



eYS3D
Microelectronics

eYs3D Linux SDK
v5.0.0.11

Generated by Doxygen 1.8.13

Contents

1	Introduction	1
2	Background	3
3	Background	9
4	Background	15
5	Background	21
6	Class Index	27
6.1	Class List	27
7	File Index	29
7.1	File List	29
8	Class Documentation	31
8.1	AccelerationTag Struct Reference	31
8.2	APCImageType Struct Reference	31
8.3	CompassTag Struct Reference	32
8.4	eSPCtrl_RectLogData Struct Reference	32
8.4.1	Member Data Documentation	33
8.4.1.1	CamDist1	33
8.4.1.2	CamDist2	33
8.4.1.3	CamMat1	33
8.4.1.4	CamMat2	33
8.4.1.5	InImgHeight	33

8.4.1.6	InImgWidth	33
8.4.1.7	LRotaMat	33
8.4.1.8	NewCamMat1	34
8.4.1.9	NewCamMat2	34
8.4.1.10	nLineBuffers	34
8.4.1.11	OutImgHeight	34
8.4.1.12	OutImgWidth	34
8.4.1.13	RECT_AvgErr	34
8.4.1.14	RECT_Crop_Col_BG_L	34
8.4.1.15	RECT_Crop_Col_ED_L	35
8.4.1.16	RECT_Crop_Row_BG	35
8.4.1.17	RECT_Crop_Row_ED	35
8.4.1.18	RECT_CropEnable	35
8.4.1.19	RECT_Scale_Col_M	35
8.4.1.20	RECT_Scale_Col_N	35
8.4.1.21	RECT_Scale_Row_M	35
8.4.1.22	RECT_Scale_Row_N	35
8.4.1.23	RECT_ScaleEnable	36
8.4.1.24	RECT_ScaleHeight	36
8.4.1.25	RECT_ScaleWidth	36
8.4.1.26	RotaMat	36
8.4.1.27	RRotaMat	36
8.4.1.28	TranMat	36
8.4.1.29	uByteArray	36
8.5	GyroTag Struct Reference	37
8.6	packet_s Struct Reference	37
8.7	PointCloudInfo Struct Reference	37
8.8	tagAPC_STREAM_INFO Struct Reference	38
8.9	tagDEVINFORMATION Struct Reference	38
8.10	tagDEVSEL Struct Reference	38
8.11	tagKEEP_DATA_CTRL Struct Reference	38
8.12	tagZDTableInfo Struct Reference	39

9	File Documentation	41
9.1	eSPDI/eSPDI.h File Reference	41
9.1.1	Detailed Description	50
9.1.2	Function Documentation	50
9.1.2.1	APC_ApplyFilters()	51
9.1.2.2	APC_CloseCmdFiFo()	51
9.1.2.3	APC_CloseDevice()	52
9.1.2.4	APC_CloseDeviceEx()	52
9.1.2.5	APC_CloseDeviceMBL()	52
9.1.2.6	APC_ColorFormat_to_RGB24()	53
9.1.2.7	APC_Convert_Depth_Y_To_Buffer()	53
9.1.2.8	APC_Convert_Depth_Y_To_Buffer_offset()	54
9.1.2.9	APC_CreateSwPostProc()	54
9.1.2.10	APC_DecryptMP4()	55
9.1.2.11	APC_DecryptString() [1/2]	55
9.1.2.12	APC_DecryptString() [2/2]	56
9.1.2.13	APC_DepthMerge()	56
9.1.2.14	APC_DisableAE()	57
9.1.2.15	APC_DisableAWB()	57
9.1.2.16	APC_DoFusion()	58
9.1.2.17	APC_DoSwPostProc()	58
9.1.2.18	APC_EdgePreServingFilter()	59
9.1.2.19	APC_EnableAE()	60
9.1.2.20	APC_EnableAWB()	60
9.1.2.21	APC_EnableGPUAcceleration()	60
9.1.2.22	APC_EnableInterleave()	62
9.1.2.23	APC_EnableSensorIF()	62
9.1.2.24	APC_EncryptMP4()	63
9.1.2.25	APC_EncryptString() [1/2]	63
9.1.2.26	APC_EncryptString() [2/2]	63

9.1.2.27	APC_FindDevice()	64
9.1.2.28	APC_FlyingDepthCancellation_D11()	64
9.1.2.29	APC_FlyingDepthCancellation_D8()	65
9.1.2.30	APC_GenerateLutFile()	65
9.1.2.31	APC_Get2Image()	66
9.1.2.32	APC_Get_150_mm_depth()	66
9.1.2.33	APC_Get_60_mm_depth()	67
9.1.2.34	APC_Get_Color_30_mm_depth()	67
9.1.2.35	APC_GetAccMeterValue()	68
9.1.2.36	APC_GetAESTatus()	68
9.1.2.37	APC_GetAutoExposureMode()	69
9.1.2.38	APC_GetAWBStatus()	69
9.1.2.39	APC_GetBusInfo()	70
9.1.2.40	APC_GetColorGain()	70
9.1.2.41	APC_GetColorImage()	71
9.1.2.42	APC_GetColorImageWithTimestamp()	71
9.1.2.43	APC_GetControlCounterMode()	72
9.1.2.44	APC_GetCTPropVal()	72
9.1.2.45	APC_GetCTRangeAndStep()	73
9.1.2.46	APC_GetCurrentIRValue()	74
9.1.2.47	APC_GetDepthDataType()	74
9.1.2.48	APC_GetDepthImage()	75
9.1.2.49	APC_GetDepthImageWithTimestamp()	75
9.1.2.50	APC_GetDeviceInfo()	76
9.1.2.51	APC_GetDeviceInfoMBL_15cm()	76
9.1.2.52	APC_GetDeviceNumber()	77
9.1.2.53	APC_GetDeviceResolutionList()	77
9.1.2.54	APC_GetExposureTime()	78
9.1.2.55	APC_GetFlexibleGyroData()	78
9.1.2.56	APC_GetFlexibleGyroLength()	79

9.1.2.57	APC_GetFWRegister()	79
9.1.2.58	APC_GetFwVersion()	80
9.1.2.59	APC_GetGlobalGain()	80
9.1.2.60	APC_GetHidGyro()	81
9.1.2.61	APC_GetHWRegister()	81
9.1.2.62	APC_GetImage()	82
9.1.2.63	APC_GetImageInterrupt()	82
9.1.2.64	APC_GetInfoHidGyro()	83
9.1.2.65	APC_GetInterleaveMode()	84
9.1.2.66	APC_GetIRMaxValue()	84
9.1.2.67	APC_GetIRMinValue()	85
9.1.2.68	APC_GetIRMode()	85
9.1.2.69	APC_GetLogData()	86
9.1.2.70	APC_GetLutData()	86
9.1.2.71	APC_GetMultiBytesHWRegister()	87
9.1.2.72	APC_GetPidVid()	87
9.1.2.73	APC_GetPointCloud()	88
9.1.2.74	APC_GetPUPropVal()	88
9.1.2.75	APC_GetPURangeAndStep()	89
9.1.2.76	APC_GetRectifyLogData()	90
9.1.2.77	APC_GetRectifyMatLogData()	90
9.1.2.78	APC_GetRectifyTable()	91
9.1.2.79	APC_GetSensorRegister()	91
9.1.2.80	APC_GetSerialNumber()	92
9.1.2.81	APC_GetSRB()	92
9.1.2.82	APC_GetThermalFD()	93
9.1.2.83	APC_GetUACData()	93
9.1.2.84	APC_getUACNAME()	93
9.1.2.85	APC_GetUserData()	94
9.1.2.86	APC_GetYOffset()	94

9.1.2.87	APC_GetZDTable()	95
9.1.2.88	APC_HoleFill()	95
9.1.2.89	APC_HoleFilled()	96
9.1.2.90	APC_ImgMirro() [1/2]	96
9.1.2.91	APC_ImgMirro() [2/2]	98
9.1.2.92	APC_Init()	98
9.1.2.93	APC_InitialCmdFiFo()	99
9.1.2.94	APC_InitialFlexibleGyro()	99
9.1.2.95	APC_InitialHidGyro()	100
9.1.2.96	APC_InitialUAC()	100
9.1.2.97	APC_InitPostProcess()	100
9.1.2.98	APC_InitSRB()	101
9.1.2.99	APC_InjectExtraDataToMp4()	101
9.1.2.100	APC_IsInterleaveDevice()	102
9.1.2.101	APC_IsMLBaseLine()	102
9.1.2.102	APC_OpenDevice()	102
9.1.2.103	APC_OpenDevice2()	103
9.1.2.104	APC_OpenDeviceMBL()	104
9.1.2.105	APC_PostProcess()	105
9.1.2.106	APC_PutSRB()	106
9.1.2.107	APC_ReadCmdFiFo()	106
9.1.2.108	APC_ReadFlashData()	106
9.1.2.109	APC_RefreshDevice()	107
9.1.2.110	APC_Release()	107
9.1.2.111	APC_ReleaseFlexibleGyro()	108
9.1.2.112	APC_ReleaseHidGyro()	108
9.1.2.113	APC_ReleasePostProcess()	108
9.1.2.114	APC_ReleaseSwPostProc()	109
9.1.2.115	APC_ReleaseUAC()	109
9.1.2.116	APC_ResetFilters()	109

9.1.2.117 APC_ResizeImgToHalf()	110
9.1.2.118 APC_RetrieveExtraDataFromMp4()	110
9.1.2.119 APC_RGB2BMP()	111
9.1.2.120 APC_RotateImg180() [1/2]	111
9.1.2.121 APC_RotateImg180() [2/2]	112
9.1.2.122 APC_RotateImg90() [1/2]	112
9.1.2.123 APC_RotateImg90() [2/2]	113
9.1.2.124 APC_SaveLutData()	114
9.1.2.125 APC_SelectDevice()	114
9.1.2.126 APC_SetAutoExposureMode()	114
9.1.2.127 APC_SetColorGain()	115
9.1.2.128 APC_SetControlCounterMode()	115
9.1.2.129 APC_SetCTPropVal()	116
9.1.2.130 APC_SetCurrentIRValue()	116
9.1.2.131 APC_SetDepthDataType()	117
9.1.2.132 APC_SetExposureTime()	117
9.1.2.133 APC_SetFWRegister()	118
9.1.2.134 APC_SetGlobalGain()	118
9.1.2.135 APC_SetHWRegister()	119
9.1.2.136 APC_SetInterleaveMode()	119
9.1.2.137 APC_SetIRMaxValue()	120
9.1.2.138 APC_SetIRMode()	120
9.1.2.139 APC_SetLogData()	121
9.1.2.140 APC_SetMultiBytesHWRegister()	121
9.1.2.141 APC_SetPidVid()	122
9.1.2.142 APC_SetPUPropVal()	122
9.1.2.143 APC_SetRectifyTable()	123
9.1.2.144 APC_SetSensorRegister()	123
9.1.2.145 APC_SetSensorTypeName()	124
9.1.2.146 APC_SetSerialNumber()	124

9.1.2.147 APC_Setup_v4l2_requestbuffers()	125
9.1.2.148 APC_SetupBlock()	125
9.1.2.149 APC_SetupHidGyro()	125
9.1.2.150 APC_SetUserData()	126
9.1.2.151 APC_SetYOffset()	126
9.1.2.152 APC_SetZDTable()	127
9.1.2.153 APC_SubSample()	127
9.1.2.154 APC_SwitchBaseline()	128
9.1.2.155 APC_TableToData()	129
9.1.2.156 APC_TemporalFilter()	129
9.1.2.157 APC_WriteCmdFiFo()	130
9.1.2.158 APC_WriteFlashData()	130
9.1.2.159 APC_WriteWaveEnd()	131
9.1.2.160 APC_WriteWaveHeader()	131
9.2 eSPDI/eSPDI_def.h File Reference	131
9.2.1 Detailed Description	139
Index	141

Chapter 1

Introduction

This document describes the usage of eYs3D Linux SDK

What's inside the SDK

Table 1.1 File List

Folder	Filename	Description
bin	All files	sample executables on Linux platform
console_tester	All files	a console program demonstrating how to use the APIs defined in eSPDI.h
cfg	All files	configuration files
eSPDI	eSPDI.h	functions definitions
	eSPDI_def.h	error/data type definitions
	eSPDI↔ version.h	SDK version declaration header
DMPreview	All files	a sample project demonstrating how to open multiple devices in an application

Chapter 2

Background

libjpeg-turbo is a JPEG image codec that uses SIMD instructions to accelerate baseline JPEG compression and decompression on x86, x86-64, ARM, PowerPC, and MIPS systems, as well as progressive JPEG compression on x86 and x86-64 systems. On such systems, libjpeg-turbo is generally 2-6x as fast as libjpeg, all else being equal. On other types of systems, libjpeg-turbo can still outperform libjpeg by a significant amount, by virtue of its highly-optimized Huffman coding routines. In many cases, the performance of libjpeg-turbo rivals that of proprietary high-speed JPEG codecs.

libjpeg-turbo implements both the traditional libjpeg API as well as the less powerful but more straightforward TurboJPEG API. libjpeg-turbo also features colorspace extensions that allow it to compress from/decompress to 32-bit and big-endian pixel buffers (RGBX, XBGR, etc.), as well as a full-featured Java interface.

libjpeg-turbo was originally based on libjpeg/SIMD, an MMX-accelerated derivative of libjpeg v6b developed by Miyasaka Masaru. The TigerVNC and VirtualGL projects made numerous enhancements to the codec in 2009, and in early 2010, libjpeg-turbo spun off into an independent project, with the goal of making high-speed JPEG compression/decompression technology available to a broader range of users and developers.

License

libjpeg-turbo is covered by three compatible BSD-style open source licenses. Refer to LICENSE.md for a roll-up of license terms.

Building libjpeg-turbo

Refer to BUILDING.md for complete instructions.

Using libjpeg-turbo

libjpeg-turbo includes two APIs that can be used to compress and decompress JPEG images:

- **TurboJPEG API**

This API provides an easy-to-use interface for compressing and decompressing JPEG images in memory. It also provides some functionality that would not be straightforward to achieve using the underlying libjpeg API, such as generating planar YUV images and performing multiple simultaneous lossless transforms on an image. The Java interface for libjpeg-turbo is written on top of the TurboJPEG API. The TurboJPEG API is recommended for first-time users of libjpeg-turbo. Refer to [tjexample.c](#) and [TJExample.java](#) for examples of its usage and to <http://libjpeg-turbo.org/Documentation/Documentation> for API documentation.

- **libjpeg API**

This is the de facto industry-standard API for compressing and decompressing JPEG images. It is more difficult to use than the TurboJPEG API but also more powerful. The libjpeg API implementation in libjpeg-turbo is both API/ABI-compatible and mathematically compatible with libjpeg v6b. It can also optionally be configured to be API/ABI-compatible with libjpeg v7 and v8 (see below.) Refer to [cjpeg.c](#) and [djpeg.c](#) for examples of its usage and to [libjpeg.txt](#) for API documentation.

There is no significant performance advantage to either API when both are used to perform similar operations.

