCH_SYNC_FAILED Isochronous image synchronization failed.

FC2_ERROR_ISOCH_BANDWIDTH_EXCEEDED Isochronous bandwidth exceeded.

FC2_ERROR_IMAGE_CONVERSION_FAILED Image conversion failed.

FC2_ERROR_IMAGE_LIBRARY_FAILURE Image library failure.

FC2_ERROR_BUFFER_TOO_SMALL Buffer is too small.

FC2_ERROR_IMAGE_CONSISTENCY_ERROR There is an image consistency error.

FC2_ERROR_FORCE_32BITS

4.2.3.9 enum fc2FrameRate

Enumerator:

FC2_FRAMERATE_1_875 1.875 fps.
FC2_FRAMERATE_3_75 3.75 fps.
FC2_FRAMERATE_7_5 7.5 fps.
FC2_FRAMERATE_15 15 fps.
FC2_FRAMERATE_30 30 fps.
FC2_FRAMERATE_60 60 fps.
FC2_FRAMERATE_120 120 fps.
FC2_FRAMERATE_240 240 fps.
FC2_FRAMERATE_FORMAT7 Custom frame rate for Format7 functionality.
FC2_NUM_FRAMERATES Number of possible camera frame rates.
FC2_FRAMERATE_FORCE_32BITS

4.2.3.10 enum fc2GigEPropertyType

Enumerator:

FC2_HEARTBEAT
FC2_HEARTBEAT_TIMEOUT

4.2.3.11 enum fc2GrabMode

Enumerator:

FC2_DROP_FRAMES
FC2_BUFFER_FRAMES
FC2_UNSPECIFIED_GRAB_MODE
FC2_GRAB_MODE_FORCE_32BITS

4.2.3.12 enum fc2GrabTimeout

Enumerator:

FC2_TIMEOUT_NONE
FC2_TIMEOUT_INFINITE
FC2_TIMEOUT_UNSPECIFIED
FC2_GRAB_TIMEOUT_FORCE_32BITS

4.2.3.13 enum fc2ImageFileFormat

Enumerator:

FC2_FROM_FILE_EXT Determine file format from file extension.

FC2_PGM Portable gray map.

FC2_PPM Portable pixmap.

FC2_BMP Bitmap.

FC2_JPEG JPEG.

FC2_JPEG2000 JPEG 2000.

FC2_TIFF Tagged image file format.

FC2_PNG Portable network graphics.

FC2_RAW Raw data.

FC2_IMAGE_FILE_FORMAT_FORCE_32BITS

4.2.3.14 enum fc2InterfaceType

Enumerator:

FC2_INTERFACE_IEEE1394

FC2_INTERFACE_USB_2

FC2_INTERFACE_USB_3

FC2_INTERFACE_GIGE

FC2_INTERFACE_UNKNOWN

FC2_INTERFACE_TYPE_FORCE_32BITS

4.2.3.15 enum fc2Mode

Enumerator:

FC2_MODE_0

FC2_MODE_1

FC2_MODE_2

FC2_MODE_3

FC2_MODE_4

FC2_MODE_5

FC2_MODE_6

FC2_MODE_7

FC2_MODE_8

FC2_MODE_9

FC2_MODE_10

FC2_MODE_11
FC2_MODE_12
FC2_MODE_13
FC2_MODE_14
FC2_MODE_15
FC2_MODE_16
FC2_MODE_17
FC2_MODE_18
FC2_MODE_19
FC2_MODE_20
FC2_MODE_21
FC2_MODE_22
FC2_MODE_23
FC2_MODE_24
FC2_MODE_25
FC2_MODE_26
FC2_MODE_27
FC2_MODE_28
FC2_MODE_29
FC2_MODE_30
FC2_MODE_31
FC2_NUM_MODES Number of modes.
FC2_MODE_FORCE_32BITS

4.2.3.16 enum fc2OSType

Enumerator:

FC2_WINDOWS_X86
FC2_WINDOWS_X64
FC2_LINUX_X86
FC2_LINUX_X64
FC2_MAC
FC2_UNKNOWN_OS
FC2_OSTYPE_FORCE_32BITS

4.2.3.17 enum fc2PCleBusSpeed

Enumerator:

FC2_PCIE_BUSSPEED_2_5
FC2_PCIE_BUSSPEED_5_0 2.5 Gb/s
FC2_PCIE_BUSSPEED_UNKNOWN 5.0 Gb/s
FC2_PCIE_BUSSPEED_FORCE_32BITS Speed is unknown.

4.2.3.18 enum fc2PixelFormat

Enumerator:

FC2_PIXEL_FORMAT_MONO8 8 bits of mono information.
FC2_PIXEL_FORMAT_411YUV8 YUV 4:1:1.
FC2_PIXEL_FORMAT_422YUV8 YUV 4:2:2.
FC2_PIXEL_FORMAT_444YUV8 YUV 4:4:4.
FC2_PIXEL_FORMAT_RGB8 R = G = B = 8 bits.
FC2_PIXEL_FORMAT_MONO16 16 bits of mono information.
FC2_PIXEL_FORMAT_RGB16 R = G = B = 16 bits.
FC2_PIXEL_FORMAT_S_MONO16 16 bits of signed mono information.
FC2_PIXEL_FORMAT_S_RGB16 R = G = B = 16 bits signed.
FC2_PIXEL_FORMAT_RAW8 8 bit raw data output of sensor.
FC2_PIXEL_FORMAT_RAW16 16 bit raw data output of sensor.
FC2_PIXEL_FORMAT_MONO12 12 bits of mono information.
FC2_PIXEL_FORMAT_RAW12 12 bit raw data output of sensor.
FC2_PIXEL_FORMAT_BGR 24 bit BGR.
FC2_PIXEL_FORMAT_BGRU 32 bit BGRU.
FC2_PIXEL_FORMAT_RGB 24 bit RGB.
FC2_PIXEL_FORMAT_RGBU 32 bit RGBU.
FC2_PIXEL_FORMAT_BGR16 R = G = B = 16 bits.
FC2_PIXEL_FORMAT_BGRU16 64 bit BGRU.
FC2_PIXEL_FORMAT_422YUV8_JPEG JPEG compressed stream.
FC2_NUM_PIXEL_FORMATS Number of pixel formats.
FC2_UNSPECIFIED_PIXEL_FORMAT Unspecified pixel format.

4.2.3.19 enum fc2PropertyType

Enumerator:

FC2_BRIGHTNESS
FC2_AUTO_EXPOSURE
FC2_SHARPNESS
FC2_WHITE_BALANCE
FC2_HUE
FC2_SATURATION
FC2_GAMMA
FC2_IRIS
FC2_FOCUS
FC2_ZOOM
FC2_PAN
FC2_TILT
FC2_SHUTTER
FC2_GAIN
FC2_TRIGGER_MODE
FC2_TRIGGER_DELAY
FC2_FRAME_RATE
FC2_TEMPERATURE
FC2_UNSPECIFIED_PROPERTY_TYPE
FC2_PROPERTY_TYPE_FORCE_32BITS

4.2.3.20 enum fc2StatisticsChannel

Enumerator:

FC2_STATISTICS_GREY
FC2_STATISTICS_RED
FC2_STATISTICS_GREEN
FC2_STATISTICS_BLUE
FC2_STATISTICS_HUE
FC2_STATISTICS_SATURATION
FC2_STATISTICS_LIGHTNESS
FC2_STATISTICS_FORCE_32BITS

4.2.3.21 enum fc2TIFFCompressionMethod

Enumerator:

FC2_TIFF_NONE
FC2_TIFF_PACKBITS
FC2_TIFF_DEFLATE
FC2_TIFF_ADOBE_DEFLATE
FC2_TIFF_CCITTFAX3
FC2_TIFF_CCITTFAX4
FC2_TIFF_LZW
FC2_TIFF_JPEG

4.2.3.22 enum fc2VideoMode

Enumerator:

FC2_VIDEOMODE_160x120YUV444 160x120 YUV444.
FC2_VIDEOMODE_320x240YUV422 320x240 YUV422.
FC2_VIDEOMODE_640x480YUV411 640x480 YUV411.
FC2_VIDEOMODE_640x480YUV422 640x480 YUV422.
FC2_VIDEOMODE_640x480RGB 640x480 24-bit RGB.
FC2_VIDEOMODE_640x480Y8 640x480 8-bit.
FC2_VIDEOMODE_640x480Y16 640x480 16-bit.
FC2_VIDEOMODE_800x600YUV422 800x600 YUV422.
FC2_VIDEOMODE_800x600RGB 800x600 RGB.
FC2_VIDEOMODE_800x600Y8 800x600 8-bit.
FC2_VIDEOMODE_800x600Y16 800x600 16-bit.
FC2_VIDEOMODE_1024x768YUV422 1024x768 YUV422.
FC2_VIDEOMODE_1024x768RGB 1024x768 RGB.
FC2_VIDEOMODE_1024x768Y8 1024x768 8-bit.
FC2_VIDEOMODE_1024x768Y16 1024x768 16-bit.
FC2_VIDEOMODE_1280x960YUV422 1280x960 YUV422.
FC2_VIDEOMODE_1280x960RGB 1280x960 RGB.
FC2_VIDEOMODE_1280x960Y8 1280x960 8-bit.
FC2_VIDEOMODE_1280x960Y16 1280x960 16-bit.
FC2_VIDEOMODE_1600x1200YUV422 1600x1200 YUV422.
FC2_VIDEOMODE_1600x1200RGB 1600x1200 RGB.
FC2_VIDEOMODE_1600x1200Y8 1600x1200 8-bit.
FC2_VIDEOMODE_1600x1200Y16 1600x1200 16-bit.
FC2_VIDEOMODE_FORMAT7 Custom video mode for Format7 functionality.
FC2_NUM_VIDEOMODES Number of possible video modes.
FC2_VIDEOMODE_FORCE_32BITS

4.3 FlyCapture2GUI_C.h File Reference

Functions

- FLYCAPTURE2_C_API [fc2Error](#) [fc2CreateGUIContext](#) ([fc2GuiContext](#) *pContext)
Create a GUI context.
- FLYCAPTURE2_C_API [fc2Error](#) [fc2DestroyGUIContext](#) ([fc2GuiContext](#) context)
Destroy a GUI context.
- FLYCAPTURE2_C_API void [fc2GUIConnect](#) ([fc2GuiContext](#) context, [fc2Context](#) cameraContext)
Connect GUI context to a camera context.
- FLYCAPTURE2_C_API void [fc2Disonnect](#) ([fc2GuiContext](#) context)
Disconnect GUI context from camera.
- FLYCAPTURE2_C_API void [fc2Show](#) ([fc2GuiContext](#) context)
Show the GUI.
- FLYCAPTURE2_C_API void [fc2Hide](#) ([fc2GuiContext](#) context)
Hide the GUI.
- FLYCAPTURE2_C_API [BOOL](#) [fc2IsVisible](#) ([fc2GuiContext](#) context)
Check if the GUI is visible.
- FLYCAPTURE2_C_API void [fc2ShowModal](#) ([fc2GuiContext](#) context, [BOOL](#) *pOkSelected, [fc2PGRGuid](#) *guidArray, unsigned int *size)
Show the camera selection dialog.

4.3.1 Function Documentation

4.3.1.1 FLYCAPTURE2_C_API [fc2Error](#) [fc2CreateGUIContext](#) ([fc2GuiContext](#) * pContext)

Create a GUI context.

Parameters

pContext	Pointer to context to be created.
--------------------------	-----------------------------------

Returns

An Error indicating the success or failure of the function.

4.3.1.2 FLYCAPTURE2_C_API [fc2Error](#) [fc2DestroyGUIContext](#) ([fc2GuiContext](#) context)

Destroy a GUI context.

Parameters

<i>context</i>	Context to be destroyed.
----------------	--------------------------

Returns

An Error indicating the success or failure of the function.

4.3.1.3 FLYCAPTURE2_C_API void fc2Disconnect (fc2GuiContext *context*)

Disconnect GUI context from camera.

Parameters

<i>context</i>	GUI context to disconnect.
----------------	----------------------------

Returns

An Error indicating the success or failure of the function.

4.3.1.4 FLYCAPTURE2_C_API void fc2GUIConnect (fc2GuiContext *context*, fc2Context *cameraContext*)

Connect GUI context to a camera context.

Parameters

<i>context</i>	GUI context to connect.
<i>camera-Context</i>	Camera context to connect.

Returns

An Error indicating the success or failure of the function.

4.3.1.5 FLYCAPTURE2_C_API void fc2Hide (fc2GuiContext *context*)

Hide the GUI.

Parameters

<i>context</i>	Pointer to context to hide.
----------------	-----------------------------

Returns

An Error indicating the success or failure of the function.

4.3.1.6 FLYCAPTURE2.C_API **BOOL** fc2IsVisible (**fc2GuiContext** *context*)

Check if the GUI is visible.

Parameters

<i>context</i>	Pointer to context to show.
----------------	-----------------------------

Returns

Whether the GUI is visible.

4.3.1.7 FLYCAPTURE2.C_API **void** fc2Show (**fc2GuiContext** *context*)

Show the GUI.

Parameters

<i>context</i>	Pointer to context to show.
----------------	-----------------------------

Returns

An Error indicating the success or failure of the function.

4.3.1.8 FLYCAPTURE2.C_API **void** fc2ShowModal (**fc2GuiContext** *context*, **BOOL** * *pOkSelected*, **fc2PGRGuid** * *guidArray*, **unsigned int** * *size*)

Show the camera selection dialog.