Colorspace Extensions

libjpeg-turbo includes extensions that allow JPEG images to be compressed directly from (and decompressed directly to) buffers that use BGR, BGRX, RGBX, XBGR, and XRGB pixel ordering. This is implemented with ten new colorspace constants:

```
JCS_EXT_RGB    /* red/green/blue */
JCS_EXT_RGBX   /* red/green/blue/x */
JCS_EXT_BGR    /* blue/green/red */
JCS_EXT_BGRX   /* blue/green/red/x */
JCS_EXT_XBGR   /* x/blue/green/red */
JCS_EXT_XRGB   /* x/red/green/blue */
JCS_EXT_RGBA   /* red/green/blue/alpha */
JCS_EXT_BGRA   /* blue/green/red/alpha */
JCS_EXT_ABGR   /* alpha/blue/green/red */
JCS_EXT_ARGB   /* alpha/red/green/blue */
```

Setting `cinfo.in_color_space` (compression) or `cinfo.out_color_space` (decompression) to one of these values will cause libjpeg-turbo to read the red, green, and blue values from (or write them to) the appropriate position in the pixel when compressing from/decompressing to an RGB buffer.

Your application can check for the existence of these extensions at compile time with:

```
#ifndef JCS_EXTENSIONS
```

At run time, attempting to use these extensions with a libjpeg implementation that does not support them will result in a "Bogus input colorspace" error. Applications can trap this error in order to test whether run-time support is available for the colorspace extensions.

When using the RGBX, BGRX, XBGR, and XRGB colorspace during decompression, the X byte is undefined, and in order to ensure the best performance, libjpeg-turbo can set that byte to whatever value it wishes. If an application expects the X byte to be used as an alpha channel, then it should specify `JCS_EXT_RGBA`, `JCS_EXT_BGRA`, `JCS_EXT_ABGR`, or `JCS_EXT_ARGB`. When these colorspace constants are used, the X byte is guaranteed to be 0xFF, which is interpreted as opaque.

Your application can check for the existence of the alpha channel colorspace extensions at compile time with:

```
#ifndef JCS_ALPHA_EXTENSIONS
```

[jcstest.c](#), located in the libjpeg-turbo source tree, demonstrates how to check for the existence of the colorspace extensions at compile time and run time.

libjpeg v7 and v8 API/ABI Emulation

With libjpeg v7 and v8, new features were added that necessitated extending the compression and decompression structures. Unfortunately, due to the exposed nature of those structures, extending them also necessitated breaking backward ABI compatibility with previous libjpeg releases. Thus, programs that were built to use libjpeg v7 or v8 did not work with libjpeg-turbo, since it is based on the libjpeg v6b code base. Although libjpeg v7 and v8 are not as widely used as v6b, enough programs (including a few Linux distros) made the switch that there was a demand to emulate the libjpeg v7 and v8 ABIs in libjpeg-turbo. It should be noted, however, that this feature was added primarily so that applications that had already been compiled to use libjpeg v7+ could take advantage of accelerated baseline JPEG encoding/decoding without recompiling. libjpeg-turbo does not claim to support all of the libjpeg v7+ features, nor to produce identical output to libjpeg v7+ in all cases (see below.)

By passing an argument of `-DWITH_JPEG7=1` or `-DWITH_JPEG8=1` to `cmake`, you can build a version of libjpeg-turbo that emulates the libjpeg v7 or v8 ABI, so that programs that are built against libjpeg v7 or v8 can be run with libjpeg-turbo. The following section describes which libjpeg v7+ features are supported and which aren't.

Support for libjpeg v7 and v8 Features

Fully supported

- **libjpeg API: IDCT scaling extensions in decompressor**
libjpeg-turbo supports IDCT scaling with scaling factors of 1/8, 1/4, 3/8, 1/2, 5/8, 3/4, 7/8, 9/8, 5/4, 11/8, 3/2, 13/8, 7/4, 15/8, and 2/1 (only 1/4 and 1/2 are SIMD-accelerated.)
- **libjpeg API: Arithmetic coding**
- **libjpeg API: In-memory source and destination managers**
See notes below.
- **cjpeg: Separate quality settings for luminance and chrominance**
Note that the libjpeg v7+ API was extended to accommodate this feature only for convenience purposes. It has always been possible to implement this feature with libjpeg v6b (see `rdswitch.c` for an example.)
- **cjpeg: 32-bit BMP support**
- **cjpeg: `-rgb` option**
- **jpegtran: Lossless cropping**
- **jpegtran: `-perfect` option**
- **jpegtran: Forcing width/height when performing lossless crop**
- **rdjpgcom: `-raw` option**
- **rdjpgcom: Locale awareness**

Not supported

NOTE: As of this writing, extensive research has been conducted into the usefulness of DCT scaling as a means of data reduction and SmartScale as a means of quality improvement. The reader is invited to peruse the research at <http://www.libjpeg-turbo.org/About/SmartScale> and draw his/her own conclusions, but it is the general belief of our project that these features have not demonstrated sufficient usefulness to justify inclusion in libjpeg-turbo.

- **libjpeg API: DCT scaling in compressor**
`cinfo.scale_num` and `cinfo.scale_denom` are silently ignored. There is no technical reason why DCT scaling could not be supported when emulating the libjpeg v7+ API/ABI, but without the SmartScale extension (see below), only scaling factors of 1/2, 8/15, 4/7, 8/13, 2/3, 8/11, 4/5, and 8/9 would be available, which is of limited usefulness.

- **libjpeg API: SmartScale**

`cinfo.block_size` is silently ignored. SmartScale is an extension to the JPEG format that allows for DCT block sizes other than 8x8. Providing support for this new format would be feasible (particularly without full acceleration.) However, until/unless the format becomes either an official industry standard or, at minimum, an accepted solution in the community, we are hesitant to implement it, as there is no sense of whether or how it might change in the future. It is our belief that SmartScale has not demonstrated sufficient usefulness as a lossless format nor as a means of quality enhancement, and thus our primary interest in providing this feature would be as a means of supporting additional DCT scaling factors.

- **libjpeg API: Fancy downsampling in compressor**

`cinfo.do_fancy_downsampling` is silently ignored. This requires the DCT scaling feature, which is not supported.

- **jpegtran: Scaling**

This requires both the DCT scaling and SmartScale features, which are not supported.

- **Lossless RGB JPEG files**

This requires the SmartScale feature, which is not supported.

What About libjpeg v9?

libjpeg v9 introduced yet another field to the JPEG compression structure (`color_transform`), thus making the ABI backward incompatible with that of libjpeg v8. This new field was introduced solely for the purpose of supporting lossless SmartScale encoding. Furthermore, there was actually no reason to extend the API in this manner, as the color transform could have just as easily been activated by way of a new JPEG colorspace constant, thus preserving backward ABI compatibility.

Our research (see link above) has shown that lossless SmartScale does not generally accomplish anything that can't already be accomplished better with existing, standard lossless formats. Therefore, at this time it is our belief that there is not sufficient technical justification for software projects to upgrade from libjpeg v8 to libjpeg v9, and thus there is not sufficient technical justification for us to emulate the libjpeg v9 ABI.

In-Memory Source/Destination Managers

By default, libjpeg-turbo 1.3 and later includes the `jpeg_mem_src()` and `jpeg_mem_dest()` functions, even when not emulating the libjpeg v8 API/ABI. Previously, it was necessary to build libjpeg-turbo from source with libjpeg v8 API/ABI emulation in order to use the in-memory source/destination managers, but several projects requested that those functions be included when emulating the libjpeg v6b API/ABI as well. This allows the use of those functions by programs that need them, without breaking ABI compatibility for programs that don't, and it allows those functions to be provided in the "official" libjpeg-turbo binaries.

Those who are concerned about maintaining strict conformance with the libjpeg v6b or v7 API can pass an argument of `-DWITH_MEM_SRCDST=0` to `cmake` prior to building libjpeg-turbo. This will restore the pre-1.3 behavior, in which `jpeg_mem_src()` and `jpeg_mem_dest()` are only included when emulating the libjpeg v8 API/ABI.

On Un*x systems, including the in-memory source/destination managers changes the dynamic library version from 62.2.0 to 62.3.0 if using libjpeg v6b API/ABI emulation and from 7.2.0 to 7.3.0 if using libjpeg v7 API/ABI emulation.

Note that, on most Un*x systems, the dynamic linker will not look for a function in a library until that function is actually used. Thus, if a program is built against libjpeg-turbo 1.3+ and uses `jpeg_mem_src()` or `jpeg_mem_dest()`, that program will not fail if run against an older version of libjpeg-turbo or against libjpeg v7- until the program actually tries to call `jpeg_mem_src()` or `jpeg_mem_dest()`. Such is not the case on Windows. If a program is built against the libjpeg-turbo 1.3+ DLL and uses `jpeg_mem_src()` or `jpeg_mem_dest()`, then it must use the libjpeg-turbo 1.3+ DLL at run time.

Both `cjpeg` and `djpeg` have been extended to allow testing the in-memory source/destination manager functions. See their respective man pages for more details.

Mathematical Compatibility

For the most part, libjpeg-turbo should produce identical output to libjpeg v6b. The one exception to this is when using the floating point DCT/IDCT, in which case the outputs of libjpeg v6b and libjpeg-turbo can differ for the following reasons:

- The SSE/SSE2 floating point DCT implementation in libjpeg-turbo is ever so slightly more accurate than the implementation in libjpeg v6b, but not by any amount perceptible to human vision (generally in the range of 0.01 to 0.08 dB gain in PNSR.)
- When not using the SIMD extensions, libjpeg-turbo uses the more accurate (and slightly faster) floating point IDCT algorithm introduced in libjpeg v8a as opposed to the algorithm used in libjpeg v6b. It should be noted, however, that this algorithm basically brings the accuracy of the floating point IDCT in line with the accuracy of the slow integer IDCT. The floating point DCT/IDCT algorithms are mainly a legacy feature, and they do not produce significantly more accuracy than the slow integer algorithms (to put numbers on this, the typical difference in PNSR between the two algorithms is less than 0.10 dB, whereas changing the quality level by 1 in the upper range of the quality scale is typically more like a 1.0 dB difference.)
- If the floating point algorithms in libjpeg-turbo are not implemented using SIMD instructions on a particular platform, then the accuracy of the floating point DCT/IDCT can depend on the compiler settings.

While libjpeg-turbo does emulate the libjpeg v8 API/ABI, under the hood it is still using the same algorithms as libjpeg v6b, so there are several specific cases in which libjpeg-turbo cannot be expected to produce the same output as libjpeg v8:

- When decompressing using scaling factors of 1/2 and 1/4, because libjpeg v8 implements those scaling algorithms differently than libjpeg v6b does, and libjpeg-turbo's SIMD extensions are based on the libjpeg v6b behavior.
- When using chrominance subsampling, because libjpeg v8 implements this with its DCT/IDCT scaling algorithms rather than with a separate downsampling/upsampling algorithm. In our testing, the subsampled/upsampled output of libjpeg v8 is less accurate than that of libjpeg v6b for this reason.
- When decompressing using a scaling factor > 1 and merged (AKA "non-fancy" or "non-smooth") chrominance upsampling, because libjpeg v8 does not support merged upsampling with scaling factors > 1 .

Performance Pitfalls

Restart Markers

The optimized Huffman decoder in libjpeg-turbo does not handle restart markers in a way that makes the rest of the libjpeg infrastructure happy, so it is necessary to use the slow Huffman decoder when decompressing a JPEG image that has restart markers. This can cause the decompression performance to drop by as much as 20%, but the performance will still be much greater than that of libjpeg. Many consumer packages, such as Photoshop, use restart markers when generating JPEG images, so images generated by those programs will experience this issue.

Fast Integer Forward DCT at High Quality Levels

The algorithm used by the SIMD-accelerated quantization function cannot produce correct results whenever the fast integer forward DCT is used along with a JPEG quality of 98-100. Thus, libjpeg-turbo must use the non-SIMD quantization function in those cases. This causes performance to drop by as much as 40%. It is therefore strongly advised that you use the slow integer forward DCT whenever encoding images with a JPEG quality of 98 or higher.

Memory Debugger Pitfalls

Valgrind and Memory Sanitizer (MSan) can generate false positives (specifically, incorrect reports of uninitialized memory accesses) when used with libjpeg-turbo's SIMD extensions. It is generally recommended that the SIMD extensions be disabled, either by passing an argument of `-DWITH_SIMD=0` to `cmake` when configuring the build or by setting the environment variable `JSIMD_FORCENONE` to 1 at run time, when testing libjpeg-turbo with Valgrind, MSan, or other memory debuggers.

Chapter 3

Background

libjpeg-turbo is a JPEG image codec that uses SIMD instructions to accelerate baseline JPEG compression and decompression on x86, x86-64, ARM, PowerPC, and MIPS systems, as well as progressive JPEG compression on x86 and x86-64 systems. On such systems, libjpeg-turbo is generally 2-6x as fast as libjpeg, all else being equal. On other types of systems, libjpeg-turbo can still outperform libjpeg by a significant amount, by virtue of its highly-optimized Huffman coding routines. In many cases, the performance of libjpeg-turbo rivals that of proprietary high-speed JPEG codecs.

libjpeg-turbo implements both the traditional libjpeg API as well as the less powerful but more straightforward TurboJPEG API. libjpeg-turbo also features colorspace extensions that allow it to compress from/decompress to 32-bit and big-endian pixel buffers (RGBX, XBGR, etc.), as well as a full-featured Java interface.

libjpeg-turbo was originally based on libjpeg/SIMD, an MMX-accelerated derivative of libjpeg v6b developed by Miyasaka Masaru. The TigerVNC and VirtualGL projects made numerous enhancements to the codec in 2009, and in early 2010, libjpeg-turbo spun off into an independent project, with the goal of making high-speed JPEG compression/decompression technology available to a broader range of users and developers.

License

libjpeg-turbo is covered by three compatible BSD-style open source licenses. Refer to LICENSE.md for a roll-up of license terms.

Building libjpeg-turbo

Refer to BUILDING.md for complete instructions.

Using libjpeg-turbo

libjpeg-turbo includes two APIs that can be used to compress and decompress JPEG images:

- **TurboJPEG API**

This API provides an easy-to-use interface for compressing and decompressing JPEG images in memory. It also provides some functionality that would not be straightforward to achieve using the underlying libjpeg API, such as generating planar YUV images and performing multiple simultaneous lossless transforms on an image. The Java interface for libjpeg-turbo is written on top of the TurboJPEG API. The TurboJPEG API is recommended for first-time users of libjpeg-turbo. Refer to [tjexample.c](#) and [TJExample.java](#) for examples of its usage and to <http://libjpeg-turbo.org/Documentation/Documentation> for API documentation.

- **libjpeg API**

This is the de facto industry-standard API for compressing and decompressing JPEG images. It is more difficult to use than the TurboJPEG API but also more powerful. The libjpeg API implementation in libjpeg-turbo is both API/ABI-compatible and mathematically compatible with libjpeg v6b. It can also optionally be configured to be API/ABI-compatible with libjpeg v7 and v8 (see below.) Refer to [cjpeg.c](#) and [djpeg.c](#) for examples of its usage and to [libjpeg.txt](#) for API documentation.

There is no significant performance advantage to either API when both are used to perform similar operations.