Parameters

<i>context</i>	Pointer to context to show.
<i>pOkSelected</i>	Whether Ok (true) or Cancel (false) was clicked.
<i>guidArray</i>	Array of PGRGuids containing the selected cameras.
<i>size</i>	Size of PGRGuid array.

4.4 FlyCapture2Internal.C.h File Reference

Data Structures

- struct [fc2InternalContext](#)
- struct [fc2InternalGuiContext](#)
- struct [fc2InternalImageCallback](#)

Functions

- bool [IsContextValid](#) (fc2Context context)
- bool [IsGuiContextValid](#) (fc2GuiContext context)
- void [SyncCpplImageToStruct](#) (fc2Image *pImage)

4.4.1 Function Documentation

4.4.1.1 bool [IsContextValid](#) (fc2Context *context*) [inline]

4.4.1.2 bool [IsGuiContextValid](#) (fc2GuiContext *context*) [inline]

4.4.1.3 void [SyncCpplImageToStruct](#) (fc2Image * *pImage*) [inline]

4.5 FlyCapture2Platform_C.h File Reference

Defines

- #define [FLYCAPTURE2_C_API](#)
- #define [FLYCAPTURE2_C_CALL_CONVEN](#)

4.5.1 Define Documentation

4.5.1.1 #define [FLYCAPTURE2_C_API](#)

4.5.1.2 #define [FLYCAPTURE2_C_CALL_CONVEN](#)

4.6 MultiSyncLibrary_C.h File Reference

Functions

- MULTISYNCLIBRARY_C_API [syncError](#) [syncCreateContext](#) ([syncContext](#) *p-Context)
Create a Sync context for MultiSync Library.
- MULTISYNCLIBRARY_C_API [syncError](#) [syncDestroyContext](#) ([syncContext](#) context)
Destory the sync context.
- MULTISYNCLIBRARY_C_API [syncError](#) [syncStart](#) ([syncContext](#) context)
Start the sync progress.
- MULTISYNCLIBRARY_C_API [syncError](#) [syncStop](#) ([syncContext](#) context)
Stop the sync progress.
- MULTISYNCLIBRARY_C_API [syncError](#) [syncRescanMasterTimingBus](#) ([sync-Context](#) context)
Scan newly connected or removed timing bus (for corss-PC syncing only)

- MULTISYNCLIBRARY_C_API [syncMessage](#) [syncGetStatus](#) ([syncContext](#) context)
Start the sync progress.
- MULTISYNCLIBRARY_C_API double [syncGetTimeSinceSynced](#) ([syncContext](#) context)
Time since sync started.
- MULTISYNCLIBRARY_C_API bool [syncIsTimingBusConnected](#) ([syncContext](#) context)
Whether syncing across PCs.
- MULTISYNCLIBRARY_C_API bool [syncEnableCrossPCsSynchronization](#) ([syncContext](#) context)
Enable across pc synchronization support.
- MULTISYNCLIBRARY_C_API bool [syncDisableCrossPCsSynchronization](#) ([syncContext](#) context)
Disable across pc synchronization support.
- MULTISYNCLIBRARY_C_API bool [syncQueryCrossPCsSynchronizationSetting](#) ([syncContext](#) context)
Query cross pc synchronizaion support status.

4.6.1 Function Documentation

4.6.1.1 MULTISYNCLIBRARY_C_API [syncError](#) [syncCreateContext](#) ([syncContext](#) * [pContext](#))

Create a Sync context for MultiSync Library.

This call must be made before any other calls that use a context will succeed.

Parameters

pContext	A pointer to the syncContext to be created.
--------------------------	---

Returns

A [syncError](#) indicating the success or failure of the function.

4.6.1.2 MULTISYNCLIBRARY_C_API [syncError](#) [syncDestroyContext](#) ([syncContext](#) [context](#))

Destory the sync context.

This must be called when the user is finished with the context in order to prevent memory leaks.

Parameters

context	The syncContext to be destroyed.
-------------------------	--

Returns

A syncError indicating the success or failure of the function.

4.6.1.3 MULTISYNCLIBRARY_C_API bool syncDisableCrossPCSynchronization (syncContext context)

Disable across pc synchronization support.

Parameters

<i>context</i>	The syncContext to be used.
----------------	-----------------------------

Returns

True if operation was successful

4.6.1.4 MULTISYNCLIBRARY_C_API bool syncEnableCrossPCSynchronization (syncContext context)

Enable across pc synchronization support.

Parameters

<i>context</i>	The syncContext to be used.
----------------	-----------------------------

Returns

True if operation was successful

4.6.1.5 MULTISYNCLIBRARY_C_API syncMessage syncGetStatus (syncContext context)

Start the sync progress.

Parameters

<i>context</i>	The syncContext to be used.
----------------	-----------------------------

Returns

A syncMessage indicating the sync status.

4.6.1.6 MULTISYNCLIBRARY_C_API double syncGetTimeSinceSynced (syncContext context)

Time since sync started.

Parameters

<i>context</i>	The syncContext to be used.
----------------	-----------------------------

Returns

Time since sync started.

4.6.1.7 MULTISYNCLIBRARY_C_API bool syncIsTimingBusConnected (syncContext context)

Whether syncing across PCs.

Parameters

<i>context</i>	The syncContext to be used.
----------------	-----------------------------

Returns

True if its syncing across PC

4.6.1.8 MULTISYNCLIBRARY_C_API bool syncQueryCrossPCsSynchronizationSetting (syncContext context)

Query cross pc synchronizaion support status.

Parameters

<i>context</i>	The syncContext to be used.
----------------	-----------------------------

Returns

True if cross pc synchronization was supported

4.6.1.9 MULTISYNCLIBRARY_C_API syncError syncRescanMasterTimingBus (syncContext context)

Scan newly connected or removed timing bus (for corss-PC syncing only)

Parameters

<i>context</i>	The syncContext to be used.
----------------	-----------------------------

Returns

A syncError indicating the success or failure of the function.

4.6.1.10 MULTISYNCLIBRARY_C_API syncError syncStart (syncContext context)

Start the sync progress.

Parameters

<i>context</i>	The syncContext to be used.
----------------	-----------------------------

Returns

A syncError indicating the success or failure of the function.

4.6.1.11 MULTISYNCLIBRARY_C_API syncError syncStop (syncContext context)

Stop the sync progress.

Parameters

<i>context</i>	The syncContext to be used.
----------------	-----------------------------

Returns

A syncError indicating the success or failure of the function.

4.7 MultiSyncLibraryDefs_C.h File Reference

Defines

- `#define FALSE 0`
- `#define TRUE 1`
- `#define FULL_32BIT_VALUE 0x7FFFFFFF`
- `#define MAX_STRING_LENGTH 512`

Typedefs

- `typedef int BOOL`
- `typedef void * syncContext`
A context to the MultiSync C library.

Enumerations

- `enum syncError { SYNC_ERROR_OK = 0, SYNC_ERROR_FAILED, SYNC_ERROR_ALREADY_STARTED, SYNC_ERROR_ALREADY_STOPPED, SYNC_ERROR_CONTEXT_NOT_INITIALIZED, SYNC_ERROR_UNKNOWN_ERROR }`
- `enum syncMessage { SYNC_MESSAGE_OK = 0, SYNC_MESSAGE_FAILED, SYNC_MESSAGE_STARTED, SYNC_MESSAGE_STOPPED, SYNC_MESSAGE_SYNCING, SYNC_MESSAGE_NOMASTER, SYNC_MESSAGE_THREAD_ERROR, SYNC_MESSAGE_DEVICE_ERROR, SYNC_MESSAGE_NOT_ENOUGH_DEVICES, SYNC_MESSAGE_BUS_RESET, SYNC_MESSAGE_NOT_INITIALIZED, SYNC_MESSAGE_UNKNOWN_ERROR }`

4.7.1 Define Documentation

4.7.1.1 `#define FALSE 0`

4.7.1.2 `#define FULL_32BIT_VALUE 0x7FFFFFFF`

4.7.1.3 `#define MAX_STRING_LENGTH 512`

4.7.1.4 `#define TRUE 1`

4.7.2 Typedef Documentation

4.7.2.1 `typedef int BOOL`

4.7.2.2 `typedef void* syncContext`

A context to the MultiSync C library.

It must be created before performing any calls to the library.

4.7.3 Enumeration Type Documentation

4.7.3.1 enum syncError

Enumerator:

SYNC_ERROR_OK
SYNC_ERROR_FAILED
SYNC_ERROR_ALREADY_STARTED
SYNC_ERROR_ALREADY_STOPPED
SYNC_ERROR_CONTEXT_NOT_INITIALIZED
SYNC_ERROR_UNKNOWN_ERROR

4.7.3.2 enum syncMessage

Enumerator:

SYNC_MESSAGE_OK
SYNC_MESSAGE_FAILED
SYNC_MESSAGE_STARTED
SYNC_MESSAGE_STOPPED
SYNC_MESSAGE_SYNCING
SYNC_MESSAGE_NOMASTER
SYNC_MESSAGE_THREAD_ERROR
SYNC_MESSAGE_DEVICE_ERROR
SYNC_MESSAGE_NOT_ENOUGH_DEVICES
SYNC_MESSAGE_BUS_RESET
SYNC_MESSAGE_NOT_INITIALIZED
SYNC_MESSAGE_UNKNOWN_ERROR

4.8 MultiSyncLibraryPlatform_C.h File Reference

Defines

- #define [MULTISYNCLIBRARY_C_API](#)
- #define [MULTISYNCLIBRARY_C_CALL_CONVEN](#)

4.8.1 Define Documentation

4.8.1.1 #define MULTISYNCLIBRARY_C_API

4.8.1.2 #define MULTISYNCLIBRARY_C_CALL_CONVEN

Index

FC2_ARRIVAL
FlyCapture2Defs_C.h, [96](#)

FC2_AUTO_EXPOSURE
FlyCapture2Defs_C.h, [104](#)

FC2_BANDWIDTH_ALLOCATION_FORCE_32BITS
FlyCapture2Defs_C.h, [96](#)

FC2_BANDWIDTH_ALLOCATION_OFF
FlyCapture2Defs_C.h, [96](#)

FC2_BANDWIDTH_ALLOCATION_ON
FlyCapture2Defs_C.h, [96](#)

FC2_BANDWIDTH_ALLOCATION_UNSPECIFIED
FlyCapture2Defs_C.h, [96](#)

FC2_BANDWIDTH_ALLOCATION_UNSUPPORTED
FlyCapture2Defs_C.h, [96](#)

FC2_BMP
FlyCapture2Defs_C.h, [101](#)

FC2_BRIGHTNESS
FlyCapture2Defs_C.h, [104](#)

FC2_BT_BGGR
FlyCapture2Defs_C.h, [96](#)

FC2_BT_FORCE_32BITS
FlyCapture2Defs_C.h, [96](#)

FC2_BT_GBRG
FlyCapture2Defs_C.h, [96](#)

FC2_BT_GRBG
FlyCapture2Defs_C.h, [96](#)

FC2_BT_NONE
FlyCapture2Defs_C.h, [96](#)

FC2_BT_RGGB
FlyCapture2Defs_C.h, [96](#)

FC2_BUFFER_FRAMES
FlyCapture2Defs_C.h, [100](#)

FC2_BUSSPEED_10000BASE_T
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_1000BASE_T
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_100BASE_T
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_10BASE_T
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_ANY
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_FORCE_32BITS
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_S100
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_S1600
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_S200
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_S3200
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_S400
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_S480
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_S5000
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_S800
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_SPEED_UNKNOWN
FlyCapture2Defs_C.h, [97](#)

FC2_BUSSPEED_S_FASTEST
FlyCapture2Defs_C.h, [97](#)

FC2_BUS_RESET
FlyCapture2Defs_C.h, [96](#)

FC2_BYTE_ORDER_BIG_ENDIAN
FlyCapture2Defs_C.h, [97](#)

FC2_BYTE_ORDER_FORCE_32BITS
FlyCapture2Defs_C.h, [97](#)

FC2_BYTE_ORDER_LITTLE_ENDIAN
FlyCapture2Defs_C.h, [97](#)

FC2_CALLBACK_TYPE_FORCE_32BITS
FlyCapture2Defs_C.h, [96](#)

FC2_COLOR_PROCESSING_ALGORITHM_FORCE_32BITS
FlyCapture2Defs_C.h, [97](#)

FC2_DEFAULT

- FlyCapture2Defs_C.h, [97](#)
- FC2_DIRECTIONAL
 - FlyCapture2Defs_C.h, [97](#)
- FC2_DRIVER_1394_CAM
 - FlyCapture2Defs_C.h, [98](#)
- FC2_DRIVER_1394_JUJU
 - FlyCapture2Defs_C.h, [98](#)
- FC2_DRIVER_1394_PRO
 - FlyCapture2Defs_C.h, [98](#)
- FC2_DRIVER_1394_RAW1394
 - FlyCapture2Defs_C.h, [98](#)
- FC2_DRIVER_1394_VIDEO1394
 - FlyCapture2Defs_C.h, [98](#)
- FC2_DRIVER_FORCE_32BITS
 - FlyCapture2Defs_C.h, [98](#)
- FC2_DRIVER_GIGE_FILTER
 - FlyCapture2Defs_C.h, [98](#)
- FC2_DRIVER_GIGE_NONE
 - FlyCapture2Defs_C.h, [98](#)
- FC2_DRIVER_GIGE_PRO
 - FlyCapture2Defs_C.h, [98](#)
- FC2_DRIVER_UNKNOWN
 - FlyCapture2Defs_C.h, [98](#)
- FC2_DRIVER_USB3_PRO
 - FlyCapture2Defs_C.h, [98](#)
- FC2_DRIVER_USB_CAM
 - FlyCapture2Defs_C.h, [98](#)
- FC2_DRIVER_USB_NONE
 - FlyCapture2Defs_C.h, [98](#)
- FC2_DROP_FRAMES
 - FlyCapture2Defs_C.h, [100](#)
- FC2_EDGE_SENSING
 - FlyCapture2Defs_C.h, [97](#)
- FC2_ERROR_BUFFER_TOO_SMALL
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_BUS_MASTER_FAILED
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_FAILED
 - FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_FAILED_BUS_MASTER_CONNECTION
 - FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_FAILED_GUID
 - FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_FORCE_32BITS
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_IIDC_FAILED
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_IMAGE_CONSISTENCY_ERROR
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_IMAGE_CONVERSION_FAILED
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_IMAGE_LIBRARY_FAILURE
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_INIT_FAILED
 - FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_INVALID_BUS_MANAGER
 - FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_INVALID_GENERATION
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_INVALID_MODE
 - FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_INVALID_PACKET_SIZE
 - FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_INVALID_PARAMETER
 - FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_INVALID_SETTINGS
 - FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_ISOCH_ALREADY_STARTED
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_ISOCH_BANDWIDTH_EXCEEDED
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_ISOCH_FAILED
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_ISOCH_NOT_STARTED
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_ISOCH_RETRIEVE_BUFFER_FAILED
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_ISOCH_START_FAILED
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_ISOCH_STOP_FAILED
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_ISOCH_SYNC_FAILED
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_LOW_LEVEL_FAILURE
 - FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_LUT_FAILED
 - FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_MEMORY_ALLOCATION_FAILED
 - FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_NOT_CONNECTED
 - FlyCapture2Defs_C.h, [98](#)

- FC2_ERROR_NOT_FOUND
FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_NOT_IMPLEMENTED
FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_NOT_INITIALIZED
FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_NOT_IN_FORMAT7
FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_NOT_SUPPORTED
FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_OK
FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_PROPERTY_FAILED
FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_PROPERTY_NOT_PRESENT
FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_READ_REGISTER_FAILED
FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_REGISTER_FAILED
FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_STROBE_FAILED
FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_TIMEOUT
FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_TRIGGER_FAILED
FlyCapture2Defs_C.h, [99](#)
- FC2_ERROR_UNDEFINED
FlyCapture2Defs_C.h, [98](#)
- FC2_ERROR_WRITE_REGISTER_FAILED
FlyCapture2Defs_C.h, [99](#)
- FC2_FOCUS
FlyCapture2Defs_C.h, [104](#)
- FC2_FRAMERATE_120
FlyCapture2Defs_C.h, [100](#)
- FC2_FRAMERATE_15
FlyCapture2Defs_C.h, [100](#)
- FC2_FRAMERATE_1_875
FlyCapture2Defs_C.h, [100](#)
- FC2_FRAMERATE_240
FlyCapture2Defs_C.h, [100](#)
- FC2_FRAMERATE_30
FlyCapture2Defs_C.h, [100](#)
- FC2_FRAMERATE_3_75
FlyCapture2Defs_C.h, [100](#)
- FC2_FRAMERATE_60
FlyCapture2Defs_C.h, [100](#)
- FC2_FRAMERATE_7_5
FlyCapture2Defs_C.h, [100](#)
- FC2_FRAMERATE_FORCE_32BITS
FlyCapture2Defs_C.h, [100](#)
- FC2_FRAMERATE_FORMAT7
FlyCapture2Defs_C.h, [100](#)
- FC2_FRAME_RATE
FlyCapture2Defs_C.h, [104](#)
- FC2_FROM_FILE_EXT
FlyCapture2Defs_C.h, [101](#)
- FC2_GAIN
FlyCapture2Defs_C.h, [104](#)
- FC2_GAMMA
FlyCapture2Defs_C.h, [104](#)
- FC2_GRAB_MODE_FORCE_32BITS
FlyCapture2Defs_C.h, [100](#)
- FC2_GRAB_TIMEOUT_FORCE_32BITS
FlyCapture2Defs_C.h, [100](#)
- FC2_HEARTBEAT
FlyCapture2Defs_C.h, [100](#)
- FC2_HEARTBEAT_TIMEOUT
FlyCapture2Defs_C.h, [100](#)
- FC2_HQ_LINEAR
FlyCapture2Defs_C.h, [97](#)
- FC2_HUE
FlyCapture2Defs_C.h, [104](#)
- FC2_IMAGE_FILE_FORMAT_FORCE_32BITS
FlyCapture2Defs_C.h, [101](#)
- FC2_INTERFACE_GIGE
FlyCapture2Defs_C.h, [101](#)
- FC2_INTERFACE_IEEE1394
FlyCapture2Defs_C.h, [101](#)
- FC2_INTERFACE_TYPE_FORCE_32BITS
FlyCapture2Defs_C.h, [101](#)
- FC2_INTERFACE_UNKNOWN
FlyCapture2Defs_C.h, [101](#)
- FC2_INTERFACE_USB_2
FlyCapture2Defs_C.h, [101](#)
- FC2_INTERFACE_USB_3
FlyCapture2Defs_C.h, [101](#)
- FC2_IPP
FlyCapture2Defs_C.h, [97](#)
- FC2_IRIS
FlyCapture2Defs_C.h, [104](#)
- FC2_JPEG
FlyCapture2Defs_C.h, [101](#)
- FC2_JPEG2000
FlyCapture2Defs_C.h, [101](#)
- FC2_LINUX_X64

- FlyCapture2Defs_C.h, [102](#)
- FC2_LINUX_X86
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MAC
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_0
 - FlyCapture2Defs_C.h, [101](#)
- FC2_MODE_1
 - FlyCapture2Defs_C.h, [101](#)
- FC2_MODE_10
 - FlyCapture2Defs_C.h, [101](#)
- FC2_MODE_11
 - FlyCapture2Defs_C.h, [101](#)
- FC2_MODE_12
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_13
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_14
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_15
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_16
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_17
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_18
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_19
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_2
 - FlyCapture2Defs_C.h, [101](#)
- FC2_MODE_20
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_21
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_22
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_23
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_24
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_25
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_26
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_27
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_28
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_29
 - FlyCapture2Defs_C.h, [102](#)
- FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_3
 - FlyCapture2Defs_C.h, [101](#)
- FC2_MODE_30
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_31
 - FlyCapture2Defs_C.h, [102](#)
- FC2_MODE_4
 - FlyCapture2Defs_C.h, [101](#)
- FC2_MODE_5
 - FlyCapture2Defs_C.h, [101](#)
- FC2_MODE_6
 - FlyCapture2Defs_C.h, [101](#)
- FC2_MODE_7
 - FlyCapture2Defs_C.h, [101](#)
- FC2_MODE_8
 - FlyCapture2Defs_C.h, [101](#)
- FC2_MODE_9
 - FlyCapture2Defs_C.h, [101](#)
- FC2_MODE_FORCE_32BITS
 - FlyCapture2Defs_C.h, [102](#)
- FC2_NEAREST_NEIGHBOR_FAST
 - FlyCapture2Defs_C.h, [97](#)
- FC2_NO_COLOR_PROCESSING
 - FlyCapture2Defs_C.h, [97](#)
- FC2_NUM_FRAMERATES
 - FlyCapture2Defs_C.h, [100](#)
- FC2_NUM_MODES
 - FlyCapture2Defs_C.h, [102](#)
- FC2_NUM_PIXEL_FORMATS
 - FlyCapture2Defs_C.h, [103](#)
- FC2_NUM_VIDEOMODES
 - FlyCapture2Defs_C.h, [105](#)
- FC2_OSTYPE_FORCE_32BITS
 - FlyCapture2Defs_C.h, [102](#)
- FC2_PAN
 - FlyCapture2Defs_C.h, [104](#)
- FC2_PCIE_BUSSPEED_2_5
 - FlyCapture2Defs_C.h, [103](#)
- FC2_PCIE_BUSSPEED_5_0
 - FlyCapture2Defs_C.h, [103](#)
- FC2_PCIE_BUSSPEED_FORCE_32BITS
 - FlyCapture2Defs_C.h, [103](#)
- FC2_PGM
 - FlyCapture2Defs_C.h, [101](#)
- FC2_PIXEL_FORMAT_411YUV8
 - FlyCapture2Defs_C.h, [103](#)

- FC2_PIXEL_FORMAT_422YUV8
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_422YUV8_JPEG
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_444YUV8
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_BGR
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_BGR16
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_BGRU
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_BGRU16
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_MONO12
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_MONO16
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_MONO8
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_RAW12
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_RAW16
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_RAW8
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_RGB
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_RGB16
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_RGB8
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_RGBU
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_S_MONO16
FlyCapture2Defs_C.h, [103](#)
- FC2_PIXEL_FORMAT_S_RGB16
FlyCapture2Defs_C.h, [103](#)
- FC2_PNG
FlyCapture2Defs_C.h, [101](#)
- FC2_PPM
FlyCapture2Defs_C.h, [101](#)
- FC2_PROPERTY_TYPE_FORCE_32BITS
FlyCapture2Defs_C.h, [104](#)
- FC2_RAW
FlyCapture2Defs_C.h, [101](#)
- FC2_REMOVAL
FlyCapture2Defs_C.h, [96](#)
- FC2_RIGOROUS
FlyCapture2Defs_C.h, [97](#)
- FC2_SATURATION
FlyCapture2Defs_C.h, [104](#)
- FC2_SHARPNESS
FlyCapture2Defs_C.h, [104](#)
- FC2_SHUTTER
FlyCapture2Defs_C.h, [104](#)
- FC2_STATISTICS_BLUE
FlyCapture2Defs_C.h, [104](#)
- FC2_STATISTICS_FORCE_32BITS
FlyCapture2Defs_C.h, [104](#)
- FC2_STATISTICS_GREEN
FlyCapture2Defs_C.h, [104](#)
- FC2_STATISTICS_GREY
FlyCapture2Defs_C.h, [104](#)
- FC2_STATISTICS_HUE
FlyCapture2Defs_C.h, [104](#)
- FC2_STATISTICS_LIGHTNESS
FlyCapture2Defs_C.h, [104](#)
- FC2_STATISTICS_RED
FlyCapture2Defs_C.h, [104](#)
- FC2_STATISTICS_SATURATION
FlyCapture2Defs_C.h, [104](#)
- FC2_TEMPERATURE
FlyCapture2Defs_C.h, [104](#)
- FC2_TIFF
FlyCapture2Defs_C.h, [101](#)
- FC2_TIFF_ADOBE_DEFLATE
FlyCapture2Defs_C.h, [105](#)
- FC2_TIFF_CCITTFAX3
FlyCapture2Defs_C.h, [105](#)
- FC2_TIFF_CCITTFAX4
FlyCapture2Defs_C.h, [105](#)
- FC2_TIFF_DEFLATE
FlyCapture2Defs_C.h, [105](#)
- FC2_TIFF_JPEG
FlyCapture2Defs_C.h, [105](#)
- FC2_TIFF_LZW
FlyCapture2Defs_C.h, [105](#)
- FC2_TIFF_NONE
FlyCapture2Defs_C.h, [105](#)
- FC2_TIFF_PACKBITS
FlyCapture2Defs_C.h, [105](#)
- FC2_TILT
FlyCapture2Defs_C.h, [104](#)
- FC2_TIMEOUT_INFINITE
FlyCapture2Defs_C.h, [100](#)
- FC2_TIMEOUT_NONE
FlyCapture2Defs_C.h, [100](#)
- FC2_TIMEOUT_UNSPECIFIED

- FlyCapture2Defs_C.h, [100](#)
- FC2_TRIGGER_DELAY
 - FlyCapture2Defs_C.h, [104](#)
- FC2_TRIGGER_MODE
 - FlyCapture2Defs_C.h, [104](#)
- FC2_UNKNOWN_OS
 - FlyCapture2Defs_C.h, [102](#)
- FC2_UNSPECIFIED_GRAB_MODE
 - FlyCapture2Defs_C.h, [100](#)
- FC2_UNSPECIFIED_PIXEL_FORMAT
 - FlyCapture2Defs_C.h, [103](#)
- FC2_UNSPECIFIED_PROPERTY_TYPE
 - FlyCapture2Defs_C.h, [104](#)
- FC2_VIDEOMODE_1024x768RGB
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_1024x768Y16
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_1024x768Y8
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_1024x768YUV422
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_1280x960RGB
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_1280x960Y16
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_1280x960Y8
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_1280x960YUV422
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_1600x1200RGB
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_1600x1200Y16
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_1600x1200Y8
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_1600x1200YUV422
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_160x120YUV444
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_320x240YUV422
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_640x480RGB
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_640x480Y16
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_640x480Y8
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_640x480YUV411
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_640x480YUV422
 - FlyCapture2Defs_C.h, [105](#)
- FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_800x600RGB
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_800x600Y16
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_800x600Y8
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_800x600YUV422
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_FORCE_32BITS
 - FlyCapture2Defs_C.h, [105](#)
- FC2_VIDEOMODE_FORMAT7
 - FlyCapture2Defs_C.h, [105](#)
- FC2_WHITE_BALANCE
 - FlyCapture2Defs_C.h, [104](#)
- FC2_WINDOWS_X64
 - FlyCapture2Defs_C.h, [102](#)
- FC2_WINDOWS_X86
 - FlyCapture2Defs_C.h, [102](#)
- FC2_ZOOM
 - FlyCapture2Defs_C.h, [104](#)
- FlyCapture2Defs_C.h
 - FC2_ARRIVAL, [96](#)
 - FC2_AUTO_EXPOSURE, [104](#)
 - FC2_BANDWIDTH_ALLOCATION_
FORCE_32BITS, [96](#)
 - FC2_BANDWIDTH_ALLOCATION_
OFF, [96](#)
 - FC2_BANDWIDTH_ALLOCATION_
ON, [96](#)
 - FC2_BANDWIDTH_ALLOCATION_
UNSPECIFIED, [96](#)
 - FC2_BANDWIDTH_ALLOCATION_
UNSUPPORTED, [96](#)
 - FC2_BMP, [101](#)
 - FC2_BRIGHTNESS, [104](#)
 - FC2_BT_BGGR, [96](#)
 - FC2_BT_FORCE_32BITS, [96](#)
 - FC2_BT_GBRG, [96](#)
 - FC2_BT_GRBG, [96](#)
 - FC2_BT_NONE, [96](#)
 - FC2_BT_RRGB, [96](#)
 - FC2_BUFFER_FRAMES, [100](#)
 - FC2_BUSSPEED_10000BASE_T,
[97](#)
 - FC2_BUSSPEED_1000BASE_T, [97](#)
 - FC2_BUSSPEED_100BASE_T, [97](#)
 - FC2_BUSSPEED_10BASE_T, [97](#)
 - FC2_BUSSPEED_ANY, [97](#)