Colorspace Extensions

libjpeg-turbo includes extensions that allow JPEG images to be compressed directly from (and decompressed directly to) buffers that use BGR, BGRX, RGBX, XBGR, and XRGB pixel ordering. This is implemented with ten new colorspace constants:

```
JCS_EXT_RGB    /* red/green/blue */
JCS_EXT_RGBX   /* red/green/blue/x */
JCS_EXT_BGR    /* blue/green/red */
JCS_EXT_BGRX   /* blue/green/red/x */
JCS_EXT_XBGR   /* x/blue/green/red */
JCS_EXT_XRGB   /* x/red/green/blue */
JCS_EXT_RGBA   /* red/green/blue/alpha */
JCS_EXT_BGRA   /* blue/green/red/alpha */
JCS_EXT_ABGR   /* alpha/blue/green/red */
JCS_EXT_ARGB   /* alpha/red/green/blue */
```

Setting `cinfo.in_color_space` (compression) or `cinfo.out_color_space` (decompression) to one of these values will cause libjpeg-turbo to read the red, green, and blue values from (or write them to) the appropriate position in the pixel when compressing from/decompressing to an RGB buffer.

Your application can check for the existence of these extensions at compile time with:

```
#ifndef JCS_EXTENSIONS
```

At run time, attempting to use these extensions with a libjpeg implementation that does not support them will result in a "Bogus input colorspace" error. Applications can trap this error in order to test whether run-time support is available for the colorspace extensions.

When using the RGBX, BGRX, XBGR, and XRGB colorspace during decompression, the X byte is undefined, and in order to ensure the best performance, libjpeg-turbo can set that byte to whatever value it wishes. If an application expects the X byte to be used as an alpha channel, then it should specify `JCS_EXT_RGBA`, `JCS_EXT_BGRA`, `JCS_EXT_ABGR`, or `JCS_EXT_ARGB`. When these colorspace constants are used, the X byte is guaranteed to be 0xFF, which is interpreted as opaque.

Your application can check for the existence of the alpha channel colorspace extensions at compile time with:

```
#ifndef JCS_ALPHA_EXTENSIONS
```

[jcstest.c](#), located in the libjpeg-turbo source tree, demonstrates how to check for the existence of the colorspace extensions at compile time and run time.

libjpeg v7 and v8 API/ABI Emulation

With libjpeg v7 and v8, new features were added that necessitated extending the compression and decompression structures. Unfortunately, due to the exposed nature of those structures, extending them also necessitated breaking backward ABI compatibility with previous libjpeg releases. Thus, programs that were built to use libjpeg v7 or v8 did not work with libjpeg-turbo, since it is based on the libjpeg v6b code base. Although libjpeg v7 and v8 are not as widely used as v6b, enough programs (including a few Linux distros) made the switch that there was a demand to emulate the libjpeg v7 and v8 ABIs in libjpeg-turbo. It should be noted, however, that this feature was added primarily so that applications that had already been compiled to use libjpeg v7+ could take advantage of accelerated baseline JPEG encoding/decoding without recompiling. libjpeg-turbo does not claim to support all of the libjpeg v7+ features, nor to produce identical output to libjpeg v7+ in all cases (see below.)

By passing an argument of `-DWITH_JPEG7=1` or `-DWITH_JPEG8=1` to `cmake`, you can build a version of libjpeg-turbo that emulates the libjpeg v7 or v8 ABI, so that programs that are built against libjpeg v7 or v8 can be run with libjpeg-turbo. The following section describes which libjpeg v7+ features are supported and which aren't.

Support for libjpeg v7 and v8 Features

Fully supported

- **libjpeg API: IDCT scaling extensions in decompressor**
libjpeg-turbo supports IDCT scaling with scaling factors of 1/8, 1/4, 3/8, 1/2, 5/8, 3/4, 7/8, 9/8, 5/4, 11/8, 3/2, 13/8, 7/4, 15/8, and 2/1 (only 1/4 and 1/2 are SIMD-accelerated.)
- **libjpeg API: Arithmetic coding**
- **libjpeg API: In-memory source and destination managers**
See notes below.
- **cjpeg: Separate quality settings for luminance and chrominance**
Note that the libjpeg v7+ API was extended to accommodate this feature only for convenience purposes. It has always been possible to implement this feature with libjpeg v6b (see `rdswitch.c` for an example.)
- **cjpeg: 32-bit BMP support**
- **cjpeg: `-rgb` option**
- **jpegtran: Lossless cropping**
- **jpegtran: `-perfect` option**
- **jpegtran: Forcing width/height when performing lossless crop**
- **rdjpgcom: `-raw` option**
- **rdjpgcom: Locale awareness**

Not supported

NOTE: As of this writing, extensive research has been conducted into the usefulness of DCT scaling as a means of data reduction and SmartScale as a means of quality improvement. The reader is invited to peruse the research at <http://www.libjpeg-turbo.org/About/SmartScale> and draw his/her own conclusions, but it is the general belief of our project that these features have not demonstrated sufficient usefulness to justify inclusion in libjpeg-turbo.

- **libjpeg API: DCT scaling in compressor**
`cinfo.scale_num` and `cinfo.scale_denom` are silently ignored. There is no technical reason why DCT scaling could not be supported when emulating the libjpeg v7+ API/ABI, but without the SmartScale extension (see below), only scaling factors of 1/2, 8/15, 4/7, 8/13, 2/3, 8/11, 4/5, and 8/9 would be available, which is of limited usefulness.

- **libjpeg API: SmartScale**

`cinfo.block_size` is silently ignored. SmartScale is an extension to the JPEG format that allows for DCT block sizes other than 8x8. Providing support for this new format would be feasible (particularly without full acceleration.) However, until/unless the format becomes either an official industry standard or, at minimum, an accepted solution in the community, we are hesitant to implement it, as there is no sense of whether or how it might change in the future. It is our belief that SmartScale has not demonstrated sufficient usefulness as a lossless format nor as a means of quality enhancement, and thus our primary interest in providing this feature would be as a means of supporting additional DCT scaling factors.

- **libjpeg API: Fancy downsampling in compressor**

`cinfo.do_fancy_downsampling` is silently ignored. This requires the DCT scaling feature, which is not supported.

- **jpegtran: Scaling**

This requires both the DCT scaling and SmartScale features, which are not supported.

- **Lossless RGB JPEG files**

This requires the SmartScale feature, which is not supported.

What About libjpeg v9?

libjpeg v9 introduced yet another field to the JPEG compression structure (`color_transform`), thus making the ABI backward incompatible with that of libjpeg v8. This new field was introduced solely for the purpose of supporting lossless SmartScale encoding. Furthermore, there was actually no reason to extend the API in this manner, as the color transform could have just as easily been activated by way of a new JPEG colorspace constant, thus preserving backward ABI compatibility.

Our research (see link above) has shown that lossless SmartScale does not generally accomplish anything that can't already be accomplished better with existing, standard lossless formats. Therefore, at this time it is our belief that there is not sufficient technical justification for software projects to upgrade from libjpeg v8 to libjpeg v9, and thus there is not sufficient technical justification for us to emulate the libjpeg v9 ABI.

In-Memory Source/Destination Managers

By default, libjpeg-turbo 1.3 and later includes the `jpeg_mem_src()` and `jpeg_mem_dest()` functions, even when not emulating the libjpeg v8 API/ABI. Previously, it was necessary to build libjpeg-turbo from source with libjpeg v8 API/ABI emulation in order to use the in-memory source/destination managers, but several projects requested that those functions be included when emulating the libjpeg v6b API/ABI as well. This allows the use of those functions by programs that need them, without breaking ABI compatibility for programs that don't, and it allows those functions to be provided in the "official" libjpeg-turbo binaries.

Those who are concerned about maintaining strict conformance with the libjpeg v6b or v7 API can pass an argument of `-DWITH_MEM_SRCDST=0` to `cmake` prior to building libjpeg-turbo. This will restore the pre-1.3 behavior, in which `jpeg_mem_src()` and `jpeg_mem_dest()` are only included when emulating the libjpeg v8 API/ABI.

On Un*x systems, including the in-memory source/destination managers changes the dynamic library version from 62.2.0 to 62.3.0 if using libjpeg v6b API/ABI emulation and from 7.2.0 to 7.3.0 if using libjpeg v7 API/ABI emulation.

Note that, on most Un*x systems, the dynamic linker will not look for a function in a library until that function is actually used. Thus, if a program is built against libjpeg-turbo 1.3+ and uses `jpeg_mem_src()` or `jpeg_mem_dest()`, that program will not fail if run against an older version of libjpeg-turbo or against libjpeg v7- until the program actually tries to call `jpeg_mem_src()` or `jpeg_mem_dest()`. Such is not the case on Windows. If a program is built against the libjpeg-turbo 1.3+ DLL and uses `jpeg_mem_src()` or `jpeg_mem_dest()`, then it must use the libjpeg-turbo 1.3+ DLL at run time.

Both `cjpeg` and `djpeg` have been extended to allow testing the in-memory source/destination manager functions. See their respective man pages for more details.

Mathematical Compatibility

For the most part, libjpeg-turbo should produce identical output to libjpeg v6b. The one exception to this is when using the floating point DCT/IDCT, in which case the outputs of libjpeg v6b and libjpeg-turbo can differ for the following reasons:

- The SSE/SSE2 floating point DCT implementation in libjpeg-turbo is ever so slightly more accurate than the implementation in libjpeg v6b, but not by any amount perceptible to human vision (generally in the range of 0.01 to 0.08 dB gain in PNSR.)
- When not using the SIMD extensions, libjpeg-turbo uses the more accurate (and slightly faster) floating point IDCT algorithm introduced in libjpeg v8a as opposed to the algorithm used in libjpeg v6b. It should be noted, however, that this algorithm basically brings the accuracy of the floating point IDCT in line with the accuracy of the slow integer IDCT. The floating point DCT/IDCT algorithms are mainly a legacy feature, and they do not produce significantly more accuracy than the slow integer algorithms (to put numbers on this, the typical difference in PNSR between the two algorithms is less than 0.10 dB, whereas changing the quality level by 1 in the upper range of the quality scale is typically more like a 1.0 dB difference.)
- If the floating point algorithms in libjpeg-turbo are not implemented using SIMD instructions on a particular platform, then the accuracy of the floating point DCT/IDCT can depend on the compiler settings.

While libjpeg-turbo does emulate the libjpeg v8 API/ABI, under the hood it is still using the same algorithms as libjpeg v6b, so there are several specific cases in which libjpeg-turbo cannot be expected to produce the same output as libjpeg v8:

- When decompressing using scaling factors of 1/2 and 1/4, because libjpeg v8 implements those scaling algorithms differently than libjpeg v6b does, and libjpeg-turbo's SIMD extensions are based on the libjpeg v6b behavior.
- When using chrominance subsampling, because libjpeg v8 implements this with its DCT/IDCT scaling algorithms rather than with a separate downsampling/upsampling algorithm. In our testing, the subsampled/upsampled output of libjpeg v8 is less accurate than that of libjpeg v6b for this reason.
- When decompressing using a scaling factor > 1 and merged (AKA "non-fancy" or "non-smooth") chrominance upsampling, because libjpeg v8 does not support merged upsampling with scaling factors > 1 .

Performance Pitfalls

Restart Markers

The optimized Huffman decoder in libjpeg-turbo does not handle restart markers in a way that makes the rest of the libjpeg infrastructure happy, so it is necessary to use the slow Huffman decoder when decompressing a JPEG image that has restart markers. This can cause the decompression performance to drop by as much as 20%, but the performance will still be much greater than that of libjpeg. Many consumer packages, such as Photoshop, use restart markers when generating JPEG images, so images generated by those programs will experience this issue.

Fast Integer Forward DCT at High Quality Levels

The algorithm used by the SIMD-accelerated quantization function cannot produce correct results whenever the fast integer forward DCT is used along with a JPEG quality of 98-100. Thus, libjpeg-turbo must use the non-SIMD quantization function in those cases. This causes performance to drop by as much as 40%. It is therefore strongly advised that you use the slow integer forward DCT whenever encoding images with a JPEG quality of 98 or higher.

Memory Debugger Pitfalls

Valgrind and Memory Sanitizer (MSan) can generate false positives (specifically, incorrect reports of uninitialized memory accesses) when used with libjpeg-turbo's SIMD extensions. It is generally recommended that the SIMD extensions be disabled, either by passing an argument of `-DWITH_SIMD=0` to `cmake` when configuring the build or by setting the environment variable `JSIMD_FORCENONE` to 1 at run time, when testing libjpeg-turbo with Valgrind, MSan, or other memory debuggers.

Chapter 4

Background

libjpeg-turbo is a JPEG image codec that uses SIMD instructions to accelerate baseline JPEG compression and decompression on x86, x86-64, ARM, PowerPC, and MIPS systems, as well as progressive JPEG compression on x86 and x86-64 systems. On such systems, libjpeg-turbo is generally 2-6x as fast as libjpeg, all else being equal. On other types of systems, libjpeg-turbo can still outperform libjpeg by a significant amount, by virtue of its highly-optimized Huffman coding routines. In many cases, the performance of libjpeg-turbo rivals that of proprietary high-speed JPEG codecs.

libjpeg-turbo implements both the traditional libjpeg API as well as the less powerful but more straightforward TurboJPEG API. libjpeg-turbo also features colorspace extensions that allow it to compress from/decompress to 32-bit and big-endian pixel buffers (RGBX, XBGR, etc.), as well as a full-featured Java interface.

libjpeg-turbo was originally based on libjpeg/SIMD, an MMX-accelerated derivative of libjpeg v6b developed by Miyasaka Masaru. The TigerVNC and VirtualGL projects made numerous enhancements to the codec in 2009, and in early 2010, libjpeg-turbo spun off into an independent project, with the goal of making high-speed JPEG compression/decompression technology available to a broader range of users and developers.

License

libjpeg-turbo is covered by three compatible BSD-style open source licenses. Refer to LICENSE.md for a roll-up of license terms.

Building libjpeg-turbo

Refer to BUILDING.md for complete instructions.

Using libjpeg-turbo

libjpeg-turbo includes two APIs that can be used to compress and decompress JPEG images:

- **TurboJPEG API**

This API provides an easy-to-use interface for compressing and decompressing JPEG images in memory. It also provides some functionality that would not be straightforward to achieve using the underlying libjpeg API, such as generating planar YUV images and performing multiple simultaneous lossless transforms on an image. The Java interface for libjpeg-turbo is written on top of the TurboJPEG API. The TurboJPEG API is recommended for first-time users of libjpeg-turbo. Refer to [tjexample.c](#) and [TJExample.java](#) for examples of its usage and to <http://libjpeg-turbo.org/Documentation/Documentation> for API documentation.

- **libjpeg API**

This is the de facto industry-standard API for compressing and decompressing JPEG images. It is more difficult to use than the TurboJPEG API but also more powerful. The libjpeg API implementation in libjpeg-turbo is both API/ABI-compatible and mathematically compatible with libjpeg v6b. It can also optionally be configured to be API/ABI-compatible with libjpeg v7 and v8 (see below.) Refer to [cjpeg.c](#) and [djpeg.c](#) for examples of its usage and to [libjpeg.txt](#) for API documentation.

There is no significant performance advantage to either API when both are used to perform similar operations.

Colorspace Extensions

libjpeg-turbo includes extensions that allow JPEG images to be compressed directly from (and decompressed directly to) buffers that use BGR, BGRX, RGBX, XBGR, and XRGB pixel ordering. This is implemented with ten new colorspace constants:

```
JCS_EXT_RGB    /* red/green/blue */
JCS_EXT_RGBX   /* red/green/blue/x */
JCS_EXT_BGR    /* blue/green/red */
JCS_EXT_BGRX   /* blue/green/red/x */
JCS_EXT_XBGR   /* x/blue/green/red */
JCS_EXT_XRGB   /* x/red/green/blue */
JCS_EXT_RGBA   /* red/green/blue/alpha */
JCS_EXT_BGRA   /* blue/green/red/alpha */
JCS_EXT_ABGR   /* alpha/blue/green/red */
JCS_EXT_ARGB   /* alpha/red/green/blue */
```

Setting `cinfo.in_color_space` (compression) or `cinfo.out_color_space` (decompression) to one of these values will cause libjpeg-turbo to read the red, green, and blue values from (or write them to) the appropriate position in the pixel when compressing from/decompressing to an RGB buffer.