- FC2_BUSSPEED_FORCE_32BITS, [97](#)
- FC2_BUSSPEED_S100, [97](#)
- FC2_BUSSPEED_S1600, [97](#)
- FC2_BUSSPEED_S200, [97](#)
- FC2_BUSSPEED_S3200, [97](#)
- FC2_BUSSPEED_S400, [97](#)
- FC2_BUSSPEED_S480, [97](#)
- FC2_BUSSPEED_S5000, [97](#)
- FC2_BUSSPEED_S800, [97](#)
- FC2_BUSSPEED_SPEED_UNKNOW-
N, [97](#)
- FC2_BUSSPEED_S_FASTEST, [97](#)
- FC2_BUS_RESET, [96](#)
- FC2_BYTE_ORDER_BIG_ENDIAN, [97](#)
- FC2_BYTE_ORDER_FORCE_32BI-
TS, [97](#)
- FC2_BYTE_ORDER_LITTLE_ENDI-
AN, [97](#)
- FC2_CALLBACK_TYPE_FORCE_-
32BITS, [96](#)
- FC2_COLOR_PROCESSING_AL-
GORITHM_FORCE_32BITS, [97](#)
- FC2_DEFAULT, [97](#)
- FC2_DIRECTIONAL, [97](#)
- FC2_DRIVER_1394_CAM, [98](#)
- FC2_DRIVER_1394_JUJU, [98](#)
- FC2_DRIVER_1394_PRO, [98](#)
- FC2_DRIVER_1394_RAW1394, [98](#)
- FC2_DRIVER_1394_VIDEO1394, [98](#)
- FC2_DRIVER_FORCE_32BITS, [98](#)
- FC2_DRIVER_GIGE_FILTER, [98](#)
- FC2_DRIVER_GIGE_NONE, [98](#)
- FC2_DRIVER_GIGE_PRO, [98](#)
- FC2_DRIVER_UNKNOWN, [98](#)
- FC2_DRIVER_USB3_PRO, [98](#)
- FC2_DRIVER_USB_CAM, [98](#)
- FC2_DRIVER_USB_NONE, [98](#)
- FC2_DROP_FRAMES, [100](#)
- FC2_EDGE_SENSING, [97](#)
- FC2_ERROR_BUFFER_TOO_SM-
ALL, [99](#)
- FC2_ERROR_BUS_MASTER_FAI-
LED, [99](#)
- FC2_ERROR_FAILED, [98](#)
- FC2_ERROR_FAILED_BUS_MAS-
TER_CONNECTION, [98](#)
- FC2_ERROR_FAILED_GUID, [98](#)
- FC2_ERROR_FORCE_32BITS, [99](#)
- FC2_ERROR_IIDC_FAILED, [99](#)
- FC2_ERROR_IMAGE_CONSISTE-
NCY_ERROR, [99](#)
- FC2_ERROR_IMAGE_CONVERSI-
ON_FAILED, [99](#)
- FC2_ERROR_IMAGE_LIBRARY_F-
AILURE, [99](#)
- FC2_ERROR_INIT_FAILED, [98](#)
- FC2_ERROR_INVALID_BUS_MAN-
AGER, [98](#)
- FC2_ERROR_INVALID_GENERAT-
ION, [99](#)
- FC2_ERROR_INVALID_MODE, [98](#)
- FC2_ERROR_INVALID_PACKET_-
SIZE, [98](#)
- FC2_ERROR_INVALID_PARAMET-
ER, [98](#)
- FC2_ERROR_INVALID_SETTINGS, [98](#)
- FC2_ERROR_ISOCH_ALREADY_-
STARTED, [99](#)
- FC2_ERROR_ISOCH_BANDWIDT-
H_EXCEEDED, [99](#)
- FC2_ERROR_ISOCH_FAILED, [99](#)
- FC2_ERROR_ISOCH_NOT_STAR-
TED, [99](#)
- FC2_ERROR_ISOCH_RETRIEVE_-
BUFFER_FAILED, [99](#)
- FC2_ERROR_ISOCH_START_FAI-
LED, [99](#)
- FC2_ERROR_ISOCH_STOP_FAI-
LED, [99](#)
- FC2_ERROR_ISOCH_SYNC_FAI-
LED, [99](#)
- FC2_ERROR_LOW_LEVEL_FAILO-
RE, [98](#)
- FC2_ERROR_LUT_FAILED, [99](#)
- FC2_ERROR_MEMORY_ALLOCA-
TION_FAILED, [98](#)
- FC2_ERROR_NOT_CONNECTED, [98](#)
- FC2_ERROR_NOT_FOUND, [98](#)
- FC2_ERROR_NOT_IMPLEMENTED, [98](#)
- FC2_ERROR_NOT_INITIALIZED, [98](#)
- FC2_ERROR_NOT_IN_FORMAT7, [99](#)

- FC2_ERROR_NOT_SUPPORTED, [99](#)
- FC2_ERROR_OK, [98](#)
- FC2_ERROR_PROPERTY_FAILED, [99](#)
- FC2_ERROR_PROPERTY_NOT_P-
RESENT, [99](#)
- FC2_ERROR_READ_REGISTER_-
FAILED, [99](#)
- FC2_ERROR_REGISTER_FAILED,
[99](#)
- FC2_ERROR_STROBE_FAILED, [99](#)
- FC2_ERROR_TIMEOUT, [99](#)
- FC2_ERROR_TRIGGER_FAILED,
[99](#)
- FC2_ERROR_UNDEFINED, [98](#)
- FC2_ERROR_WRITE_REGISTER_-
_FAILED, [99](#)
- FC2_FOCUS, [104](#)
- FC2_FRAMERATE_120, [100](#)
- FC2_FRAMERATE_15, [100](#)
- FC2_FRAMERATE_1_875, [100](#)
- FC2_FRAMERATE_240, [100](#)
- FC2_FRAMERATE_30, [100](#)
- FC2_FRAMERATE_3_75, [100](#)
- FC2_FRAMERATE_60, [100](#)
- FC2_FRAMERATE_7_5, [100](#)
- FC2_FRAMERATE_FORCE_32BIT-
S, [100](#)
- FC2_FRAMERATE_FORMAT7, [100](#)
- FC2_FRAME_RATE, [104](#)
- FC2_FROM_FILE_EXT, [101](#)
- FC2_GAIN, [104](#)
- FC2_GAMMA, [104](#)
- FC2_GRAB_MODE_FORCE_32BI-
TS, [100](#)
- FC2_GRAB_TIMEOUT_FORCE_-
32BITS, [100](#)
- FC2_HEARTBEAT, [100](#)
- FC2_HEARTBEAT_TIMEOUT, [100](#)
- FC2_HQ_LINEAR, [97](#)
- FC2_HUE, [104](#)
- FC2_IMAGE_FILE_FORMAT_FOR-
CE_32BITS, [101](#)
- FC2_INTERFACE_GIGE, [101](#)
- FC2_INTERFACE_IEEE1394, [101](#)
- FC2_INTERFACE_TYPE_FORCE_-
32BITS, [101](#)
- FC2_INTERFACE_UNKNOWN, [101](#)
- FC2_INTERFACE_USB_2, [101](#)
- FC2_INTERFACE_USB_3, [101](#)
- FC2_IPP, [97](#)
- FC2_IRIS, [104](#)
- FC2_JPEG, [101](#)
- FC2_JPEG2000, [101](#)
- FC2_LINUX_X64, [102](#)
- FC2_LINUX_X86, [102](#)
- FC2_MAC, [102](#)
- FC2_MODE_0, [101](#)
- FC2_MODE_1, [101](#)
- FC2_MODE_10, [101](#)
- FC2_MODE_11, [101](#)
- FC2_MODE_12, [102](#)
- FC2_MODE_13, [102](#)
- FC2_MODE_14, [102](#)
- FC2_MODE_15, [102](#)
- FC2_MODE_16, [102](#)
- FC2_MODE_17, [102](#)
- FC2_MODE_18, [102](#)
- FC2_MODE_19, [102](#)
- FC2_MODE_2, [101](#)
- FC2_MODE_20, [102](#)
- FC2_MODE_21, [102](#)
- FC2_MODE_22, [102](#)
- FC2_MODE_23, [102](#)
- FC2_MODE_24, [102](#)
- FC2_MODE_25, [102](#)
- FC2_MODE_26, [102](#)
- FC2_MODE_27, [102](#)
- FC2_MODE_28, [102](#)
- FC2_MODE_29, [102](#)
- FC2_MODE_3, [101](#)
- FC2_MODE_30, [102](#)
- FC2_MODE_31, [102](#)
- FC2_MODE_4, [101](#)
- FC2_MODE_5, [101](#)
- FC2_MODE_6, [101](#)
- FC2_MODE_7, [101](#)
- FC2_MODE_8, [101](#)
- FC2_MODE_9, [101](#)
- FC2_MODE_FORCE_32BITS, [102](#)
- FC2_NEAREST_NEIGHBOR_FAS-
T, [97](#)
- FC2_NO_COLOR_PROCESSING,
[97](#)
- FC2_NUM_FRAMERATES, [100](#)
- FC2_NUM_MODES, [102](#)
- FC2_NUM_PIXEL_FORMATS, [103](#)
- FC2_NUM_VIDEOMODES, [105](#)

- FC2_OSTYPE_FORCE_32BITS, [102](#)
- FC2_PAN, [104](#)
- FC2_PCIE_BUSSPEED_2_5, [103](#)
- FC2_PCIE_BUSSPEED_5_0, [103](#)
- FC2_PCIE_BUSSPEED_FORCE_32BITS, [103](#)
- FC2_PCIE_BUSSPEED_UNKNOWN, [103](#)
- FC2_PGM, [101](#)
- FC2_PIXEL_FORMAT_411YUV8, [103](#)
- FC2_PIXEL_FORMAT_422YUV8, [103](#)
- FC2_PIXEL_FORMAT_422YUV8_JPEG, [103](#)
- FC2_PIXEL_FORMAT_444YUV8, [103](#)
- FC2_PIXEL_FORMAT_BGR, [103](#)
- FC2_PIXEL_FORMAT_BGR16, [103](#)
- FC2_PIXEL_FORMAT_BGRU, [103](#)
- FC2_PIXEL_FORMAT_BGRU16, [103](#)
- FC2_PIXEL_FORMAT_MONO12, [103](#)
- FC2_PIXEL_FORMAT_MONO16, [103](#)
- FC2_PIXEL_FORMAT_MONO8, [103](#)
- FC2_PIXEL_FORMAT_RAW12, [103](#)
- FC2_PIXEL_FORMAT_RAW16, [103](#)
- FC2_PIXEL_FORMAT_RAW8, [103](#)
- FC2_PIXEL_FORMAT_RGB, [103](#)
- FC2_PIXEL_FORMAT_RGB16, [103](#)
- FC2_PIXEL_FORMAT_RGB8, [103](#)
- FC2_PIXEL_FORMAT_RGBU, [103](#)
- FC2_PIXEL_FORMAT_S_MONO16, [103](#)
- FC2_PIXEL_FORMAT_S_RGB16, [103](#)
- FC2_PNG, [101](#)
- FC2_PPM, [101](#)
- FC2_PROPERTY_TYPE_FORCE_32BITS, [104](#)
- FC2_RAW, [101](#)
- FC2_REMOVAL, [96](#)
- FC2_RIGOROUS, [97](#)
- FC2_SATURATION, [104](#)
- FC2_SHARPNESS, [104](#)
- FC2_SHUTTER, [104](#)
- FC2_STATISTICS_BLUE, [104](#)
- FC2_STATISTICS_FORCE_32BITS, [104](#)
- FC2_STATISTICS_GREEN, [104](#)
- FC2_STATISTICS_GREY, [104](#)
- FC2_STATISTICS_HUE, [104](#)
- FC2_STATISTICS_LIGHTNESS, [104](#)
- FC2_STATISTICS_RED, [104](#)
- FC2_STATISTICS_SATURATION, [104](#)
- FC2_TEMPERATURE, [104](#)
- FC2_TIFF, [101](#)
- FC2_TIFF_ADOBE_DEFLATE, [105](#)
- FC2_TIFF_CCITTFAX3, [105](#)
- FC2_TIFF_CCITTFAX4, [105](#)
- FC2_TIFF_DEFLATE, [105](#)
- FC2_TIFF_JPEG, [105](#)
- FC2_TIFF_LZW, [105](#)
- FC2_TIFF_NONE, [105](#)
- FC2_TIFF_PACKBITS, [105](#)
- FC2_TILT, [104](#)
- FC2_TIMEOUT_INFINITE, [100](#)
- FC2_TIMEOUT_NONE, [100](#)
- FC2_TIMEOUT_UNSPECIFIED, [100](#)
- FC2_TRIGGER_DELAY, [104](#)
- FC2_TRIGGER_MODE, [104](#)
- FC2_UNKNOWN_OS, [102](#)
- FC2_UNSPECIFIED_GRAB_MODE, [100](#)
- FC2_UNSPECIFIED_PIXEL_FORMAT, [103](#)
- FC2_UNSPECIFIED_PROPERTY_TYPE, [104](#)
- FC2_VIDEOMODE_1024x768RGB, [105](#)
- FC2_VIDEOMODE_1024x768Y16, [105](#)
- FC2_VIDEOMODE_1024x768Y8, [105](#)
- FC2_VIDEOMODE_1024x768YUV422, [105](#)
- FC2_VIDEOMODE_1280x960RGB, [105](#)
- FC2_VIDEOMODE_1280x960Y16, [105](#)
- FC2_VIDEOMODE_1280x960Y8, [105](#)
- FC2_VIDEOMODE_1280x960YUV422, [105](#)

- FC2_VIDEOMODE_1600x1200RGB, [105](#)
- FC2_VIDEOMODE_1600x1200Y16, [105](#)
- FC2_VIDEOMODE_1600x1200Y8, [105](#)
- FC2_VIDEOMODE_1600x1200YU-V422, [105](#)
- FC2_VIDEOMODE_160x120YU-V444, [105](#)
- FC2_VIDEOMODE_320x240YU-V422, [105](#)
- FC2_VIDEOMODE_640x480RGB, [105](#)
- FC2_VIDEOMODE_640x480Y16, [105](#)
- FC2_VIDEOMODE_640x480Y8, [105](#)
- FC2_VIDEOMODE_640x480YU-V411, [105](#)
- FC2_VIDEOMODE_640x480YU-V422, [105](#)
- FC2_VIDEOMODE_800x600RGB, [105](#)
- FC2_VIDEOMODE_800x600Y16, [105](#)
- FC2_VIDEOMODE_800x600Y8, [105](#)
- FC2_VIDEOMODE_800x600YU-V422, [105](#)
- FC2_VIDEOMODE_FORCE_32BITS, [105](#)
- FC2_VIDEOMODE_FORMAT7, [105](#)
- FC2_WHITE_BALANCE, [104](#)
- FC2_WINDOWS_X64, [102](#)
- FC2_WINDOWS_X86, [102](#)
- FC2_ZOOM, [104](#)
- MultiSyncLibraryDefs_C.h
 - SYNC_ERROR_ALREADY_STARTED, [115](#)
 - SYNC_ERROR_ALREADY_STOPPED, [115](#)
 - SYNC_ERROR_CONTEXT_NOT_INITIALIZED, [115](#)
 - SYNC_ERROR_FAILED, [115](#)
 - SYNC_ERROR_OK, [115](#)
 - SYNC_ERROR_UNKNOWN_ERROR, [115](#)
 - SYNC_MESSAGE_BUS_RESET, [115](#)
 - SYNC_MESSAGE_DEVICE_ERROR, [115](#)
 - SYNC_MESSAGE_FAILED, [115](#)
 - SYNC_MESSAGE_NOMASTER, [115](#)
 - SYNC_MESSAGE_NOT_ENOUGH_DEVICES, [115](#)
 - SYNC_MESSAGE_NOT_INITIALIZED, [115](#)
 - SYNC_MESSAGE_OK, [115](#)
 - SYNC_MESSAGE_STARTED, [115](#)
 - SYNC_MESSAGE_STOPPED, [115](#)
 - SYNC_MESSAGE_SYNCING, [115](#)
 - SYNC_MESSAGE_THREAD_ERROR, [115](#)
 - SYNC_MESSAGE_UNKNOWN_ERROR, [115](#)
 - SYNC_ERROR_ALREADY_STARTED_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_ERROR_ALREADY_STOPPED_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_ERROR_CONTEXT_NOT_INITIALIZED_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_ERROR_FAILED_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_ERROR_OK_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_ERROR_UNKNOWN_ERROR_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_MESSAGE_BUS_RESET_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_MESSAGE_DEVICE_ERROR_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_MESSAGE_FAILED_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_MESSAGE_NOMASTER_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_MESSAGE_NOT_ENOUGH_DEVICES_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_MESSAGE_NOT_INITIALIZED_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_MESSAGE_OK_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_MESSAGE_STARTED_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_MESSAGE_STOPPED_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_MESSAGE_SYNCING_MultiSyncLibraryDefs_C.h, [115](#)
 - SYNC_MESSAGE_THREAD_ERROR_MultiSyncLibraryDefs_C.h, [115](#)

- MultiSyncLibraryDefs_C.h, [115](#)
- SYNC_MESSAGE_UNKNOWN_ERROR
 - MultiSyncLibraryDefs_C.h, [115](#)
- BOOL
 - FlyCapture2Defs_C.h, [95](#)
 - MultiSyncLibraryDefs_C.h, [114](#)
- FALSE
 - FlyCapture2Defs_C.h, [95](#)
 - MultiSyncLibraryDefs_C.h, [114](#)
- FULL_32BIT_VALUE
 - FlyCapture2Defs_C.h, [95](#)
 - MultiSyncLibraryDefs_C.h, [114](#)
- FlyCapture2Defs_C.h, [90](#)
 - BOOL, [95](#)
 - FALSE, [95](#)
 - TRUE, [95](#)
 - fc2AVIContext, [95](#)
 - fc2AsyncCommandCallback, [95](#)
 - fc2BandwidthAllocation, [96](#)
 - fc2BayerTileFormat, [96](#)
 - fc2BusCallbackType, [96](#)
 - fc2BusEventCallback, [95](#)
 - fc2BusSpeed, [96](#)
 - fc2ByteOrder, [97](#)
 - fc2CallbackHandle, [95](#)
 - fc2ColorProcessingAlgorithm, [97](#)
 - fc2Context, [95](#)
 - fc2DriverType, [97](#)
 - fc2Error, [98](#)
 - fc2FrameRate, [99](#)
 - fc2GigEPropertyType, [100](#)
 - fc2GrabMode, [100](#)
 - fc2GrabTimeout, [100](#)
 - fc2GuiContext, [95](#)
 - fc2ImageEventCallback, [95](#)
 - fc2ImageFileFormat, [100](#)
 - fc2ImageImpl, [96](#)
 - fc2ImageStatisticsContext, [96](#)
 - fc2InterfaceType, [101](#)
 - fc2Mode, [101](#)
 - fc2OSType, [102](#)
 - fc2PCleBusSpeed, [102](#)
 - fc2PixelFormat, [103](#)
 - fc2PropertyType, [103](#)
 - fc2StatisticsChannel, [104](#)
 - fc2TIFFCompressionMethod, [104](#)
 - fc2VideoMode, [105](#)
- FlyCapture2GUI_C.h, [106](#)
 - fc2CreateGuiContext, [106](#)
 - fc2DestroyGuiContext, [106](#)
 - fc2Disconnect, [107](#)
 - fc2GUIConnect, [107](#)
 - fc2Hide, [107](#)
 - fc2IsVisible, [107](#)
 - fc2Show, [108](#)
 - fc2ShowModal, [108](#)
- FlyCapture2Internal_C.h, [108](#)
 - IsContextValid, [109](#)
 - IsGuiContextValid, [109](#)
 - SyncCpplImageToStruct, [109](#)
- FlyCapture2Platform_C.h, [109](#)
- FlyCapture2_C.h, [37](#)
 - fc2AVIAppend, [46](#)
 - fc2AVIClose, [46](#)
 - fc2AVIOpen, [47](#)
 - fc2CalculateImageStatistics, [47](#)
 - fc2Connect, [47](#)
 - fc2ConvertImage, [48](#)
 - fc2ConvertImageTo, [48](#)
 - fc2CreateAVI, [48](#)
 - fc2CreateContext, [49](#)
 - fc2CreateGigEContext, [49](#)
 - fc2CreateImage, [49](#)
 - fc2CreateImageStatistics, [50](#)
 - fc2DestroyAVI, [50](#)
 - fc2DestroyContext, [50](#)
 - fc2DestroyImage, [50](#)
 - fc2DestroyImageStatistics, [51](#)
 - fc2DetermineBitsPerPixel, [51](#)
 - fc2Disconnect, [51](#)
 - fc2DiscoverGigECameras, [52](#)
 - fc2EnableLUT, [52](#)
 - fc2ErrorToDescription, [52](#)
 - fc2FireBusReset, [53](#)
 - fc2FireSoftwareTrigger, [53](#)
 - fc2FireSoftwareTriggerBroadcast, [53](#)
 - fc2ForceAllIPAddressesAutomatically, [54](#)
 - fc2ForceIPAddressAutomatically, [54](#)
 - fc2ForceIPAddressToCamera, [54](#)
 - fc2GetActiveLUTBank, [55](#)
 - fc2GetCameraFromIndex, [55](#)
 - fc2GetCameraFromSerialNumber, [55](#)
 - fc2GetCameraInfo, [56](#)
 - fc2GetCameraSerialNumberFromIndex, [56](#)
 - fc2GetChannelStatus, [56](#)
 - fc2GetConfiguration, [57](#)
 - fc2GetCycleTime, [57](#)

- fc2GetDefaultColorProcessing, 58
- fc2GetDefaultOutputFormat, 58
- fc2GetDeviceFromIndex, 58
- fc2GetEmbeddedImageInfo, 59
- fc2GetFormat7Configuration, 59
- fc2GetFormat7Info, 60
- fc2GetGPIOPinDirection, 61
- fc2GetGigEConfig, 60
- fc2GetGigEImageBinningSettings, 60
- fc2GetGigEImageSettings, 60
- fc2GetGigEImageSettingsInfo, 60
- fc2GetGigEImagingMode, 60
- fc2GetGigEProperty, 60
- fc2GetGigEStreamChannelInfo, 61
- fc2GetImageData, 61
- fc2GetImageStatistics, 62
- fc2GetImageTimeStamp, 62
- fc2GetInterfaceTypeFromGuid, 62
- fc2GetLUTBankInfo, 63
- fc2GetLUTChannel, 64
- fc2GetLUTInfo, 64
- fc2GetLibraryVersion, 63
- fc2GetMemoryChannel, 64
- fc2GetMemoryChannelInfo, 65
- fc2GetNumOfCameras, 65
- fc2GetNumOfDevices, 65
- fc2GetNumStreamChannels, 66
- fc2GetProperty, 66
- fc2GetPropertyInfo, 66
- fc2GetRegisterString, 66
- fc2GetStrobe, 67
- fc2GetStrobeInfo, 67
- fc2GetSystemInfo, 67
- fc2GetTriggerDelay, 68
- fc2GetTriggerDelayInfo, 68
- fc2GetTriggerMode, 68
- fc2GetTriggerModeInfo, 68
- fc2GetVideoModeAndFrameRate, 69
- fc2GetVideoModeAndFrameRateInfo, 69
- fc2H264Open, 70
- fc2IsCameraControlable, 70
- fc2LaunchBrowser, 70
- fc2LaunchCommand, 71
- fc2LaunchCommandAsync, 71
- fc2LaunchHelp, 71
- fc2MJPGOpen, 71
- fc2QueryGigEImagingMode, 72
- fc2ReadGVCPMemory, 72
- fc2ReadGVCPRegister, 72
- fc2ReadGVCPRegisterBlock, 73
- fc2ReadRegister, 73
- fc2ReadRegisterBlock, 73
- fc2RegisterCallback, 74
- fc2RescanBus, 74
- fc2RestoreFromMemoryChannel, 74
- fc2RetrieveBuffer, 75
- fc2SaveImage, 75
- fc2SaveImageWithOption, 75
- fc2SaveToMemoryChannel, 76
- fc2SetActiveLUTBank, 76
- fc2SetCallback, 76
- fc2SetChannelStatus, 77
- fc2SetConfiguration, 77
- fc2SetDefaultColorProcessing, 77
- fc2SetDefaultOutputFormat, 78
- fc2SetEmbeddedImageInfo, 78
- fc2SetFormat7Configuration, 79
- fc2SetFormat7ConfigurationPacket, 79
- fc2SetGPIOPinDirection, 80
- fc2SetGPIOPinDirectionBroadcast, 80
- fc2SetGigEConfig, 79
- fc2SetGigEImageBinningSettings, 79
- fc2SetGigEImageSettings, 79
- fc2SetGigEImagingMode, 79
- fc2SetGigEProperty, 80
- fc2SetGigEStreamChannelInfo, 80
- fc2SetImageData, 81
- fc2SetImageDimensions, 81
- fc2SetLUTChannel, 81
- fc2SetProperty, 82
- fc2SetPropertyBroadcast, 82
- fc2SetStrobe, 83
- fc2SetStrobeBroadcast, 83
- fc2SetTriggerDelay, 83
- fc2SetTriggerDelayBroadcast, 84
- fc2SetTriggerMode, 84
- fc2SetTriggerModeBroadcast, 84
- fc2SetUserBuffers, 84
- fc2SetVideoModeAndFrameRate, 85
- fc2StartCapture, 85
- fc2StartCaptureCallback, 86
- fc2StartSyncCapture, 86
- fc2StartSyncCaptureCallback, 86
- fc2StopCapture, 87
- fc2UnregisterCallback, 87