Your application can check for the existence of these extensions at compile time with:

```
#ifndef JCS_EXTENSIONS
```

At run time, attempting to use these extensions with a libjpeg implementation that does not support them will result in a "Bogus input colorspace" error. Applications can trap this error in order to test whether run-time support is available for the colorspace extensions.

When using the RGBX, BGRX, XBGR, and XRGB colorspace during decompression, the X byte is undefined, and in order to ensure the best performance, libjpeg-turbo can set that byte to whatever value it wishes. If an application expects the X byte to be used as an alpha channel, then it should specify `JCS_EXT_RGBA`, `JCS_EXT_BGRA`, `JCS_EXT_ABGR`, or `JCS_EXT_ARGB`. When these colorspace constants are used, the X byte is guaranteed to be 0xFF, which is interpreted as opaque.

Your application can check for the existence of the alpha channel colorspace extensions at compile time with:

```
#ifndef JCS_ALPHA_EXTENSIONS
```

[jcstest.c](#), located in the libjpeg-turbo source tree, demonstrates how to check for the existence of the colorspace extensions at compile time and run time.

libjpeg v7 and v8 API/ABI Emulation

With libjpeg v7 and v8, new features were added that necessitated extending the compression and decompression structures. Unfortunately, due to the exposed nature of those structures, extending them also necessitated breaking backward ABI compatibility with previous libjpeg releases. Thus, programs that were built to use libjpeg v7 or v8 did not work with libjpeg-turbo, since it is based on the libjpeg v6b code base. Although libjpeg v7 and v8 are not as widely used as v6b, enough programs (including a few Linux distros) made the switch that there was a demand to emulate the libjpeg v7 and v8 ABIs in libjpeg-turbo. It should be noted, however, that this feature was added primarily so that applications that had already been compiled to use libjpeg v7+ could take advantage of accelerated baseline JPEG encoding/decoding without recompiling. libjpeg-turbo does not claim to support all of the libjpeg v7+ features, nor to produce identical output to libjpeg v7+ in all cases (see below.)

By passing an argument of `-DWITH_JPEG7=1` or `-DWITH_JPEG8=1` to `cmake`, you can build a version of libjpeg-turbo that emulates the libjpeg v7 or v8 ABI, so that programs that are built against libjpeg v7 or v8 can be run with libjpeg-turbo. The following section describes which libjpeg v7+ features are supported and which aren't.

Support for libjpeg v7 and v8 Features

Fully supported

- **libjpeg API: IDCT scaling extensions in decompressor**
libjpeg-turbo supports IDCT scaling with scaling factors of 1/8, 1/4, 3/8, 1/2, 5/8, 3/4, 7/8, 9/8, 5/4, 11/8, 3/2, 13/8, 7/4, 15/8, and 2/1 (only 1/4 and 1/2 are SIMD-accelerated.)
- **libjpeg API: Arithmetic coding**
- **libjpeg API: In-memory source and destination managers**
See notes below.
- **cjpeg: Separate quality settings for luminance and chrominance**
Note that the libjpeg v7+ API was extended to accommodate this feature only for convenience purposes. It has always been possible to implement this feature with libjpeg v6b (see `rdswitch.c` for an example.)
- **cjpeg: 32-bit BMP support**
- **cjpeg: `-rgb` option**
- **jpegtran: Lossless cropping**
- **jpegtran: `-perfect` option**
- **jpegtran: Forcing width/height when performing lossless crop**
- **rdjpgcom: `-raw` option**
- **rdjpgcom: Locale awareness**

Not supported

NOTE: As of this writing, extensive research has been conducted into the usefulness of DCT scaling as a means of data reduction and SmartScale as a means of quality improvement. The reader is invited to peruse the research at <http://www.libjpeg-turbo.org/About/SmartScale> and draw his/her own conclusions, but it is the general belief of our project that these features have not demonstrated sufficient usefulness to justify inclusion in libjpeg-turbo.

- **libjpeg API: DCT scaling in compressor**
`cinfo.scale_num` and `cinfo.scale_denom` are silently ignored. There is no technical reason why DCT scaling could not be supported when emulating the libjpeg v7+ API/ABI, but without the SmartScale extension (see below), only scaling factors of 1/2, 8/15, 4/7, 8/13, 2/3, 8/11, 4/5, and 8/9 would be available, which is of limited usefulness.

- **libjpeg API: SmartScale**

`cinfo.block_size` is silently ignored. SmartScale is an extension to the JPEG format that allows for DCT block sizes other than 8x8. Providing support for this new format would be feasible (particularly without full acceleration.) However, until/unless the format becomes either an official industry standard or, at minimum, an accepted solution in the community, we are hesitant to implement it, as there is no sense of whether or how it might change in the future. It is our belief that SmartScale has not demonstrated sufficient usefulness as a lossless format nor as a means of quality enhancement, and thus our primary interest in providing this feature would be as a means of supporting additional DCT scaling factors.

- **libjpeg API: Fancy downsampling in compressor**

`cinfo.do_fancy_downsampling` is silently ignored. This requires the DCT scaling feature, which is not supported.

- **jpegtran: Scaling**

This requires both the DCT scaling and SmartScale features, which are not supported.

- **Lossless RGB JPEG files**

This requires the SmartScale feature, which is not supported.

What About libjpeg v9?

libjpeg v9 introduced yet another field to the JPEG compression structure (`color_transform`), thus making the ABI backward incompatible with that of libjpeg v8. This new field was introduced solely for the purpose of supporting lossless SmartScale encoding. Furthermore, there was actually no reason to extend the API in this manner, as the color transform could have just as easily been activated by way of a new JPEG colorspace constant, thus preserving backward ABI compatibility.

Our research (see link above) has shown that lossless SmartScale does not generally accomplish anything that can't already be accomplished better with existing, standard lossless formats. Therefore, at this time it is our belief that there is not sufficient technical justification for software projects to upgrade from libjpeg v8 to libjpeg v9, and thus there is not sufficient technical justification for us to emulate the libjpeg v9 ABI.

In-Memory Source/Destination Managers

By default, libjpeg-turbo 1.3 and later includes the `jpeg_mem_src()` and `jpeg_mem_dest()` functions, even when not emulating the libjpeg v8 API/ABI. Previously, it was necessary to build libjpeg-turbo from source with libjpeg v8 API/ABI emulation in order to use the in-memory source/destination managers, but several projects requested that those functions be included when emulating the libjpeg v6b API/ABI as well. This allows the use of those functions by programs that need them, without breaking ABI compatibility for programs that don't, and it allows those functions to be provided in the "official" libjpeg-turbo binaries.

Those who are concerned about maintaining strict conformance with the libjpeg v6b or v7 API can pass an argument of `-DWITH_MEM_SRCDST=0` to `cmake` prior to building libjpeg-turbo. This will restore the pre-1.3 behavior, in which `jpeg_mem_src()` and `jpeg_mem_dest()` are only included when emulating the libjpeg v8 API/ABI.

On Un*x systems, including the in-memory source/destination managers changes the dynamic library version from 62.2.0 to 62.3.0 if using libjpeg v6b API/ABI emulation and from 7.2.0 to 7.3.0 if using libjpeg v7 API/ABI emulation.

Note that, on most Un*x systems, the dynamic linker will not look for a function in a library until that function is actually used. Thus, if a program is built against libjpeg-turbo 1.3+ and uses `jpeg_mem_src()` or `jpeg_mem_dest()`, that program will not fail if run against an older version of libjpeg-turbo or against libjpeg v7- until the program actually tries to call `jpeg_mem_src()` or `jpeg_mem_dest()`. Such is not the case on Windows. If a program is built against the libjpeg-turbo 1.3+ DLL and uses `jpeg_mem_src()` or `jpeg_mem_dest()`, then it must use the libjpeg-turbo 1.3+ DLL at run time.

Both `cjpeg` and `djpeg` have been extended to allow testing the in-memory source/destination manager functions. See their respective man pages for more details.

Mathematical Compatibility

For the most part, libjpeg-turbo should produce identical output to libjpeg v6b. The one exception to this is when using the floating point DCT/IDCT, in which case the outputs of libjpeg v6b and libjpeg-turbo can differ for the following reasons:

- The SSE/SSE2 floating point DCT implementation in libjpeg-turbo is ever so slightly more accurate than the implementation in libjpeg v6b, but not by any amount perceptible to human vision (generally in the range of 0.01 to 0.08 dB gain in PNSR.)
- When not using the SIMD extensions, libjpeg-turbo uses the more accurate (and slightly faster) floating point IDCT algorithm introduced in libjpeg v8a as opposed to the algorithm used in libjpeg v6b. It should be noted, however, that this algorithm basically brings the accuracy of the floating point IDCT in line with the accuracy of the slow integer IDCT. The floating point DCT/IDCT algorithms are mainly a legacy feature, and they do not produce significantly more accuracy than the slow integer algorithms (to put numbers on this, the typical difference in PNSR between the two algorithms is less than 0.10 dB, whereas changing the quality level by 1 in the upper range of the quality scale is typically more like a 1.0 dB difference.)
- If the floating point algorithms in libjpeg-turbo are not implemented using SIMD instructions on a particular platform, then the accuracy of the floating point DCT/IDCT can depend on the compiler settings.

While libjpeg-turbo does emulate the libjpeg v8 API/ABI, under the hood it is still using the same algorithms as libjpeg v6b, so there are several specific cases in which libjpeg-turbo cannot be expected to produce the same output as libjpeg v8:

- When decompressing using scaling factors of 1/2 and 1/4, because libjpeg v8 implements those scaling algorithms differently than libjpeg v6b does, and libjpeg-turbo's SIMD extensions are based on the libjpeg v6b behavior.
- When using chrominance subsampling, because libjpeg v8 implements this with its DCT/IDCT scaling algorithms rather than with a separate downsampling/upsampling algorithm. In our testing, the subsampled/upsampled output of libjpeg v8 is less accurate than that of libjpeg v6b for this reason.
- When decompressing using a scaling factor > 1 and merged (AKA "non-fancy" or "non-smooth") chrominance upsampling, because libjpeg v8 does not support merged upsampling with scaling factors > 1 .

Performance Pitfalls

Restart Markers

The optimized Huffman decoder in libjpeg-turbo does not handle restart markers in a way that makes the rest of the libjpeg infrastructure happy, so it is necessary to use the slow Huffman decoder when decompressing a JPEG image that has restart markers. This can cause the decompression performance to drop by as much as 20%, but the performance will still be much greater than that of libjpeg. Many consumer packages, such as Photoshop, use restart markers when generating JPEG images, so images generated by those programs will experience this issue.

Fast Integer Forward DCT at High Quality Levels

The algorithm used by the SIMD-accelerated quantization function cannot produce correct results whenever the fast integer forward DCT is used along with a JPEG quality of 98-100. Thus, libjpeg-turbo must use the non-SIMD quantization function in those cases. This causes performance to drop by as much as 40%. It is therefore strongly advised that you use the slow integer forward DCT whenever encoding images with a JPEG quality of 98 or higher.

Memory Debugger Pitfalls

Valgrind and Memory Sanitizer (MSan) can generate false positives (specifically, incorrect reports of uninitialized memory accesses) when used with libjpeg-turbo's SIMD extensions. It is generally recommended that the SIMD extensions be disabled, either by passing an argument of `-DWITH_SIMD=0` to `cmake` when configuring the build or by setting the environment variable `JSIMD_FORCENONE` to 1 at run time, when testing libjpeg-turbo with Valgrind, MSan, or other memory debuggers.

Chapter 5

Background

libjpeg-turbo is a JPEG image codec that uses SIMD instructions to accelerate baseline JPEG compression and decompression on x86, x86-64, ARM, PowerPC, and MIPS systems, as well as progressive JPEG compression on x86 and x86-64 systems. On such systems, libjpeg-turbo is generally 2-6x as fast as libjpeg, all else being equal. On other types of systems, libjpeg-turbo can still outperform libjpeg by a significant amount, by virtue of its highly-optimized Huffman coding routines. In many cases, the performance of libjpeg-turbo rivals that of proprietary high-speed JPEG codecs.

libjpeg-turbo implements both the traditional libjpeg API as well as the less powerful but more straightforward TurboJPEG API. libjpeg-turbo also features colorspace extensions that allow it to compress from/decompress to 32-bit and big-endian pixel buffers (RGBX, XBGR, etc.), as well as a full-featured Java interface.

libjpeg-turbo was originally based on libjpeg/SIMD, an MMX-accelerated derivative of libjpeg v6b developed by Miyasaka Masaru. The TigerVNC and VirtualGL projects made numerous enhancements to the codec in 2009, and in early 2010, libjpeg-turbo spun off into an independent project, with the goal of making high-speed JPEG compression/decompression technology available to a broader range of users and developers.

License

libjpeg-turbo is covered by three compatible BSD-style open source licenses. Refer to LICENSE.md for a roll-up of license terms.

Building libjpeg-turbo

Refer to BUILDING.md for complete instructions.

Using libjpeg-turbo

libjpeg-turbo includes two APIs that can be used to compress and decompress JPEG images:

- **TurboJPEG API**

This API provides an easy-to-use interface for compressing and decompressing JPEG images in memory. It also provides some functionality that would not be straightforward to achieve using the underlying libjpeg API, such as generating planar YUV images and performing multiple simultaneous lossless transforms on an image. The Java interface for libjpeg-turbo is written on top of the TurboJPEG API. The TurboJPEG API is recommended for first-time users of libjpeg-turbo. Refer to [tjexample.c](#) and [TJExample.java](#) for examples of its usage and to <http://libjpeg-turbo.org/Documentation/Documentation> for API documentation.

- **libjpeg API**

This is the de facto industry-standard API for compressing and decompressing JPEG images. It is more difficult to use than the TurboJPEG API but also more powerful. The libjpeg API implementation in libjpeg-turbo is both API/ABI-compatible and mathematically compatible with libjpeg v6b. It can also optionally be configured to be API/ABI-compatible with libjpeg v7 and v8 (see below.) Refer to [cjpeg.c](#) and [djpeg.c](#) for examples of its usage and to [libjpeg.txt](#) for API documentation.

There is no significant performance advantage to either API when both are used to perform similar operations.

Colorspace Extensions

libjpeg-turbo includes extensions that allow JPEG images to be compressed directly from (and decompressed directly to) buffers that use BGR, BGRX, RGBX, XBGR, and XRGB pixel ordering. This is implemented with ten new colorspace constants:

```
JCS_EXT_RGB    /* red/green/blue */
JCS_EXT_RGBX   /* red/green/blue/x */
JCS_EXT_BGR    /* blue/green/red */
JCS_EXT_BGRX   /* blue/green/red/x */
JCS_EXT_XBGR   /* x/blue/green/red */
JCS_EXT_XRGB   /* x/red/green/blue */
JCS_EXT_RGBA   /* red/green/blue/alpha */
JCS_EXT_BGRA   /* blue/green/red/alpha */
JCS_EXT_ABGR   /* alpha/blue/green/red */
JCS_EXT_ARGB   /* alpha/red/green/blue */
```

Setting `cinfo.in_color_space` (compression) or `cinfo.out_color_space` (decompression) to one of these values will cause libjpeg-turbo to read the red, green, and blue values from (or write them to) the appropriate position in the pixel when compressing from/decompressing to an RGB buffer.