- fc2ValidateFormat7Settings, 87
- fc2WriteGVCPMemory, 88
- fc2WriteGVCPRegister, 88
- fc2WriteGVCPRegisterBlock, 89
- fc2WriteGVCPRegisterBroadcast, 89
- fc2WriteRegister, 89
- fc2WriteRegisterBlock, 90
- fc2WriteRegisterBroadcast, 90
- GPIOPinState
 - fc2EmbeddedImageInfo, 12
- IsContextValid
 - FlyCapture2Internal_C.h, 109
- IsGuiContextValid
 - FlyCapture2Internal_C.h, 109
- MultiSyncLibraryDefs_C.h, 113
 - BOOL, 114
 - FALSE, 114
 - TRUE, 114
 - syncContext, 114
 - syncError, 115
 - syncMessage, 115
- MultiSyncLibraryPlatform_C.h, 115
- MultiSyncLibrary_C.h, 109
 - syncCreateContext, 110
 - syncDestroyContext, 110
 - syncDisableCrossPCsSynchronization, 111
 - syncEnableCrossPCsSynchronization, 111
 - syncGetStatus, 111
 - syncGetTimeSinceSynced, 112
 - syncIsTimingBusConnected, 112
 - syncQueryCrossPCsSynchronizationSetting, 112
 - syncRescanMasterTimingBus, 113
 - syncStart, 113
 - syncStop, 113
- ROIPosition
 - fc2EmbeddedImageInfo, 12
- SyncCpplImageToStruct
 - FlyCapture2Internal_C.h, 109
- TRUE
 - FlyCapture2Defs_C.h, 95
 - MultiSyncLibraryDefs_C.h, 114
- absControl
 - fc2TriggerDelay, 32
- absMax
 - fc2TriggerDelayInfo, 33
- absMin
 - fc2TriggerDelayInfo, 33
- absValSupported
 - fc2TriggerDelayInfo, 33
- absValue
 - fc2TriggerDelay, 32
- applicationIPAddress
 - fc2CameraInfo, 7
- applicationPort
 - fc2CameraInfo, 7
- asyncBusSpeed
 - fc2Config, 9
- autoManualMode
 - fc2TriggerDelay, 32
- autoSupported
 - fc2TriggerDelayInfo, 33
- available
 - fc2EmbeddedImageInfoProperty, 12
- bandwidthAllocation
 - fc2Config, 9
- bayerFormat
 - fc2Image, 20
- bayerTileFormat
 - fc2CameraInfo, 7
- binaryFile
 - fc2PGMOption, 26
 - fc2PPMOption, 28
- bitrate
 - fc2H264Option, 19
- brightness
 - fc2EmbeddedImageInfo, 11
- build
 - fc2Version, 35
- busNumber
 - fc2CameraInfo, 7
- byteOrder
 - fc2SystemInfo, 30
- ccpStatus
 - fc2CameraInfo, 7
- chipIdHi
 - fc2ConfigROM, 10
- chipIdLo
 - fc2ConfigROM, 10
- cols
 - fc2Image, 20
- compression
 - fc2TIFFOption, 31
- compressionLevel
 - fc2PNGOption, 27

- configROM
 - fc2CameraInfo, [7](#)
- cpuDescription
 - fc2SystemInfo, [30](#)
- cycleCount
 - fc2TimeStamp, [31](#)
- cycleOffset
 - fc2TimeStamp, [31](#)
- cycleSeconds
 - fc2TimeStamp, [31](#)
- dataSize
 - fc2Image, [20](#)
- defaultGateway
 - fc2CameraInfo, [7](#)
- delay
 - fc2StrobeControl, [28](#)
- destinationIpAddress
 - fc2GigEStreamChannel, [18](#)
- doNotFragment
 - fc2GigEStreamChannel, [18](#)
- driverList
 - fc2SystemInfo, [30](#)
- driverName
 - fc2CameraInfo, [7](#)
- driverType
 - fc2CameraInfo, [7](#)
- duration
 - fc2StrobeControl, [28](#)
- embeddedBrightness
 - fc2ImageMetadata, [21](#)
- embeddedExposure
 - fc2ImageMetadata, [21](#)
- embeddedFrameCounter
 - fc2ImageMetadata, [21](#)
- embeddedGPIOPinState
 - fc2ImageMetadata, [21](#)
- embeddedGain
 - fc2ImageMetadata, [21](#)
- embeddedROIPosition
 - fc2ImageMetadata, [21](#)
- embeddedShutter
 - fc2ImageMetadata, [21](#)
- embeddedStrobePattern
 - fc2ImageMetadata, [21](#)
- embeddedTimeStamp
 - fc2ImageMetadata, [21](#)
- embeddedWhiteBalance
 - fc2ImageMetadata, [21](#)
- enablePacketResend
 - fc2GigEConfig, [15](#)
- enabled
 - fc2LUTData, [25](#)
- exposure
 - fc2EmbeddedImageInfo, [11](#)
- fc2AVIAppend
 - FlyCapture2_C.h, [46](#)
- fc2AVIClose
 - FlyCapture2_C.h, [46](#)
- fc2AVIContext
 - FlyCapture2Defs_C.h, [95](#)
- fc2AVIOpen
 - FlyCapture2_C.h, [47](#)
- fc2AVIOption, [5](#)
 - frameRate, [5](#)
 - reserved, [5](#)
- fc2AsyncCommandCallback
 - FlyCapture2Defs_C.h, [95](#)
- fc2BandwidthAllocation
 - FlyCapture2Defs_C.h, [96](#)
- fc2BayerTileFormat
 - FlyCapture2Defs_C.h, [96](#)
- fc2BusCallbackType
 - FlyCapture2Defs_C.h, [96](#)
- fc2BusEventCallback
 - FlyCapture2Defs_C.h, [95](#)
- fc2BusSpeed
 - FlyCapture2Defs_C.h, [96](#)
- fc2ByteOrder
 - FlyCapture2Defs_C.h, [97](#)
- fc2CalculateImageStatistics
 - FlyCapture2_C.h, [47](#)
- fc2CallbackHandle
 - FlyCapture2Defs_C.h, [95](#)
- fc2CameraInfo, [6](#)
 - applicationIPAddress, [7](#)
 - applicationPort, [7](#)
 - bayerTileFormat, [7](#)
 - busNumber, [7](#)
 - ccpStatus, [7](#)
 - configROM, [7](#)
 - defaultGateway, [7](#)
 - driverName, [7](#)
 - driverType, [7](#)
 - firmwareBuildTime, [7](#)
 - firmwareVersion, [7](#)
 - gigEMajorVersion, [7](#)
 - gigEMinorVersion, [7](#)

- iidcVer, 8
- interfaceType, 8
- ipAddress, 8
- isColorCamera, 8
- macAddress, 8
- maximumBusSpeed, 8
- modelName, 8
- nodeNumber, 8
- pcieBusSpeed, 8
- reserved, 8
- sensorInfo, 8
- sensorResolution, 8
- serialNumber, 8
- subnetMask, 8
- userDefinedName, 8
- vendorName, 8
- xmlURL1, 8
- xmlURL2, 8
- fc2ColorProcessingAlgorithm
 - FlyCapture2Defs_C.h, 97
- fc2Config, 8
 - asyncBusSpeed, 9
 - bandwidthAllocation, 9
 - grabMode, 9
 - grabTimeout, 9
 - isochBusSpeed, 9
 - minNumImageNotifications, 9
 - numBuffers, 9
 - numImageNotifications, 9
 - registerTimeout, 9
 - registerTimeoutRetries, 9
 - reserved, 9
- fc2ConfigROM, 9
 - chipIdHi, 10
 - chipIdLo, 10
 - nodeVendorId, 10
 - pszKeyword, 10
 - reserved, 10
 - unitSWVer, 10
 - unitSpecId, 10
 - unitSubSWVer, 10
 - vendorUniqueInfo_0, 10
 - vendorUniqueInfo_1, 10
 - vendorUniqueInfo_2, 10
 - vendorUniqueInfo_3, 10
- fc2Connect
 - FlyCapture2_C.h, 47
- fc2Context
 - FlyCapture2Defs_C.h, 95
- fc2ConvertImage
 - FlyCapture2_C.h, 48
- fc2ConvertImageTo
 - FlyCapture2_C.h, 48
- fc2CreateAVI
 - FlyCapture2_C.h, 48
- fc2CreateContext
 - FlyCapture2_C.h, 49
- fc2CreateGUIContext
 - FlyCapture2GUI_C.h, 106
- fc2CreateGigEContext
 - FlyCapture2_C.h, 49
- fc2CreateImage
 - FlyCapture2_C.h, 49
- fc2CreateImageStatistics
 - FlyCapture2_C.h, 50
- fc2DestroyAVI
 - FlyCapture2_C.h, 50
- fc2DestroyContext
 - FlyCapture2_C.h, 50
- fc2DestroyGUIContext
 - FlyCapture2GUI_C.h, 106
- fc2DestroyImage
 - FlyCapture2_C.h, 50
- fc2DestroyImageStatistics
 - FlyCapture2_C.h, 51
- fc2DetermineBitsPerPixel
 - FlyCapture2_C.h, 51
- fc2Disconnect
 - FlyCapture2_C.h, 51
- fc2DiscoverGigECameras
 - FlyCapture2_C.h, 52
- fc2Disonnect
 - FlyCapture2GUI_C.h, 107
- fc2DriverType
 - FlyCapture2Defs_C.h, 97
- fc2EmbeddedImageInfo, 11
 - GPIOPinState, 12
 - ROIPosition, 12
 - brightness, 11
 - exposure, 11
 - frameCounter, 12
 - gain, 12
 - shutter, 12
 - strobePattern, 12
 - timestamp, 12
 - whiteBalance, 12
- fc2EmbeddedImageInfoProperty, 12
 - available, 12
 - onOff, 12
- fc2EnableLUT

- FlyCapture2_C.h, [52](#)
- fc2Error
 - FlyCapture2Defs_C.h, [98](#)
- fc2ErrorToDescription
 - FlyCapture2_C.h, [52](#)
- fc2FireBusReset
 - FlyCapture2_C.h, [53](#)
- fc2FireSoftwareTrigger
 - FlyCapture2_C.h, [53](#)
- fc2FireSoftwareTriggerBroadcast
 - FlyCapture2_C.h, [53](#)
- fc2ForceAllIPAddressesAutomatically
 - FlyCapture2_C.h, [54](#)
- fc2ForceIPAddressAutomatically
 - FlyCapture2_C.h, [54](#)
- fc2ForceIPAddressToCamera
 - FlyCapture2_C.h, [54](#)
- fc2Format7ImageSettings, [12](#)
 - height, [13](#)
 - mode, [13](#)
 - offsetX, [13](#)
 - offsetY, [13](#)
 - pixelFormat, [13](#)
 - reserved, [13](#)
 - width, [13](#)
- fc2Format7Info, [13](#)
 - imageHStepSize, [14](#)
 - imageVStepSize, [14](#)
 - maxHeight, [14](#)
 - maxPacketSize, [14](#)
 - maxWidth, [14](#)
 - minPacketSize, [14](#)
 - mode, [14](#)
 - offsetHStepSize, [14](#)
 - offsetVStepSize, [14](#)
 - packetSize, [14](#)
 - percentage, [14](#)
 - pixelFormatBitField, [14](#)
 - reserved, [14](#)
 - vendorPixelFormatBitField, [14](#)
- fc2Format7PacketInfo, [14](#)
 - maxBytesPerPacket, [14](#)
 - recommendedBytesPerPacket, [15](#)
 - reserved, [15](#)
 - unitBytesPerPacket, [15](#)
- fc2FrameRate
 - FlyCapture2Defs_C.h, [99](#)
- fc2GUIConnect
 - FlyCapture2GUI_C.h, [107](#)
- fc2GetActiveLUTBank
 - FlyCapture2_C.h, [55](#)
- fc2GetCameraFromIndex
 - FlyCapture2_C.h, [55](#)
- fc2GetCameraFromSerialNumber
 - FlyCapture2_C.h, [55](#)
- fc2GetCameraInfo
 - FlyCapture2_C.h, [56](#)
- fc2GetCameraSerialNumberFromIndex
 - FlyCapture2_C.h, [56](#)
- fc2GetChannelStatus
 - FlyCapture2_C.h, [56](#)
- fc2GetConfiguration
 - FlyCapture2_C.h, [57](#)
- fc2GetCycleTime
 - FlyCapture2_C.h, [57](#)
- fc2GetDefaultColorProcessing
 - FlyCapture2_C.h, [58](#)
- fc2GetDefaultOutputFormat
 - FlyCapture2_C.h, [58](#)
- fc2GetDeviceFromIndex
 - FlyCapture2_C.h, [58](#)
- fc2GetEmbeddedImageInfo
 - FlyCapture2_C.h, [59](#)
- fc2GetFormat7Configuration
 - FlyCapture2_C.h, [59](#)
- fc2GetFormat7Info
 - FlyCapture2_C.h, [60](#)
- fc2GetGPIOPinDirection
 - FlyCapture2_C.h, [61](#)
- fc2GetGigEConfig
 - FlyCapture2_C.h, [60](#)
- fc2GetGigEImageBinningSettings
 - FlyCapture2_C.h, [60](#)
- fc2GetGigEImageSettings
 - FlyCapture2_C.h, [60](#)
- fc2GetGigEImageSettingsInfo
 - FlyCapture2_C.h, [60](#)
- fc2GetGigEImagingMode
 - FlyCapture2_C.h, [60](#)
- fc2GetGigEProperty
 - FlyCapture2_C.h, [60](#)
- fc2GetGigEStreamChannelInfo
 - FlyCapture2_C.h, [61](#)
- fc2GetImageData
 - FlyCapture2_C.h, [61](#)
- fc2GetImageStatistics
 - FlyCapture2_C.h, [62](#)
- fc2GetImageTimeStamp
 - FlyCapture2_C.h, [62](#)
- fc2GetInterfaceTypeFromGuid