Your application can check for the existence of these extensions at compile time with:

```
#ifndef JCS_EXTENSIONS
```

At run time, attempting to use these extensions with a libjpeg implementation that does not support them will result in a "Bogus input colorspace" error. Applications can trap this error in order to test whether run-time support is available for the colorspace extensions.

When using the RGBX, BGRX, XBGR, and XRGB colorspace during decompression, the X byte is undefined, and in order to ensure the best performance, libjpeg-turbo can set that byte to whatever value it wishes. If an application expects the X byte to be used as an alpha channel, then it should specify `JCS_EXT_RGBA`, `JCS_EXT_BGRA`, `JCS_EXT_ABGR`, or `JCS_EXT_ARGB`. When these colorspace constants are used, the X byte is guaranteed to be 0xFF, which is interpreted as opaque.

Your application can check for the existence of the alpha channel colorspace extensions at compile time with:

```
#ifndef JCS_ALPHA_EXTENSIONS
```

[jcstest.c](#), located in the libjpeg-turbo source tree, demonstrates how to check for the existence of the colorspace extensions at compile time and run time.

libjpeg v7 and v8 API/ABI Emulation

With libjpeg v7 and v8, new features were added that necessitated extending the compression and decompression structures. Unfortunately, due to the exposed nature of those structures, extending them also necessitated breaking backward ABI compatibility with previous libjpeg releases. Thus, programs that were built to use libjpeg v7 or v8 did not work with libjpeg-turbo, since it is based on the libjpeg v6b code base. Although libjpeg v7 and v8 are not as widely used as v6b, enough programs (including a few Linux distros) made the switch that there was a demand to emulate the libjpeg v7 and v8 ABIs in libjpeg-turbo. It should be noted, however, that this feature was added primarily so that applications that had already been compiled to use libjpeg v7+ could take advantage of accelerated baseline JPEG encoding/decoding without recompiling. libjpeg-turbo does not claim to support all of the libjpeg v7+ features, nor to produce identical output to libjpeg v7+ in all cases (see below.)

By passing an argument of `-DWITH_JPEG7=1` or `-DWITH_JPEG8=1` to `cmake`, you can build a version of libjpeg-turbo that emulates the libjpeg v7 or v8 ABI, so that programs that are built against libjpeg v7 or v8 can be run with libjpeg-turbo. The following section describes which libjpeg v7+ features are supported and which aren't.

Support for libjpeg v7 and v8 Features

Fully supported

- **libjpeg API: IDCT scaling extensions in decompressor**
libjpeg-turbo supports IDCT scaling with scaling factors of 1/8, 1/4, 3/8, 1/2, 5/8, 3/4, 7/8, 9/8, 5/4, 11/8, 3/2, 13/8, 7/4, 15/8, and 2/1 (only 1/4 and 1/2 are SIMD-accelerated.)
- **libjpeg API: Arithmetic coding**
- **libjpeg API: In-memory source and destination managers**
See notes below.
- **cjpeg: Separate quality settings for luminance and chrominance**
Note that the libjpeg v7+ API was extended to accommodate this feature only for convenience purposes. It has always been possible to implement this feature with libjpeg v6b (see `rdswitch.c` for an example.)
- **cjpeg: 32-bit BMP support**
- **cjpeg: `-rgb` option**
- **jpegtran: Lossless cropping**
- **jpegtran: `-perfect` option**
- **jpegtran: Forcing width/height when performing lossless crop**
- **rdjpgcom: `-raw` option**
- **rdjpgcom: Locale awareness**

Not supported

NOTE: As of this writing, extensive research has been conducted into the usefulness of DCT scaling as a means of data reduction and SmartScale as a means of quality improvement. The reader is invited to peruse the research at <http://www.libjpeg-turbo.org/About/SmartScale> and draw his/her own conclusions, but it is the general belief of our project that these features have not demonstrated sufficient usefulness to justify inclusion in libjpeg-turbo.

- **libjpeg API: DCT scaling in compressor**
`cinfo.scale_num` and `cinfo.scale_denom` are silently ignored. There is no technical reason why DCT scaling could not be supported when emulating the libjpeg v7+ API/ABI, but without the SmartScale extension (see below), only scaling factors of 1/2, 8/15, 4/7, 8/13, 2/3, 8/11, 4/5, and 8/9 would be available, which is of limited usefulness.

- **libjpeg API: SmartScale**

`cinfo.block_size` is silently ignored. SmartScale is an extension to the JPEG format that allows for DCT block sizes other than 8x8. Providing support for this new format would be feasible (particularly without full acceleration.) However, until/unless the format becomes either an official industry standard or, at minimum, an accepted solution in the community, we are hesitant to implement it, as there is no sense of whether or how it might change in the future. It is our belief that SmartScale has not demonstrated sufficient usefulness as a lossless format nor as a means of quality enhancement, and thus our primary interest in providing this feature would be as a means of supporting additional DCT scaling factors.

- **libjpeg API: Fancy downsampling in compressor**

`cinfo.do_fancy_downsampling` is silently ignored. This requires the DCT scaling feature, which is not supported.

- **jpegtran: Scaling**

This requires both the DCT scaling and SmartScale features, which are not supported.

- **Lossless RGB JPEG files**

This requires the SmartScale feature, which is not supported.

What About libjpeg v9?

libjpeg v9 introduced yet another field to the JPEG compression structure (`color_transform`), thus making the ABI backward incompatible with that of libjpeg v8. This new field was introduced solely for the purpose of supporting lossless SmartScale encoding. Furthermore, there was actually no reason to extend the API in this manner, as the color transform could have just as easily been activated by way of a new JPEG colorspace constant, thus preserving backward ABI compatibility.

Our research (see link above) has shown that lossless SmartScale does not generally accomplish anything that can't already be accomplished better with existing, standard lossless formats. Therefore, at this time it is our belief that there is not sufficient technical justification for software projects to upgrade from libjpeg v8 to libjpeg v9, and thus there is not sufficient technical justification for us to emulate the libjpeg v9 ABI.

In-Memory Source/Destination Managers

By default, libjpeg-turbo 1.3 and later includes the `jpeg_mem_src()` and `jpeg_mem_dest()` functions, even when not emulating the libjpeg v8 API/ABI. Previously, it was necessary to build libjpeg-turbo from source with libjpeg v8 API/ABI emulation in order to use the in-memory source/destination managers, but several projects requested that those functions be included when emulating the libjpeg v6b API/ABI as well. This allows the use of those functions by programs that need them, without breaking ABI compatibility for programs that don't, and it allows those functions to be provided in the "official" libjpeg-turbo binaries.

Those who are concerned about maintaining strict conformance with the libjpeg v6b or v7 API can pass an argument of `-DWITH_MEM_SRCDST=0` to `cmake` prior to building libjpeg-turbo. This will restore the pre-1.3 behavior, in which `jpeg_mem_src()` and `jpeg_mem_dest()` are only included when emulating the libjpeg v8 API/ABI.

On Unix systems, including the in-memory source/destination managers changes the dynamic library version from 62.2.0 to 62.3.0 if using libjpeg v6b API/ABI emulation and from 7.2.0 to 7.3.0 if using libjpeg v7 API/ABI emulation.

Note that, on most Unix systems, the dynamic linker will not look for a function in a library until that function is actually used. Thus, if a program is built against libjpeg-turbo 1.3+ and uses `jpeg_mem_src()` or `jpeg_mem_dest()`, that program will not fail if run against an older version of libjpeg-turbo or against libjpeg v7- until the program actually tries to call `jpeg_mem_src()` or `jpeg_mem_dest()`. Such is not the case on Windows. If a program is built against the libjpeg-turbo 1.3+ DLL and uses `jpeg_mem_src()` or `jpeg_mem_dest()`, then it must use the libjpeg-turbo 1.3+ DLL at run time.

Both `cjpeg` and `djpeg` have been extended to allow testing the in-memory source/destination manager functions. See their respective man pages for more details.

Mathematical Compatibility

For the most part, libjpeg-turbo should produce identical output to libjpeg v6b. The one exception to this is when using the floating point DCT/IDCT, in which case the outputs of libjpeg v6b and libjpeg-turbo can differ for the following reasons:

- The SSE/SSE2 floating point DCT implementation in libjpeg-turbo is ever so slightly more accurate than the implementation in libjpeg v6b, but not by any amount perceptible to human vision (generally in the range of 0.01 to 0.08 dB gain in PNSR.)
- When not using the SIMD extensions, libjpeg-turbo uses the more accurate (and slightly faster) floating point IDCT algorithm introduced in libjpeg v8a as opposed to the algorithm used in libjpeg v6b. It should be noted, however, that this algorithm basically brings the accuracy of the floating point IDCT in line with the accuracy of the slow integer IDCT. The floating point DCT/IDCT algorithms are mainly a legacy feature, and they do not produce significantly more accuracy than the slow integer algorithms (to put numbers on this, the typical difference in PNSR between the two algorithms is less than 0.10 dB, whereas changing the quality level by 1 in the upper range of the quality scale is typically more like a 1.0 dB difference.)
- If the floating point algorithms in libjpeg-turbo are not implemented using SIMD instructions on a particular platform, then the accuracy of the floating point DCT/IDCT can depend on the compiler settings.

While libjpeg-turbo does emulate the libjpeg v8 API/ABI, under the hood it is still using the same algorithms as libjpeg v6b, so there are several specific cases in which libjpeg-turbo cannot be expected to produce the same output as libjpeg v8:

- When decompressing using scaling factors of 1/2 and 1/4, because libjpeg v8 implements those scaling algorithms differently than libjpeg v6b does, and libjpeg-turbo's SIMD extensions are based on the libjpeg v6b behavior.
- When using chrominance subsampling, because libjpeg v8 implements this with its DCT/IDCT scaling algorithms rather than with a separate downsampling/upsampling algorithm. In our testing, the subsampled/upsampled output of libjpeg v8 is less accurate than that of libjpeg v6b for this reason.
- When decompressing using a scaling factor > 1 and merged (AKA "non-fancy" or "non-smooth") chrominance upsampling, because libjpeg v8 does not support merged upsampling with scaling factors > 1 .

Performance Pitfalls

Restart Markers

The optimized Huffman decoder in libjpeg-turbo does not handle restart markers in a way that makes the rest of the libjpeg infrastructure happy, so it is necessary to use the slow Huffman decoder when decompressing a JPEG image that has restart markers. This can cause the decompression performance to drop by as much as 20%, but the performance will still be much greater than that of libjpeg. Many consumer packages, such as Photoshop, use restart markers when generating JPEG images, so images generated by those programs will experience this issue.

Fast Integer Forward DCT at High Quality Levels

The algorithm used by the SIMD-accelerated quantization function cannot produce correct results whenever the fast integer forward DCT is used along with a JPEG quality of 98-100. Thus, libjpeg-turbo must use the non-SIMD quantization function in those cases. This causes performance to drop by as much as 40%. It is therefore strongly advised that you use the slow integer forward DCT whenever encoding images with a JPEG quality of 98 or higher.

Memory Debugger Pitfalls

Valgrind and Memory Sanitizer (MSan) can generate false positives (specifically, incorrect reports of uninitialized memory accesses) when used with libjpeg-turbo's SIMD extensions. It is generally recommended that the SIMD extensions be disabled, either by passing an argument of `-DWITH_SIMD=0` to `cmake` when configuring the build or by setting the environment variable `JSIMD_FORCENONE` to 1 at run time, when testing libjpeg-turbo with Valgrind, MSan, or other memory debuggers.

Chapter 6

Class Index

6.1 Class List

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

AccelerationTag	31
APCImageType	31
CompassTag	32
eSPCtrl_RectLogData	32
GyroTag	37
packet_s	37
PointCloudInfo	37
tagAPC_STREAM_INFO	38
tagDEVINFORMATION	38
tagDEVSEL	38
tagKEEP_DATA_CTRL	38
tagZDTableInfo	39

Chapter 7

File Index

7.1 File List

Here is a list of all documented files with brief descriptions:

eSPDI/ eSPDI.h	
Functions definitions	41
eSPDI/ eSPDI_def.h	
Error/data type definitions	131
eSPDI/ eSPDI_version.h	??

Chapter 8

Class Documentation

8.1 AccelerationTag Struct Reference

Public Attributes

- short **x**
- short **y**
- short **z**

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

- [eSPDI/eSPDI_def.h](#)

8.2 APCImageType Struct Reference

Public Types

- enum **Value** {
 IMAGE_UNKNOWN = -1, **COLOR_YUY2** = 0, **COLOR_RGB24**, **COLOR_MJPEG**,
 COLOR_UYVY, **DEPTH_8BITS** = 100, **DEPTH_8BITS_0x80**, **DEPTH_11BITS**,
 DEPTH_14BITS }

Static Public Member Functions

- static bool **IsImageColor** (APCImageType::Value type)
- static bool **IsImageDepth** (APCImageType::Value type)
- static APCImageType::Value **DepthDataTypeToDepthImageType** (WORD dataType)

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

- [eSPDI/eSPDI_def.h](#)

8.3 CompassTag Struct Reference

Public Attributes

- short **x**
- short **y**
- short **z**

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

- [eSPDI/eSPDI_def.h](#)

8.4 eSPCtrl_RectLogData Struct Reference

Public Attributes

- ```

union {
 unsigned char uByteArray [1024]
 struct {
 unsigned short InImgWidth
 unsigned short InImgHeight
 unsigned short OutImgWidth
 unsigned short OutImgHeight
 int RECT_ScaleEnable
 int RECT_CropEnable
 unsigned short RECT_ScaleWidth
 unsigned short RECT_ScaleHeight
 float CamMat1 [9]
 float CamDist1 [8]
 float CamMat2 [9]
 float CamDist2 [8]
 float RotaMat [9]
 float TranMat [3]
 float LRotaMat [9]
 float RRotaMat [9]
 float NewCamMat1 [12]
 float NewCamMat2 [12]
 unsigned short RECT_Crop_Row_BG
 unsigned short RECT_Crop_Row_ED
 unsigned short RECT_Crop_Col_BG_L
 unsigned short RECT_Crop_Col_ED_L
 unsigned char RECT_Scale_Col_M
 unsigned char RECT_Scale_Col_N
 unsigned char RECT_Scale_Row_M
 unsigned char RECT_Scale_Row_N
 float RECT_AvgErr
 unsigned short nLineBuffers
 float ReProjectMat [16]
 }
};

```

### 8.4.1 Member Data Documentation

#### 8.4.1.1 CamDist1

```
float eSPCtrl_RectLogData::CamDist1[8]
```

Left Camera Distortion Matrix k1, k2, p1, p2, k3, k4, k5, k6 k1~k6 : radial distort ; p1,p2 : tangential distort

#### 8.4.1.2 CamDist2

```
float eSPCtrl_RectLogData::CamDist2[8]
```

Right Camera Distortion Matrix k1, k2, p1, p2, k3, k4, k5, k6 k1~k6 : radial distort ; p1,p2 : tangential distort

#### 8.4.1.3 CamMat1

```
float eSPCtrl_RectLogData::CamMat1[9]
```

Left Camera Matrix fx, 0, cx, 0, fy, cy, 0, 0, 1 fx,fy : focus ; cx,cy : principle point

#### 8.4.1.4 CamMat2

```
float eSPCtrl_RectLogData::CamMat2[9]
```

Right Camera Matrix fx, 0, cx, 0, fy, cy, 0, 0, 1 fx,fy : focus ; cx,cy : principle point

#### 8.4.1.5 InImgHeight

```
unsigned short eSPCtrl_RectLogData::InImgHeight
```

Input image height

#### 8.4.1.6 InImgWidth

```
unsigned short eSPCtrl_RectLogData::InImgWidth
```

Input image width(SideBySide image)

#### 8.4.1.7 LRotaMat

```
float eSPCtrl_RectLogData::LRotaMat[9]
```

3x3 rectification transform (rotation matrix) for the left camera.  $\begin{bmatrix} [0] & [1] & [2] \\ [3] & [4] & [5] \end{bmatrix} \cdot \begin{bmatrix} |Xcl| \\ |Ycl| \end{bmatrix} \Rightarrow cl = \text{left camera coordinate}$   $\begin{bmatrix} [6] & [7] & [8] \end{bmatrix} \cdot \begin{bmatrix} |Zcl| \end{bmatrix}$

#### 8.4.1.8 NewCamMat1

```
float eSPCtrl_RectLogData::NewCamMat1[12]
```

3x4 projection matrix in the (rectified) coordinate systems for the left camera.  $fx'$   $0$   $cx'$   $0$   $0$   $fy'$   $cy'$   $0$   $0$   $0$   $1$   $0$   $fx',fy'$  : rectified focus ;  $cx'$ ,  $cy'$  : rectified principle point

#### 8.4.1.9 NewCamMat2

```
float eSPCtrl_RectLogData::NewCamMat2[12]
```

3x4 projection matrix in the (rectified) coordinate systems for the right camera.  $fx'$   $0$   $cx'$   $TranMat[0]*0$   $fy'$   $cy'$   $0$   $0$   $0$   $1$   $0$   $fx',fy'$  : rectified focus ;  $cx'$ ,  $cy'$  : rectified principle point

#### 8.4.1.10 nLineBuffers

```
unsigned short eSPCtrl_RectLogData::nLineBuffers
```

Linebuffer for Hardware limitation < 60

#### 8.4.1.11 OutImgHeight

```
unsigned short eSPCtrl_RectLogData::OutImgHeight
```

Output image height

#### 8.4.1.12 OutImgWidth

```
unsigned short eSPCtrl_RectLogData::OutImgWidth
```

Output image width(SideBySide image)

#### 8.4.1.13 RECT\_AvgErr

```
float eSPCtrl_RectLogData::RECT_AvgErr
```

Reprojection error

#### 8.4.1.14 RECT\_Crop\_Col\_BG\_L

```
unsigned short eSPCtrl_RectLogData::RECT_Crop_Col_BG_L
```

Rectified image crop column begin

**8.4.1.15 RECT\_Crop\_Col\_ED\_L**

```
unsigned short eSPCtrl_RectLogData::RECT_Crop_Col_ED_L
```

Rectified image crop column end

**8.4.1.16 RECT\_Crop\_Row\_BG**

```
unsigned short eSPCtrl_RectLogData::RECT_Crop_Row_BG
```

Rectified image crop row begin

**8.4.1.17 RECT\_Crop\_Row\_ED**

```
unsigned short eSPCtrl_RectLogData::RECT_Crop_Row_ED
```

Rectified image crop row end

**8.4.1.18 RECT\_CropEnable**

```
int eSPCtrl_RectLogData::RECT_CropEnable
```

Rectified image crop

**8.4.1.19 RECT\_Scale\_Col\_M**

```
unsigned char eSPCtrl_RectLogData::RECT_Scale_Col_M
```

Rectified image scale column factor M

**8.4.1.20 RECT\_Scale\_Col\_N**

```
unsigned char eSPCtrl_RectLogData::RECT_Scale_Col_N
```

Rectified image scale column factor N Rectified image scale column ratio = Scale\_Col\_N/ Scale\_Col\_M

**8.4.1.21 RECT\_Scale\_Row\_M**

```
unsigned char eSPCtrl_RectLogData::RECT_Scale_Row_M
```

Rectified image scale row factor M

**8.4.1.22 RECT\_Scale\_Row\_N**

```
unsigned char eSPCtrl_RectLogData::RECT_Scale_Row_N
```

Rectified image scale row factor N

**8.4.1.23 RECT\_ScaleEnable**

```
int eSPCtrl_RectLogData::RECT_ScaleEnable
```

Rectified image scale

**8.4.1.24 RECT\_ScaleHeight**

```
unsigned short eSPCtrl_RectLogData::RECT_ScaleHeight
```

Input image height(Single image) \*RECT\_Scale\_Row\_N /RECT\_Scale\_Row\_M

**8.4.1.25 RECT\_ScaleWidth**

```
unsigned short eSPCtrl_RectLogData::RECT_ScaleWidth
```

Input image width(Single image) \*RECT\_Scale\_Col\_N /RECT\_Scale\_Col\_M

**8.4.1.26 RotaMat**

```
float eSPCtrl_RectLogData::RotaMat[9]
```

Rotation matrix between the left and right camera coordinate systems.  $\begin{bmatrix} 0 & 1 & 2 \\ | & | & | \\ X_{cr} & & \\ | & | & | \\ 3 & 4 & 5 \\ | & | & | \\ * & & \\ Y_{cr} \end{bmatrix} \Rightarrow cr$   
= right camera coordinate  $\begin{bmatrix} 6 & 7 & 8 \\ | & | & | \\ Z_{cr} \end{bmatrix}$

**8.4.1.27 RRotaMat**

```
float eSPCtrl_RectLogData::RRotaMat[9]
```

3x3 rectification transform (rotation matrix) for the left camera.  $\begin{bmatrix} 0 & 1 & 2 \\ | & | & | \\ X_{cr} & & \\ | & | & | \\ 3 & 4 & 5 \\ | & | & | \\ * & & \\ Y_{cr} \end{bmatrix} \Rightarrow cr$  = right camera coordinate  $\begin{bmatrix} 6 & 7 & 8 \\ | & | & | \\ Z_{cr} \end{bmatrix}$

**8.4.1.28 TranMat**

```
float eSPCtrl_RectLogData::TranMat[3]
```

Translation vector between the coordinate systems of the cameras.  $\begin{bmatrix} 0 \\ | & | \\ X_{cr} & \\ | & | \\ 1 \\ + & \\ Y_{cr} \end{bmatrix} \Rightarrow cr$  = right camera coordinate  $\begin{bmatrix} 2 \\ | & | \\ Z_{cr} \end{bmatrix}$

**8.4.1.29 uByteArray**

```
unsigned char eSPCtrl_RectLogData::uByteArray[1024]
```

union data defined as below struct { }

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

- [eSPDI/eSPDI\\_def.h](#)

## 8.5 GyroTag Struct Reference

### Public Attributes

- short **x**
- short **y**
- short **z**

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

- [eSPDI/eSPDI\\_def.h](#)

## 8.6 packet\_s Struct Reference

### Public Attributes

- int **len**
- int **serial**
- bool **bisRGB**
- bool **bisReady**
- union {  
    unsigned char **buffer\_yuyv** [2 \*2560 \*2560]  
    unsigned char **buffer\_RGB** [3 \*2560 \*2560]  
};

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

- [eSPDI/eSPDI\\_def.h](#)

## 8.7 PointCloudInfo Struct Reference

### Public Attributes

- float **centerX**
- float **centerY**
- float **focalLength**
- float **disparityToW** [2048]
- int **disparity\_len**
- WORD **wDepthType**
- float **focalLength\_K**
- float **baseline\_K**
- float **diff\_K**
- int **depth\_image\_edian**

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

- [eSPDI/eSPDI\\_def.h](#)

## 8.8 tagAPC\_STREAM\_INFO Struct Reference

### Public Attributes

- int **nWidth**
- int **nHeight**
- BOOL **bFormatMJPG**

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

- [eSPDI/eSPDI\\_def.h](#)

## 8.9 tagDEVINFORMATION Struct Reference

### Public Attributes

- unsigned short **wPID**
- unsigned short **wVID**
- char \* **strDevName**
- unsigned short **nChipID**
- unsigned short **nDevType**

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

- [eSPDI/eSPDI\\_def.h](#)

## 8.10 tagDEVSEL Struct Reference

### Public Attributes

- int **index**

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

- [eSPDI/eSPDI\\_def.h](#)

## 8.11 tagKEEP\_DATA\_CTRL Struct Reference

### Public Attributes

- bool **blsSerialNumberKeep**
- bool **blsSensorPositionKeep**
- bool **blsRectificationTableKeep**
- bool **blsZDTableKeep**
- bool **blsCalibrationLogKeep**

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

- [eSPDI/eSPDI\\_def.h](#)



## 8.12 tagZDTableInfo Struct Reference

### Public Attributes

- int **nIndex**
- int **nDataType**

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

- [eSPDI/eSPDI\\_def.h](#)



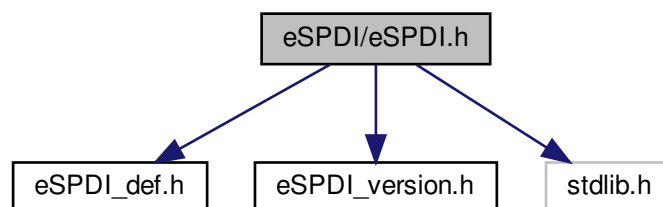
## Chapter 9

# File Documentation

### 9.1 eSPDI/eSPDI.h File Reference

functions definitions

```
#include "eSPDI_def.h"
#include "eSPDI_version.h"
#include <stdlib.h>
Include dependency graph for eSPDI.h:
```



#### Functions

- int [APC\\_Init](#) (void \*\*ppHandleEYSD, bool blsLogEnabled)  
*entry point of EYSD camera SDK including 1.create a CEYSD class for accessing oncoming APIs 2.find out EYSD devices 3.create a CVideoDevice class for video streaming and hardware access*
- int [APC\\_FindDevice](#) (void \*pHandleEYSD)  
*find out all EYSD USB devices by PID, VID and ChipID, also remember device types*
- void [APC\\_Release](#) (void \*\*ppHandleEYSD)  
*release resource that APC\_Init had allocated*
- int [APC\\_RefreshDevice](#) (void \*pHandleEYSD)  
*refresh all EYSD UVC devices*
- int [APC\\_SwitchBaseline](#) (int index)  
*Swich the baseline index.*

- bool [APC\\_IsMLBaseLine](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)  
*Check the device is multiple baseline device.*
- int [APC\\_DoFusion](#) (unsigned char \*\*pDepthBufList, double \*pDepthMerge, unsigned char \*pDepthMergeFlag, int nDWidth, int nDHeight, double fFocus, double \*pBaseline, double \*pWRNear, double \*pWRFar, double \*pWRFusion, int nMergeNum, bool bdepth2Byte11bit, int method)  
*Do Fusion Merge.*
- int [APC\\_GetDeviceNumber](#) (void \*pHandleEYSD)  
*get EYSD USB device numbers*
- int [APC\\_GetDeviceInfo](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, [DEVINFORMATION](#) \*pdevinfo)  
*get informations of EYSD UVC devices, see DEVINFORMATION*
- int [APC\\_GetDeviceInfoMBL\\_15cm](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, [DEVINFORMATION](#) \*pdevinfo)  
*get informations of EYSD UVC devices, see DEVINFORMATION*
- int [APC\\_SelectDevice](#) (void \*pHandleEYSD, int dev\_index)  
*do not support currently*
- bool [APC\\_IsInterleaveDevice](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)  
*check module support interleave function or not*
- int [APC\\_EnableInterleave](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, bool enable)  
*enable or disable interleave function*
- int [APC\\_SetPixelFormat](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, [PIXEL\\_FMT](#) fmt)
- int [APC\\_SetControlCounterMode](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char nValue)  
*enable or disable interleave function*
- int [APC\\_GetControlCounterMode](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*nValue)  
*enable or disable interleave function*
- int [APC\\_GetSensorRegister](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nId, unsigned short address, unsigned short \*pValue, int flag, [SENSORMODE\\_INFO](#) SensorMode)
- int [APC\\_SetSensorRegister](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nId, unsigned short address, unsigned short nValue, int flag, [SENSORMODE\\_INFO](#) SensorMode)  
*set sensor register value*
- int [APC\\_GetFWRegister](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short address, unsigned short \*pValue, int flag)  
*get firmware register value*
- int [APC\\_SetFWRegister](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short address, unsigned short nValue, int flag)  
*set firmware register value*
- int [APC\\_GetHWRegister](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short address, unsigned short \*pValue, int flag)  
*get hardware register value*
- int [APC\\_SetHWRegister](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short address, unsigned short nValue, int flag)  
*set hardware register*
- int [APC\\_GetMultiBytesHWRegister](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short address, unsigned char \*Data, int size, int flag)  
*set hardware register*
- int [APC\\_SetMultiBytesHWRegister](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short address, unsigned char \*Data, int size, int flag)  
*set hardware register*
- int [APC\\_GetBusInfo](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, char \*pszBusInfo, int \*pActualLength)  
*get the firmware version of device, the version is a string*
- int [APC\\_GetFwVersion](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, char \*pszFwVersion, int nBufferSize, int \*pActualLength)

- get the firmware version of device, the version is a string*

  - int [APC\\_GetPidVid](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short \*pPidBuf, unsigned short \*pVidBuf)

*get PID(product ID) and VID(vendor ID) of device*

  - int [APC\\_SetPidVid](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short \*pPidBuf, unsigned short \*pVidBuf)

*set PID and VID to device*

  - int [APC\\_GetSerialNumber](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*pData, int nbufferSize, int \*pLen)

*get device serial number*

  - int [APC\\_SetSerialNumber](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*pData, int nLen)

*set serial number to device*

  - int [APC\\_GetYOffset](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*buffer, int BufferLength, int \*pActualLength, int index)

*get Y offset (file ID 30+) value*

  - int [APC\\_GetRectifyTable](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*buffer, int BufferLength, int \*pActualLength, int index)

*get rectify values (file ID 40+) from flash*

  - int [APC\\_GetZDTable](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*buffer, int BufferLength, int \*pActualLength, [PZDTABLEINFO](#) pZDTableInfo)

*get disparity and Z values from flash*

  - int [APC\\_GetLogData](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*buffer, int BufferLength, int \*pActualLength, int index, CALIBRATION\_LOG\_TYPE type)

*get log data from flash*

  - int [APC\\_GetUserData](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*buffer, int BufferLength, USERDATA\_SECTION\_INDEX usi)

*get user data from flash*

  - int [APC\\_SetYOffset](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*buffer, int BufferLength, int \*pActualLength, int index)

*set Y offset values*

  - int [APC\\_SetRectifyTable](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*buffer, int BufferLength, int \*pActualLength, int index)

*set rectify values to flash*

  - int [APC\\_SetZDTable](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*buffer, int BufferLength, int \*pActualLength, [PZDTABLEINFO](#) pZDTableInfo)

*set disparity and Z values to flash*

  - int [APC\\_SetLogData](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*buffer, int BufferLength, int \*pActualLength, int index)