- FlyCapture2_C.h, 62
- fc2GetLUTBankInfo
 - FlyCapture2_C.h, 63
- fc2GetLUTChannel
 - FlyCapture2_C.h, 64
- fc2GetLUTInfo
 - FlyCapture2_C.h, 64
- fc2GetLibraryVersion
 - FlyCapture2_C.h, 63
- fc2GetMemoryChannel
 - FlyCapture2_C.h, 64
- fc2GetMemoryChannelInfo
 - FlyCapture2_C.h, 65
- fc2GetNumOfCameras
 - FlyCapture2_C.h, 65
- fc2GetNumOfDevices
 - FlyCapture2_C.h, 65
- fc2GetNumStreamChannels
 - FlyCapture2_C.h, 66
- fc2GetProperty
 - FlyCapture2_C.h, 66
- fc2GetPropertyInfo
 - FlyCapture2_C.h, 66
- fc2GetRegisterString
 - FlyCapture2_C.h, 66
- fc2GetStrobe
 - FlyCapture2_C.h, 67
- fc2GetStrobeInfo
 - FlyCapture2_C.h, 67
- fc2GetSystemInfo
 - FlyCapture2_C.h, 67
- fc2GetTriggerDelay
 - FlyCapture2_C.h, 68
- fc2GetTriggerDelayInfo
 - FlyCapture2_C.h, 68
- fc2GetTriggerMode
 - FlyCapture2_C.h, 68
- fc2GetTriggerModelInfo
 - FlyCapture2_C.h, 68
- fc2GetVideoModeAndFrameRate
 - FlyCapture2_C.h, 69
- fc2GetVideoModeAndFrameRateInfo
 - FlyCapture2_C.h, 69
- fc2GigEConfig, 15
 - enablePacketResend, 15
 - maxPacketsToResend, 15
 - reserved, 15
 - timeoutForPacketResend, 15
- fc2GigEImageSettings, 16
 - height, 16
 - offsetX, 16
 - offsetY, 16
 - pixelFormat, 16
 - reserved, 16
 - width, 16
- fc2GigEImageSettingsInfo, 16
 - imageHStepSize, 17
 - imageVStepSize, 17
 - maxHeight, 17
 - maxWidth, 17
 - offsetHStepSize, 17
 - offsetVStepSize, 17
 - pixelFormatBitField, 17
 - reserved, 17
 - vendorPixelFormatBitField, 17
- fc2GigEProperty, 17
 - isReadable, 17
 - isWritable, 17
 - max, 17
 - min, 17
 - propType, 17
 - reserved, 18
 - value, 18
- fc2GigEPropertyType
 - FlyCapture2Defs_C.h, 100
- fc2GigEStreamChannel, 18
 - destinationIpAddress, 18
 - doNotFragment, 18
 - hostPort, 19
 - interPacketDelay, 19
 - networkInterfaceIndex, 19
 - packetSize, 19
 - reserved, 19
 - sourcePort, 19
- fc2GrabMode
 - FlyCapture2Defs_C.h, 100
- fc2GrabTimeout
 - FlyCapture2Defs_C.h, 100
- fc2GuiContext
 - FlyCapture2Defs_C.h, 95
- fc2H264Open
 - FlyCapture2_C.h, 70
- fc2H264Option, 19
 - bitrate, 19
 - frameRate, 19
 - height, 19
 - reserved, 19
 - width, 19
- fc2Hide
 - FlyCapture2GUI_C.h, 107

- fc2IPAddress, [23](#)
 - octets, [23](#)
- fc2Image, [20](#)
 - bayerFormat, [20](#)
 - cols, [20](#)
 - dataSize, [20](#)
 - format, [20](#)
 - imageImpl, [20](#)
 - pData, [20](#)
 - receivedDataSize, [20](#)
 - rows, [20](#)
 - stride, [20](#)
- fc2ImageEventCallback
 - FlyCapture2Defs_C.h, [95](#)
- fc2ImageFileFormat
 - FlyCapture2Defs_C.h, [100](#)
- fc2ImageImpl
 - FlyCapture2Defs_C.h, [96](#)
- fc2ImageMetadata, [20](#)
 - embeddedBrightness, [21](#)
 - embeddedExposure, [21](#)
 - embeddedFrameCounter, [21](#)
 - embeddedGPIOPinState, [21](#)
 - embeddedGain, [21](#)
 - embeddedROIPosition, [21](#)
 - embeddedShutter, [21](#)
 - embeddedStrobePattern, [21](#)
 - embeddedTimeStamp, [21](#)
 - embeddedWhiteBalance, [21](#)
 - reserved, [21](#)
- fc2ImageStatisticsContext
 - FlyCapture2Defs_C.h, [96](#)
- fc2InterfaceType
 - FlyCapture2Defs_C.h, [101](#)
- fc2InternalContext, [21](#)
 - pBusMgr, [22](#)
 - pCamera, [22](#)
- fc2InternalGuiContext, [22](#)
 - pCameraControlDlg, [22](#)
 - pCameraSelectionDlg, [22](#)
- fc2InternalImageCallback, [23](#)
 - pCallback, [23](#)
 - pCallbackData, [23](#)
- fc2IsCameraControlable
 - FlyCapture2_C.h, [70](#)
- fc2IsVisible
 - FlyCapture2GUI_C.h, [107](#)
- fc2JPEGOption, [24](#)
 - progressive, [24](#)
 - quality, [24](#)
 - reserved, [24](#)
- fc2JPG2Option, [24](#)
 - quality, [24](#)
 - reserved, [24](#)
- fc2LUTData, [25](#)
 - enabled, [25](#)
 - inputBitDepth, [25](#)
 - numBanks, [25](#)
 - numChannels, [25](#)
 - numEntries, [25](#)
 - outputBitDepth, [25](#)
 - reserved, [25](#)
 - supported, [25](#)
- fc2LaunchBrowser
 - FlyCapture2_C.h, [70](#)
- fc2LaunchCommand
 - FlyCapture2_C.h, [71](#)
- fc2LaunchCommandAsync
 - FlyCapture2_C.h, [71](#)
- fc2LaunchHelp
 - FlyCapture2_C.h, [71](#)
- fc2MACAddress, [25](#)
 - octets, [25](#)
- fc2MJPEGOpen
 - FlyCapture2_C.h, [71](#)
- fc2MJPEGOption, [26](#)
 - frameRate, [26](#)
 - quality, [26](#)
 - reserved, [26](#)
- fc2Mode
 - FlyCapture2Defs_C.h, [101](#)
- fc2OSType
 - FlyCapture2Defs_C.h, [102](#)
- fc2PCleBusSpeed
 - FlyCapture2Defs_C.h, [102](#)
- fc2PGMOption, [26](#)
 - binaryFile, [26](#)
 - reserved, [26](#)
- fc2PGRGuid, [27](#)
 - value, [27](#)
- fc2PNGOption, [27](#)
 - compressionLevel, [27](#)
 - interlaced, [27](#)
 - reserved, [27](#)
- fc2PPMOption, [28](#)
 - binaryFile, [28](#)
 - reserved, [28](#)
- fc2PixelFormat
 - FlyCapture2Defs_C.h, [103](#)
- fc2PropertyType

- FlyCapture2Defs_C.h, 103
- fc2QueryGigEImagingMode
 - FlyCapture2_C.h, 72
- fc2ReadGVCPMemory
 - FlyCapture2_C.h, 72
- fc2ReadGVCPRegister
 - FlyCapture2_C.h, 72
- fc2ReadGVCPRegisterBlock
 - FlyCapture2_C.h, 73
- fc2ReadRegister
 - FlyCapture2_C.h, 73
- fc2ReadRegisterBlock
 - FlyCapture2_C.h, 73
- fc2RegisterCallback
 - FlyCapture2_C.h, 74
- fc2RescanBus
 - FlyCapture2_C.h, 74
- fc2RestoreFromMemoryChannel
 - FlyCapture2_C.h, 74
- fc2RetrieveBuffer
 - FlyCapture2_C.h, 75
- fc2SavelImage
 - FlyCapture2_C.h, 75
- fc2SavelImageWithOption
 - FlyCapture2_C.h, 75
- fc2SaveToMemoryChannel
 - FlyCapture2_C.h, 76
- fc2SetActiveLUTBank
 - FlyCapture2_C.h, 76
- fc2SetCallback
 - FlyCapture2_C.h, 76
- fc2SetChannelStatus
 - FlyCapture2_C.h, 77
- fc2SetConfiguration
 - FlyCapture2_C.h, 77
- fc2SetDefaultColorProcessing
 - FlyCapture2_C.h, 77
- fc2SetDefaultOutputFormat
 - FlyCapture2_C.h, 78
- fc2SetEmbeddedImageInfo
 - FlyCapture2_C.h, 78
- fc2SetFormat7Configuration
 - FlyCapture2_C.h, 79
- fc2SetFormat7ConfigurationPacket
 - FlyCapture2_C.h, 79
- fc2SetGPIOPinDirection
 - FlyCapture2_C.h, 80
- fc2SetGPIOPinDirectionBroadcast
 - FlyCapture2_C.h, 80
- fc2SetGigEConfig
 - FlyCapture2_C.h, 79
- fc2SetGigEImageBinningSettings
 - FlyCapture2_C.h, 79
- fc2SetGigEImageSettings
 - FlyCapture2_C.h, 79
- fc2SetGigEImagingMode
 - FlyCapture2_C.h, 79
- fc2SetGigEProperty
 - FlyCapture2_C.h, 80
- fc2SetGigEStreamChannelInfo
 - FlyCapture2_C.h, 80
- fc2SetImageData
 - FlyCapture2_C.h, 81
- fc2SetImageDimensions
 - FlyCapture2_C.h, 81
- fc2SetLUTChannel
 - FlyCapture2_C.h, 81
- fc2SetProperty
 - FlyCapture2_C.h, 82
- fc2SetPropertyBroadcast
 - FlyCapture2_C.h, 82
- fc2SetStrobe
 - FlyCapture2_C.h, 83
- fc2SetStrobeBroadcast
 - FlyCapture2_C.h, 83
- fc2SetTriggerDelay
 - FlyCapture2_C.h, 83
- fc2SetTriggerDelayBroadcast
 - FlyCapture2_C.h, 84
- fc2SetTriggerMode
 - FlyCapture2_C.h, 84
- fc2SetTriggerModeBroadcast
 - FlyCapture2_C.h, 84
- fc2SetUserBuffers
 - FlyCapture2_C.h, 84
- fc2SetVideoModeAndFrameRate
 - FlyCapture2_C.h, 85
- fc2Show
 - FlyCapture2GUI_C.h, 108
- fc2ShowModal
 - FlyCapture2GUI_C.h, 108
- fc2StartCapture
 - FlyCapture2_C.h, 85
- fc2StartCaptureCallback
 - FlyCapture2_C.h, 86
- fc2StartSyncCapture
 - FlyCapture2_C.h, 86
- fc2StartSyncCaptureCallback
 - FlyCapture2_C.h, 86
- fc2StatisticsChannel

- FlyCapture2Defs_C.h, 104
- fc2StopCapture
 - FlyCapture2_C.h, 87
- fc2StrobeControl, 28
 - delay, 28
 - duration, 28
 - onOff, 28
 - polarity, 28
 - reserved, 28
 - source, 28
- fc2StrobeInfo, 29
 - maxValue, 29
 - minValue, 29
 - onOffSupported, 29
 - polaritySupported, 29
 - present, 29
 - readOutSupported, 29
 - reserved, 29
 - source, 29
- fc2SystemInfo, 29
 - byteOrder, 30
 - cpuDescription, 30
 - driverList, 30
 - gpuDescription, 30
 - libraryList, 30
 - numCpuCores, 30
 - osDescription, 30
 - osType, 30
 - reserved, 30
 - screenHeight, 30
 - screenWidth, 30
 - sysMemSize, 30
- fc2TIFFCompressionMethod
 - FlyCapture2Defs_C.h, 104
- fc2TIFFOption, 30
 - compression, 31
 - reserved, 31
- fc2TimeStamp, 31
 - cycleCount, 31
 - cycleOffset, 31
 - cycleSeconds, 31
 - microSeconds, 31
 - reserved, 31
 - seconds, 31
- fc2TriggerDelay, 31
 - absControl, 32
 - absValue, 32
 - autoManualMode, 32
 - onOff, 32
 - onePush, 32
 - present, 32
 - reserved, 32
 - type, 32
 - valueA, 32
 - valueB, 32
- fc2TriggerDelayInfo, 32
 - absMax, 33
 - absMin, 33
 - absValSupported, 33
 - autoSupported, 33
 - manualSupported, 33
 - max, 33
 - min, 33
 - onOffSupported, 33
 - onePushSupported, 33
 - pUnitAbbr, 33
 - pUnits, 33
 - present, 33
 - readOutSupported, 33
 - reserved, 33
 - type, 33
- fc2TriggerMode, 34
 - mode, 34
 - onOff, 34
 - parameter, 34
 - polarity, 34
 - reserved, 34
 - source, 34
- fc2TriggerModelInfo, 34
 - modeMask, 35
 - onOffSupported, 35
 - polaritySupported, 35
 - present, 35
 - readOutSupported, 35
 - reserved, 35
 - softwareTriggerSupported, 35
 - sourceMask, 35
 - valueReadable, 35
- fc2UnregisterCallback
 - FlyCapture2_C.h, 87
- fc2ValidateFormat7Settings
 - FlyCapture2_C.h, 87
- fc2Version, 35
 - build, 35
 - major, 35
 - minor, 35
 - type, 35
- fc2VideoMode
 - FlyCapture2Defs_C.h, 105
- fc2WriteGVCPMemory