*set log data to flash*

  - int [APC\\_SetUserData](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*buffer, int BufferLength, USERDATA\_SECTION\_INDEX usi)

*set user data to flash*

  - int [APC\\_ReadFlashData](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, FLASH\_DATA\_TYPE fdt, BYTE \*pBuffer, unsigned long int BufferLength, unsigned long int \*pActualLength)

*read firmware code(.bin) form flash The firmware code is the combination of boot loader, firmware body and plug-in data. This input buffer length has to match with the flash data type*

  - int [APC\\_WriteFlashData](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, FLASH\_DATA\_TYPE fdt, BYTE \*pBuffer, unsigned long int BufferLength, bool blsDataVerify, [KEEP\\_DATA\\_CTRL](#) kdc)

*write firmware code(.bin) to flash The firmware code is the combination of boot loader, firmware body and plug-in data, also can keep original functions(Serial Number, Sensor Position, RectificationTable, ZD Table and CalibrationLog) on camera flash by KEEP\_DATA\_CTRL control*

  - int [APC\\_GetDevicePortType](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, USB\_PORT\_TYPE \*pUSB\_Port\_Type)

*Get Device USB-port-type.*

- int [APC\\_GetDeviceResolutionList](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nMaxCount, [APC\\_STREAM\\_INFO](#) \*pStreamInfo0, int nMaxCvoidount1, [APC\\_STREAM\\_INFO](#) \*pStreamInfo1)

*get the device resolution list*

- int [APC\\_Setup\\_v4l2\\_requestbuffers](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int cnt)

*Setup v4l2 request buffers, default = 4.*

- int [APC\\_OpenDevice](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nEP0Width, int nEP0Height, bool bEP0MJPEG, int nEP1Width, int nEP1Height, DEPTH\_TRANSFER\_CTRL dtc=DEPTH\_IMG\_NON\_TRANSFER, bool blsOutputRGB24=false, void \*phWndNotice=0, int \*pFPS=0, CONTROL\_MODE cm=IMAGE\_SN\_NONSYNC)

*the implement layer to open EYSD camera device by V4L2(<https://en.wikipedia.org/wiki/Video4Linux>), can open color and depth at one time call, do functions as below,*

- int [APC\\_OpenDevice2](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nEP0Width, int nEP0Height, bool bEP0MJPEG, int nEP1Width, int nEP1Height, DEPTH\_TRANSFER\_CTRL dtc=DEPTH\_IMG\_NON\_TRANSFER, bool blsOutputRGB24=false, void \*phWndNotice=0, int \*pFPS=0, CONTROL\_MODE cm=IMAGE\_SN\_NONSYNC)

*the implement layer to open EYSD camera device by V4L2(<https://en.wikipedia.org/wiki/Video4Linux>), can open color and depth at one time call, do functions as below,*

- int [APC\\_OpenDeviceMBL](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nEP0Width, int nEP0Height, bool bEP0MJPEG, int nEP1Width, int nEP1Height, DEPTH\_TRANSFER\_CTRL dtc=DEPTH\_IMG\_NON\_TRANSFER, bool blsOutputRGB24=false, void \*phWndNotice=0, int \*pFPS=0, CONTROL\_MODE cm=IMAGE\_SN\_NONSYNC)

*the implement layer to open Multiple Base Line EYSD camera device by V4L2(<https://en.wikipedia.org/wiki/Video4Linux>), can open color and depth at one time call, do functions as below,*

- int [APC\\_CloseDeviceMBL](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)

*close Multiple Base Linedevice and free resource*

- int [APC\\_CloseDevice](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)

*close device and free resource*

- int [APC\\_CloseDeviceEx](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)

*close device and free resource for warm reset*

- int [APC\\_GetImage](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*pBuf, unsigned long int \*pImageSize, int \*pSerial=0, int nDepthDataType=0)

*get color or depth pin image by issuing V4L2's IOCTL to get frame data*

- int [APC\\_GetColorImage](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*pBuf, unsigned long int \*pImageSize, int \*pSerial=0, int nDepthDataType=0)

*get color image by issuing V4L2's IOCTL to get frame data*

- int [APC\\_GetColorImageWithTimestamp](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*pBuf, unsigned long int \*pImageSize, int \*pSerial, int nDepthDataType, int64\_t \*pcur\_tv\_sec, int64\_t \*pcur\_tv\_usec)

*get color image by issuing V4L2's IOCTL to get frame data*

- int [APC\\_GetDepthImage](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*pBuf, unsigned long int \*pImageSize, int \*pSerial=0, int nDepthDataType=0)

*get depth image by issuing V4L2's IOCTL to get frame data*

- int [APC\\_GetDepthImageWithTimestamp](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*pBuf, unsigned long int \*pImageSize, int \*pSerial, int nDepthDataType, int64\_t \*pcur\_tv\_sec, int64\_t \*pcur\_tv\_usec)

*get color image by issuing V4L2's IOCTL to get frame data*

- int [APC\\_SetupBlock](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, bool enable)

*get color or depth pin image by issuing V4L2's IOCTL to get frame data*

- int [APC\\_Get\\_Color\\_30\\_mm\\_depth](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*pBuf, unsigned long int \*pImageSize, int \*pSerial=0, int nDepthDataType=0)

*get color or depth pin image by issuing V4L2's IOCTL to get frame data*

- int [APC\\_Get\\_60\\_mm\\_depth](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*pBuf, unsigned long int \*pImageSize, int \*pSerial=0, int nDepthDataType=0)

- get color or depth pin image by issuing V4L2's IOCTL to get frame data*
- int [APC\\_Get\\_150\\_mm\\_depth](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*pBuf, unsigned long int \*pImageSize, int \*pSerial=0, int nDepthDataType=0)
- get color or depth pin image by issuing V4L2's IOCTL to get frame data*
- int [APC\\_Get2Image](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*pColorImgBuf, BYTE \*pDepthImgBuf, unsigned long int \*pColorImageSize, unsigned long int \*pDepthImageSize, int \*pSerial=0, int \*pSerial2=0, int nDepthDataType=0)
- get color and/or depth pin images see APC\_GetImage for detailed description*
- int [APC\\_Get2ImageWithTimestamp](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*pColorImgBuf, BYTE \*pDepthImgBuf, unsigned long int \*pColorImageSize, unsigned long int \*pDepthImageSize, int \*pColorSerial, int \*pDepthSerial, int nDepthDataType, int64\_t \*pcur\_tv\_sec, int64\_t \*pcur\_tv\_usec)
- int [APC\\_GetExposureTime](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nSensorMode, float \*pfExpTimeMS)
- get exposure time of ISP setting in millisecond the target sensor type was set in [APC\\_SetSensorTypeName\(\)](#)*
- int [APC\\_SetExposureTime](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nSensorMode, float fExpTimeMS)
- set exposure time of ISP sensor setting the target sensor type was set in [APC\\_SetSensorTypeName\(\)](#)*
- int [APC\\_GetGlobalGain](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nSensorMode, float \*pfGlobalGain)
- get global gain of ISP setting the target sensor type was set in [APC\\_SetSensorTypeName\(\)](#)*
- int [APC\\_SetGlobalGain](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nSensorMode, float fGlobalGain)
- set global gain of ISP sensor setting the target sensor type was set in [APC\\_SetSensorTypeName\(\)](#)*
- int [APC\\_SetSensorTypeName](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, SENSOR\_TYPE\_NAME stn)
- set the sensor type you want to work on*
- int [APC\\_GetColorGain](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nSensorMode, float \*pfGainR, float \*pfGainG, float \*pfGainB)
- get color gain of ISP setting the target sensor type was set in [APC\\_SetSensorTypeName\(\)](#)*
- int [APC\\_SetColorGain](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nSensorMode, float fGainR, float fGainG, float fGainB)
- set color gain of ISP*
- bool [APC\\_GetThermalFD](#) (void \*pHandleEYSD, int \*p\_FD)
- get file description of thermal device*
- int [APC\\_GetAccMeterValue](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int \*pX, int \*pY, int \*pZ)
- get acc meter value*
- int [APC\\_EnableAE](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)
- enable auto exposure(AE) function of ISP*
- int [APC\\_DisableAE](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)
- disable auto exposure(AE) function of ISP*
- int [APC\\_EnableAWB](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)
- enable auto white balance function of ISP*
- int [APC\\_DisableAWB](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)
- disable auto white balance of ISP*
- int [APC\\_GetAESTatus](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, PAE\_STATUS pAESTatus)
- get auto exposure(AE) is enabled or disable*
- int [APC\\_GetAWBStatus](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, PAWB\_STATUS pAWBStatus)
- get auto white balance(AWB) is enabled or disable*
- int [APC\\_GetGPIOValue](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nGPIOIndex, BYTE \*pValue)
- get GPIO values*
- int [APC\\_SetGPIOValue](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nGPIOIndex, BYTE nValue)
- set GPIO values*

- int [APC\\_SetGPIOCtrl](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nGPIOIndex, BYTE nValue)  
*set GPIO I/O control*
- int [APC\\_GetCTPropVal](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nId, long int \*pValue)  
*get camera terminal(CT) property value By v4l2\_control to get control value of camera terminal*
- int [APC\\_SetCTPropVal](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nId, long int nValue)  
*set camera terminal property values By v4l2\_control to set*
- int [APC\\_GetPUPPropVal](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nId, long int \*pValue)  
*get processing unit property value by v4l2\_control to get processing unit(PU) property value*
- int [APC\\_SetPUPPropVal](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nId, long int nValue)  
*set processing unit property value by v4l2\_control to set processing unit(PU) property value*
- int [APC\\_GetCTRRangeAndStep](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nId, int \*pMax, int \*pMin, int \*pStep, int \*pDefault, int \*pFlags)  
*set camera terminal property values By v4l2\_queryctrl to get control values of camera terminal(CT) this enumeration contained the following properties: V4L2\_CID\_EXPOSURE\_AUTO V4L2\_CID\_EXPOSURE\_AUTO\_PRIORITY V4L2\_CID\_EXPOSURE\_ABSOLUTE V4L2\_CID\_EXPOSURE V4L2\_CID\_FOCUS\_ABSOLUTE V4L2\_CID\_FOCUS\_RELATIVE V4L2\_CID\_FOCUS\_AUTO V4L2\_CID\_IRIS\_ABSOLUTE V4L2\_CID\_IRIS\_RELATIVE V4L2\_CID\_ZOOM\_ABSOLUTE V4L2\_CID\_ZOOM\_RELATIVE V4L2\_CID\_PAN\_ABSOLUTE V4L2\_CID\_PAN\_RELATIVE V4L2\_CID\_TILT\_ABSOLUTE V4L2\_CID\_TILT\_RELATIVE V4L2\_CID\_PRIVACY*
- int [APC\\_GetPURangeAndStep](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int nId, int \*pMax, int \*pMin, int \*pStep, int \*pDefault, int \*pFlags)  
*get processing unit property value By v4l2\_queryctrl to get property values of processing unit(PU) this enumeration contained the following properties: V4L2\_CID\_BACKLIGHT\_COMPENSATION V4L2\_CID\_BRIGHTNESS V4L2\_CID\_CONTRAST V4L2\_CID\_GAIN V4L2\_CID\_POWER\_LINE\_FREQUENCY V4L2\_CID\_HUE V4L2\_CID\_HUE\_AUTO V4L2\_CID\_SATURATION V4L2\_CID\_SHARPNESS V4L2\_CID\_GAMMA V4L2\_CID\_WHITE\_BALANCE\_TEMPERATURE V4L2\_CID\_AUTO\_WHITE\_BALANCE*
- int [APC\\_SetDepthDataType](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short nValue)  
*set depth data type, 11 bit for disparity data, 14 bit for Z data notice: only PUMA type IC can support this setting*
- int [APC\\_GetDepthDataType](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short \*pValue)  
*get current depth data type setting*
- int [APC\\_SetInterleaveMode](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, bool enable)  
*set depth data type, 11 bit for disparity data, 14 bit for Z data notice: only PUMA type IC can support this setting*
- int [APC\\_GetInterleaveMode](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, bool \*pValue)  
*get current depth data type setting*
- int [APC\\_SetCurrentIRValue](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short nValue)  
*set infrared radiation(IR) value of PUMA type IC*
- int [APC\\_GetCurrentIRValue](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short \*pValue)  
*get infrared radiation(IR) value of PUMA type IC*
- int [APC\\_GetIRMinValue](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short \*pValue)  
*get minimum IR value of camera module*
- int [APC\\_SetIRMaxValue](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short nValue)  
*get maximum IR value of camera module*
- int [APC\\_GetIRMaxValue](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short \*pValue)  
*get maximum IR value of camera module*
- int [APC\\_SetIRMode](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short nValue)  
*enable or disable IRs*
- int [APC\\_GetIRMode](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short \*pValue)  
*to check IR is turn on or off*
- int [APC\\_GetRectifyLogData](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, [eSPCtrl\\_RectLogData](#) \*pData, int index)  
*get rectify log data from flash, just for AXES1 device type*
- int [APC\\_GetRectifyMatLogData](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, [eSPCtrl\\_RectLogData](#) \*pData, int index)  
*get rectify log data from flash, just for PUMA device type*



- int [APC\\_EnablePostProcess](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, bool bEnable)  
*Not support now.*
- int [APC\\_PostInitial](#) (void \*pHandleEYSD)  
*Not support now.*
- int [APC\\_PostEnd](#) (void \*pHandleEYSD)  
*Not support now.*
- int [APC\\_ProcessFrame](#) (void \*pHandleEYSD, unsigned char \*pYUY2Buf, unsigned char \*pDepthBuf, unsigned char \*OutputBuf, int width, int height)  
*Not support now.*
- int [APC\\_PostSetParam](#) (void \*pHandleEYSD, int Idx, int Val)  
*Not support now.*
- int [APC\\_PostGetParam](#) (void \*pHandleEYSD, int Idx, int \*pVal)  
*Not support now.*
- int [APC\\_CreateSwPostProc](#) (int depthBits, void \*\*handle)  
*create a software post process class*
- int [APC\\_ReleaseSwPostProc](#) (void \*\*handle)  
*release a software post process class*
- int [APC\\_DoSwPostProc](#) (void \*pHandleEYSD, unsigned char \*colorBuf, bool isColorRgb24, unsigned char \*depthBuf, unsigned char \*outputBuf, int width, int height)  
*do software post process on a depth buffer*
- int [APC\\_FlyingDepthCancellation\\_D8](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*pdepthD8, int width, int height)  
*Flying Pixel Depth Cancellation, just for EX8029.*
- int [APC\\_FlyingDepthCancellation\\_D11](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*pdepthD11, int width, int height)  
*Flying Pixel Depth Cancellation.*
- int [APC\\_Convert\\_Depth\\_Y\\_To\\_Buffer](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*depth\_y, unsigned char \*rgb, unsigned int width, unsigned int height, bool color, unsigned short nDepth↵  
DataType)  
*Convert Depth to RGB color or gray.*
- int [APC\\_Convert\\_Depth\\_Y\\_To\\_Buffer\\_offset](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*depth\_y, unsigned char \*rgb, unsigned int width, unsigned int height, bool color, unsigned short n↵  
DepthDataType, int offset)  
*Convert Depth to RGB color or gray, added offset for 3cm baseline.*
- int [APC\\_EnableSensorIF](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, bool blsEnable)  
*enable or disable sensor IF*
- int [APC\\_getUACNAME](#) (char \*input, char \*output)  
*Get EYSD UAC Name.*
- int [APC\\_InitialUAC](#) (char \*deviceName)  
*UAC inital function.*
- int [APC\\_WriteWaveHeader](#) (int fd)  
*Write Wave Header.*
- int [APC\\_WriteWaveEnd](#) (int fd, size\_t length)  
*Modified Wave Header.*
- int [APC\\_GetUACData](#) (unsigned char \*buffer, int length)  
*UAC inital function.*
- int [APC\\_ReleaseUAC](#) (void)  
*UAC inital function.*
- int [APC\\_InitialFlexibleGyro](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)  
*gyro sensor inital function*
- int [APC\\_ReleaseFlexibleGyro](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)  
*gyro sensor release function*