- FlyCapture2_C.h, 88
- fc2WriteGVCPRegister
 - FlyCapture2_C.h, 88
- fc2WriteGVCPRegisterBlock
 - FlyCapture2_C.h, 89
- fc2WriteGVCPRegisterBroadcast
 - FlyCapture2_C.h, 89
- fc2WriteRegister
 - FlyCapture2_C.h, 89
- fc2WriteRegisterBlock
 - FlyCapture2_C.h, 90
- fc2WriteRegisterBroadcast
 - FlyCapture2_C.h, 90
- firmwareBuildTime
 - fc2CameraInfo, 7
- firmwareVersion
 - fc2CameraInfo, 7
- format
 - fc2Image, 20
- frameCounter
 - fc2EmbeddedImageInfo, 12
- frameRate
 - fc2AVIOption, 5
 - fc2H264Option, 19
 - fc2MJPEGOption, 26
- gain
 - fc2EmbeddedImageInfo, 12
- gigEMajorVersion
 - fc2CameraInfo, 7
- gigEMinorVersion
 - fc2CameraInfo, 7
- gpuDescription
 - fc2SystemInfo, 30
- grabMode
 - fc2Config, 9
- grabTimeout
 - fc2Config, 9
- height
 - fc2Format7ImageSettings, 13
 - fc2GigEImageSettings, 16
 - fc2H264Option, 19
- hostPost
 - fc2GigEStreamChannel, 19
- iidcVer
 - fc2CameraInfo, 8
- imageHStepSize
 - fc2Format7Info, 14
- fc2GigEImageSettingsInfo, 17
- imageImpl
 - fc2Image, 20
- imageVStepSize
 - fc2Format7Info, 14
 - fc2GigEImageSettingsInfo, 17
- inputBitDepth
 - fc2LUTData, 25
- interPacketDelay
 - fc2GigEStreamChannel, 19
- interfaceType
 - fc2CameraInfo, 8
- interlaced
 - fc2PNGOption, 27
- ipAddress
 - fc2CameraInfo, 8
- isColorCamera
 - fc2CameraInfo, 8
- isReadable
 - fc2GigEProperty, 17
- isWritable
 - fc2GigEProperty, 17
- isochBusSpeed
 - fc2Config, 9
- libraryList
 - fc2SystemInfo, 30
- macAddress
 - fc2CameraInfo, 8
- major
 - fc2Version, 35
- manualSupported
 - fc2TriggerDelayInfo, 33
- max
 - fc2GigEProperty, 17
 - fc2TriggerDelayInfo, 33
- maxBytesPerPacket
 - fc2Format7PacketInfo, 14
- maxHeight
 - fc2Format7Info, 14
 - fc2GigEImageSettingsInfo, 17
- maxPacketSize
 - fc2Format7Info, 14
- maxPacketsToResend
 - fc2GigEConfig, 15
- maxValue
 - fc2StrobeInfo, 29
- maxWidth
 - fc2Format7Info, 14

- fc2GigEImageSettingsInfo, [17](#)
- maximumBusSpeed
 - fc2CameraInfo, [8](#)
- microSeconds
 - fc2TimeStamp, [31](#)
- min
 - fc2GigEProperty, [17](#)
 - fc2TriggerDelayInfo, [33](#)
- minNumImageNotifications
 - fc2Config, [9](#)
- minPacketSize
 - fc2Format7Info, [14](#)
- minValue
 - fc2StrobeInfo, [29](#)
- minor
 - fc2Version, [35](#)
- mode
 - fc2Format7ImageSettings, [13](#)
 - fc2Format7Info, [14](#)
 - fc2TriggerMode, [34](#)
- modeMask
 - fc2TriggerModelInfo, [35](#)
- modelName
 - fc2CameraInfo, [8](#)
- networkInterfaceIndex
 - fc2GigEStreamChannel, [19](#)
- nodeNumber
 - fc2CameraInfo, [8](#)
- nodeVendorId
 - fc2ConfigROM, [10](#)
- numBanks
 - fc2LUTData, [25](#)
- numBuffers
 - fc2Config, [9](#)
- numChannels
 - fc2LUTData, [25](#)
- numCpuCores
 - fc2SystemInfo, [30](#)
- numEntries
 - fc2LUTData, [25](#)
- numImageNotifications
 - fc2Config, [9](#)
- octets
 - fc2IPAddress, [23](#)
 - fc2MACAddress, [25](#)
- offsetHStepSize
 - fc2Format7Info, [14](#)
 - fc2GigEImageSettingsInfo, [17](#)
- offsetVStepSize
 - fc2Format7Info, [14](#)
 - fc2GigEImageSettingsInfo, [17](#)
- offsetX
 - fc2Format7ImageSettings, [13](#)
 - fc2GigEImageSettings, [16](#)
- offsetY
 - fc2Format7ImageSettings, [13](#)
 - fc2GigEImageSettings, [16](#)
- onOff
 - fc2EmbeddedImageInfoProperty, [12](#)
 - fc2StrobeControl, [28](#)
 - fc2TriggerDelay, [32](#)
 - fc2TriggerMode, [34](#)
- onOffSupported
 - fc2StrobeInfo, [29](#)
 - fc2TriggerDelayInfo, [33](#)
 - fc2TriggerModelInfo, [35](#)
- onePush
 - fc2TriggerDelay, [32](#)
- onePushSupported
 - fc2TriggerDelayInfo, [33](#)
- osDescription
 - fc2SystemInfo, [30](#)
- osType
 - fc2SystemInfo, [30](#)
- outputBitDepth
 - fc2LUTData, [25](#)
- pBusMgr
 - fc2InternalContext, [22](#)
- pCallback
 - fc2InternalImageCallback, [23](#)
- pCallbackData
 - fc2InternalImageCallback, [23](#)
- pCamera
 - fc2InternalContext, [22](#)
- pCameraControlDlg
 - fc2InternalGuiContext, [22](#)
- pCameraSelectionDlg
 - fc2InternalGuiContext, [22](#)
- pData
 - fc2Image, [20](#)
- pUnitAbbr
 - fc2TriggerDelayInfo, [33](#)
- pUnits
 - fc2TriggerDelayInfo, [33](#)
- packetSize
 - fc2Format7Info, [14](#)
 - fc2GigEStreamChannel, [19](#)

- parameter
 - fc2TriggerMode, 34
- pcieBusSpeed
 - fc2CameraInfo, 8
- percentage
 - fc2Format7Info, 14
- pixelFormat
 - fc2Format7ImageSettings, 13
 - fc2GigEImageSettings, 16
- pixelFormatBitField
 - fc2Format7Info, 14
 - fc2GigEImageSettingsInfo, 17
- polarity
 - fc2StrobeControl, 28
 - fc2TriggerMode, 34
- polaritySupported
 - fc2StrobeInfo, 29
 - fc2TriggerModelInfo, 35
- present
 - fc2StrobeInfo, 29
 - fc2TriggerDelay, 32
 - fc2TriggerDelayInfo, 33
 - fc2TriggerModelInfo, 35
- progressive
 - fc2JPEGOption, 24
- propType
 - fc2GigEProperty, 17
- pszKeyword
 - fc2ConfigROM, 10
- quality
 - fc2JPEGOption, 24
 - fc2JPG2Option, 24
 - fc2MJPGOption, 26
- readOutSupported
 - fc2StrobeInfo, 29
 - fc2TriggerDelayInfo, 33
 - fc2TriggerModelInfo, 35
- receivedDataSize
 - fc2Image, 20
- recommendedBytesPerPacket
 - fc2Format7PacketInfo, 15
- registerTimeout
 - fc2Config, 9
- registerTimeoutRetries
 - fc2Config, 9
- reserved
 - fc2AVIOption, 5
 - fc2CameraInfo, 8
 - fc2Config, 9
 - fc2ConfigROM, 10
 - fc2Format7ImageSettings, 13
 - fc2Format7Info, 14
 - fc2Format7PacketInfo, 15
 - fc2GigEConfig, 15
 - fc2GigEImageSettings, 16
 - fc2GigEImageSettingsInfo, 17
 - fc2GigEProperty, 18
 - fc2GigEStreamChannel, 19
 - fc2H264Option, 19
 - fc2ImageMetadata, 21
 - fc2JPEGOption, 24
 - fc2JPG2Option, 24
 - fc2LUTData, 25
 - fc2MJPGOption, 26
 - fc2PGMOption, 26
 - fc2PNGOption, 27
 - fc2PPMOption, 28
 - fc2StrobeControl, 28
 - fc2StrobeInfo, 29
 - fc2SystemInfo, 30
 - fc2TIFFOption, 31
 - fc2TimeStamp, 31
 - fc2TriggerDelay, 32
 - fc2TriggerDelayInfo, 33
 - fc2TriggerMode, 34
 - fc2TriggerModelInfo, 35
- rows
 - fc2Image, 20
- screenHeight
 - fc2SystemInfo, 30
- screenWidth
 - fc2SystemInfo, 30
- seconds
 - fc2TimeStamp, 31
- sensorInfo
 - fc2CameraInfo, 8
- sensorResolution
 - fc2CameraInfo, 8
- serialNumber
 - fc2CameraInfo, 8
- shutter
 - fc2EmbeddedImageInfo, 12
- softwareTriggerSupported
 - fc2TriggerModelInfo, 35
- source
 - fc2StrobeControl, 28
 - fc2StrobeInfo, 29

- fc2TriggerMode, [34](#)
- sourceMask
 - fc2TriggerModelInfo, [35](#)
- sourcePort
 - fc2GigEStreamChannel, [19](#)
- stride
 - fc2Image, [20](#)
- strobePattern
 - fc2EmbeddedImageInfo, [12](#)
- subnetMask
 - fc2CameraInfo, [8](#)
- supported
 - fc2LUTData, [25](#)
- syncContext
 - MultiSyncLibraryDefs_C.h, [114](#)
- syncCreateContext
 - MultiSyncLibrary_C.h, [110](#)
- syncDestroyContext
 - MultiSyncLibrary_C.h, [110](#)
- syncDisableCrossPCSSynchronization
 - MultiSyncLibrary_C.h, [111](#)
- syncEnableCrossPCSSynchronization
 - MultiSyncLibrary_C.h, [111](#)
- syncError
 - MultiSyncLibraryDefs_C.h, [115](#)
- syncGetStatus
 - MultiSyncLibrary_C.h, [111](#)
- syncGetTimeSinceSynced
 - MultiSyncLibrary_C.h, [112](#)
- syncIsTimingBusConnected
 - MultiSyncLibrary_C.h, [112](#)
- syncMessage
 - MultiSyncLibraryDefs_C.h, [115](#)
- syncQueryCrossPCSSynchronization-Setting
 - MultiSyncLibrary_C.h, [112](#)
- syncRescanMasterTimingBus
 - MultiSyncLibrary_C.h, [113](#)
- syncStart
 - MultiSyncLibrary_C.h, [113](#)
- syncStop
 - MultiSyncLibrary_C.h, [113](#)
- sysMemSize
 - fc2SystemInfo, [30](#)
- timeoutForPacketResend
 - fc2GigEConfig, [15](#)
- timestamp
 - fc2EmbeddedImageInfo, [12](#)
- type
 - fc2TriggerDelay, [32](#)
 - fc2TriggerDelayInfo, [33](#)
 - fc2Version, [35](#)
- unitBytesPerPacket
 - fc2Format7PacketInfo, [15](#)
- unitSWVer
 - fc2ConfigROM, [10](#)
- unitSpecId
 - fc2ConfigROM, [10](#)
- unitSubSWVer
 - fc2ConfigROM, [10](#)
- userDefinedName
 - fc2CameraInfo, [8](#)
- value
 - fc2GigEProperty, [18](#)
 - fc2PGRGuid, [27](#)
- valueA
 - fc2TriggerDelay, [32](#)
- valueB
 - fc2TriggerDelay, [32](#)
- valueReadable
 - fc2TriggerModelInfo, [35](#)
- vendorName
 - fc2CameraInfo, [8](#)
- vendorPixelFormatBitField
 - fc2Format7Info, [14](#)
 - fc2GigEImageSettingsInfo, [17](#)
- vendorUniqueInfo_0
 - fc2ConfigROM, [10](#)
- vendorUniqueInfo_1
 - fc2ConfigROM, [10](#)
- vendorUniqueInfo_2
 - fc2ConfigROM, [10](#)
- vendorUniqueInfo_3
 - fc2ConfigROM, [10](#)
- whiteBalance
 - fc2EmbeddedImageInfo, [12](#)
- width
 - fc2Format7ImageSettings, [13](#)
 - fc2GigEImageSettings, [16](#)
 - fc2H264Option, [19](#)
- xmlURL1
 - fc2CameraInfo, [8](#)
- xmlURL2
 - fc2CameraInfo, [8](#)