- int [APC\\_GetFlexibleGyroData](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int length, unsigned char \*pGyroData)  
*getting gyro data function*
- int [APC\\_GetFlexibleGyroLength](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short \*GyroLen)  
*getting length of gyro data function.*
- int [APC\\_GetImageInterrupt](#) (void)  
*Get Image interrupt function Get the image interrupt and then read Gyro data.*
- int [APC\\_InitialHidGyro](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)  
*gyro sensor initial function*
- int [APC\\_ReleaseHidGyro](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)  
*gyro sensor release function*
- int [APC\\_GetHidGyro](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*pBuffer, int length)  
*getting gyro data function*
- int [APC\\_SetupHidGyro](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*pCmdBuf, int cmdlength)  
*getting gyro data function*
- int [APC\\_GetInfoHidGyro](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*pCmdBuf, int cmdlength, unsigned char \*pResponseBuf, int \*resplength)  
*getting gyro data function*
- int [APC\\_GenerateLutFile](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, const char \*filename)  
*generate look up table(LUT) for spherical display this function reads the camera user data and generate a LUT file using for 360 degree preview*
- int [APC\\_SaveLutData](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, const char \*filename)  
*Save LUT parameters in the specified file.*
- int [APC\\_GetLutData](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, BYTE \*buffer, int nSize)  
*Read LUT parameters into the specified buffer.*
- int [APC\\_EncryptMP4](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, const char \*filename)  
*encrypt a H.264 video*
- int [APC\\_DecryptMP4](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, const char \*filename)  
*decrypt a H.264 video was generated by [APC\\_EncryptMP4\(\)](#)*
- int [APC\\_InjectExtraDataToMp4](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, const char \*filename, const char \*data, int dataLen)  
*APC\_InjectExtraDataToMp4.*
- int [APC\\_RetrieveExtraDataFromMp4](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, const char \*filename, char \*data, int \*dataLen)  
*APC\_RetrieveExtraDataFromMp4.*
- int [APC\\_EncryptString](#) (const char \*src, char \*dst)  
*APC\_EncryptString.*
- int [APC\\_DecryptString](#) (const char \*src, char \*dst)  
*APC\_DecryptString.*
- int [APC\\_EncryptString](#) (const char \*src1, const char \*src2, char \*dst)  
*APC\_EncryptString.*
- int [APC\\_DecryptString](#) (const char \*src, char \*dst1, char \*dst2)  
*APC\_DecryptString.*
- int [APC\\_GetAutoExposureMode](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short \*mode)  
*Get Auto Exposure Mode.*
- int [APC\\_SetAutoExposureMode](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned short mode)  
*Setup Auto Exposure Mode.*
- int [APC\\_RotateImg90](#) ([APCImageType::Value](#) imgType, int width, int height, unsigned char \*src, unsigned char \*dst, int len, bool clockwise)  
*Rotate the image to 90 degree.*

- int [APC\\_RotateImg180](#) (APCImageType::Value imgType, int width, int height, unsigned char \*src, unsigned char \*dst, int len)  
*Rotate the image to 180 degree.*
- int [APC\\_ResizeImgToHalf](#) (APCImageType::Value imgType, int width, int height, unsigned char \*src, unsigned char \*dst, int len)  
*Resize the image to half.*
- int [APC\\_ImgMirro](#) (APCImageType::Value imgType, int width, int height, unsigned char \*src, unsigned char \*dst)  
*Make the image to Mirro.*
- int [APC\\_RGB2BMP](#) (char \*filename, int width, int height, unsigned char \*data)  
*RGB to BMP.*
- int [APC\\_HoleFilled](#) (unsigned short \*pDImgIn, unsigned short \*pDImgOut, int width, int height, int holeFilldiff)  
*Hole Filled.*
- int [APC\\_InitialCmdFiFo](#) (const char \*pfifoName, int \*pFileDescription, bool bRead)  
*Cmd FiFo Initial function.*
- int [APC\\_CloseCmdFiFo](#) (int FileDescription)  
*Cmd FiFo Close function.*
- int [APC\\_WriteCmdFiFo](#) (int FileDescription, unsigned char \*pCmd, int len)  
*Write Cmd FiFo function.*
- int [APC\\_ReadCmdFiFo](#) (int FileDescription, unsigned char \*pBuf, int len)  
*Read Cmd FiFo function.*
- int [APC\\_InitSRB](#) (void \*\*pSmbHandle, int QueueSize, char \*queueName)  
*Intial the SRB(Share Ring Buffering)*
- int [APC\\_PutSRB](#) (void \*pSmbHandle, [srb\\_packet\\_s](#) \*pPacket)  
*Put Packet to SRB.*
- int [APC\\_GetSRB](#) (void \*pSmbHandle, [srb\\_packet\\_s](#) \*pPacket)  
*Get Packet from SRB.*
- int [APC\\_DepthMerge](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*\*pDepthBufList, float \*pDepthMergeOut, unsigned char \*pDepthMergeFlag, int nDWidth, int nDHeight, float fFocus, float \*pBaseline, float \*pWRNear, float \*pWRFar, float \*pWRFusion, int nMergeNum)  
*do depth merge*
- int [APC\\_GetPointCloud](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*ImgColor, int CW, int CH, unsigned char \*ImgDepth, int DW, int DH, [PointCloudInfo](#) \*pPointCloudInfo, unsigned char \*pPointCloudRGB, float \*pPointCloudXYZ, float Near, float Far)  
*get point cloud*
- int [APC\\_ColorFormat\\_to\\_RGB24](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*ImgDst, unsigned char \*ImgSrc, int SrcSize, int width, int height, APCImageType::Value type)  
*get hardware post processing status*
- int [APC\\_ColorFormat\\_to\\_BGR24](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*ImgDst, unsigned char \*ImgSrc, int SrcSize, int width, int height, APCImageType::Value type)
- int [APC\\_RotateImg90](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, APCImageType::Value imgType, int width, int height, unsigned char \*src, unsigned char \*dstBuf, int len, bool clockwise)  
*Make the image to rotate.*
- int [APC\\_RotateImg180](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, APCImageType::Value imgType, int width, int height, unsigned char \*src, unsigned char \*dst, int len)  
*Rotate the image to 180 degree.*
- int [APC\\_ImgMirro](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, APCImageType::Value imgType, int width, int height, unsigned char \*src, unsigned char \*dstBuf)  
*Make the image to Mirro.*
- int [APC\\_SubSample](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*\*SubSample, unsigned char \*depthBuf, int bytesPerPixel, int width, int height, int &new\_width, int &new\_height, int mode=0, int factor=3)

- APC\_SubSample.*
- int [APC\\_HoleFill](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*depthBuf, int bytesPerPixel, int kernel\_size, int width, int height, int level, bool horizontal)
- APC\_HoleFill.*
- int [APC\\_TemporalFilter](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*depthBuf, int bytesPerPixel, int width, int height, float alpha, int history)
- APC\_TemporalFilter.*
- int [APC\\_EdgePreServingFilter](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*depthBuf, int type, int width, int height, int level, float sigma, float lumda)
- APC\_EdgePreServingFilter.*
- int [APC\\_ApplyFilters](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, unsigned char \*depthBuf, unsigned char \*subDisparity, int bytesPerPixel, int width, int height, int sub\_w, int sub\_h, int threshold=64)
- APC\_ApplyFilters.*
- int [APC\\_ResetFilters](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo)
- APC\_ResetFilters.*
- int [APC\\_EnableGPUAcceleration](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, bool enable)
- APC\_EnableGPUAcceleration.*
- int [APC\\_TableToData](#) (void \*pHandleEYSD, [PDEVSELINFO](#) pDevSelInfo, int width, int height, int TableSize, unsigned short \*Table, unsigned short \*Src, unsigned short \*Dst)
- transfer Src to Dst by Table*
- int [APC\\_InitPostProcess](#) (void \*\*ppPostProcessHandle, unsigned int nWidth, unsigned int nHeight, [APC\\_ImageType::Value](#) imageType)
- APC\_InitPostProcess.*
- int [APC\\_PostProcess](#) (void \*pPostProcessHandle, unsigned char \*pDepthData)
- APC\_PostProcess.*
- int [APC\\_ReleasePostProcess](#) (void \*pPostProcessHandle)
- APC\_ReleasePostProcess.*

### 9.1.1 Detailed Description

functions definitions

Copyright:

This file copyright (C) 2021 by eYs3D Microelectronics, Co.

An unpublished work. All rights reserved.

This file is proprietary information, and may not be disclosed or copied without the prior permission of eYs3D.

### 9.1.2 Function Documentation

## 9.1.2.1 APC\_ApplyFilters()

```
int APC_ApplyFilters (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * depthBuf,
 unsigned char * subDisparity,
 int bytesPerPixel,
 int width,
 int height,
 int sub_w,
 int sub_h,
 int threshold = 64)
```

APC\_ApplyFilters.

## Parameters

|                    |                                                              |
|--------------------|--------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD the pointer to the initilized EYSD SDK instance |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                   |
| <i>unsigned</i>    | char* depthBuf depth buffer pointer                          |
| <i>unsigned</i>    | char* subDisparity [TODO]                                    |
| <i>int</i>         | bytesPerPixel byte number of one pixel                       |
| <i>int</i>         | width depth width                                            |
| <i>int</i>         | height depth height                                          |
| <i>int</i>         | sub_w [TODO]                                                 |
| <i>int</i>         | sub_h [TODO]                                                 |
| <i>int</i>         | threshold [TODO]                                             |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.2 APC\_CloseCmdFiFo()

```
int APC_CloseCmdFiFo (
 int FileDescription)
```

Cmd FiFo Close function.

## Parameters

|            |                                  |
|------------|----------------------------------|
| <i>int</i> | FileDescription File Description |
|------------|----------------------------------|

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

### 9.1.2.3 APC\_CloseDevice()

```
int APC_CloseDevice (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

close device and free resource

#### Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

### 9.1.2.4 APC\_CloseDeviceEx()

```
int APC_CloseDeviceEx (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

close device and free resource for warm reset

#### Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

### 9.1.2.5 APC\_CloseDeviceMBL()

```
int APC_CloseDeviceMBL (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

close Multiple Base Linedevice and free resource

#### Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.6 APC\_ColorFormat\_to\_RGB24()

```
int APC_ColorFormat_to_RGB24 (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * ImgDst,
 unsigned char * ImgSrc,
 int SrcSize,
 int width,
 int height,
 APCImageType::Value type)
```

get hardware post processing status

## Parameters

|                            |                                                              |
|----------------------------|--------------------------------------------------------------|
| <i>void</i>                | *pHandleEYSD the pointer to the initilized EYSD SDK instance |
| <i>PDEVSELINFO</i>         | pDevSelInfo pointer of device select index                   |
| <i>unsigned</i>            | char *ImgDst output image buffer                             |
| <i>unsigned</i>            | char *ImgSrc input image buffer                              |
| <i>int</i>                 | SrcSize sizeof of source image                               |
| <i>int</i>                 | width input image width                                      |
| <i>int</i>                 | height input image height                                    |
| <i>APCImageType::Value</i> | type input image-format                                      |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.7 APC\_Convert\_Depth\_Y\_To\_Buffer()

```
int APC_Convert_Depth_Y_To_Buffer (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * depth_y,
 unsigned char * rgb,
 unsigned int width,
 unsigned int height,
 bool color,
 unsigned short nDepthDataType)
```

Convert Depth to RGB color or gray.

**Parameters**

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>unsigned</i>    | char *depth_y depth data,                  |
| <i>unsigned</i>    | char *rgb output data,                     |
| <i>int</i>         | width image width,                         |
| <i>int</i>         | height image height,                       |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.8 APC\_Convert\_Depth\_Y\_To\_Buffer\_offset()**

```
int APC_Convert_Depth_Y_To_Buffer_offset (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * depth_y,
 unsigned char * rgb,
 unsigned int width,
 unsigned int height,
 bool color,
 unsigned short nDepthDataType,
 int offset)
```

Convert Depth to RGB color or gray, added offset for 3cm baseline.

**Parameters**

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>unsigned</i>    | char *depth_y depth data,                  |
| <i>unsigned</i>    | char *rgb output data,                     |
| <i>int</i>         | width image width,                         |
| <i>int</i>         | height image height,                       |
| <i>int</i>         | offset dpeth_y offset,                     |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.9 APC\_CreateSwPostProc()**

```
int APC_CreateSwPostProc (
 int depthBits,
 void ** handle)
```



create a software post process class

#### Parameters

|             |                                                             |
|-------------|-------------------------------------------------------------|
| <i>int</i>  | depthBits depth bit to set                                  |
| <i>void</i> | **handle handle pointer to this software post process class |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.10 APC\_DecryptMP4()

```
int APC_DecryptMP4 (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 const char * filename)
```

decrypt a H.264 video was generated by [APC\\_EncryptMP4\(\)](#)

#### Parameters

|                    |                                                    |
|--------------------|----------------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                                 |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index         |
| <i>const</i>       | char *filename the input video file for decryption |

#### Returns

success: APC\_OK, others:see [eSPDI\\_def.h](#)

#### 9.1.2.11 APC\_DecryptString() [1/2]

```
int APC_DecryptString (
 const char * src,
 char * dst)
```

APC\_DecryptString.

#### Parameters

|              |                               |
|--------------|-------------------------------|
| <i>const</i> | char* src input string        |
| <i>char*</i> | dst output string (decrypted) |

**Returns**

success: APC\_OK, others:see [eSPDI\\_def.h](#)

**9.1.2.12 APC\_DecryptString()** [2/2]

```
int APC_DecryptString (
 const char * src,
 char * dst1,
 char * dst2)
```

APC\_DecryptString.

**Parameters**

|              |                                   |
|--------------|-----------------------------------|
| <i>const</i> | char* src input string            |
| <i>char*</i> | dst1 output string #1 (decrypted) |
| <i>char*</i> | dst2 output string #2 (decrypted) |

**Returns**

success: APC\_OK, others:see [eSPDI\\_def.h](#)

**9.1.2.13 APC\_DepthMerge()**

```
int APC_DepthMerge (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char ** pDepthBufList,
 float * pDepthMergeOut,
 unsigned char * pDepthMergeFlag,
 int nDWidth,
 int nDHeight,
 float fFocus,
 float * pBaseline,
 float * pWRNear,
 float * pWRFar,
 float * pWRFusion,
 int nMergeNum)
```

do depth merge

**Parameters**

|                    |                                                              |
|--------------------|--------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD the pointer to the initilized EYSD SDK instance |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                   |
| <i>unsigned</i>    | char** pDepthBufList [TODO]                                  |
| <i>float</i>       | *pDepthMergeOut [TODO]                                       |

## Parameters

|                 |                              |
|-----------------|------------------------------|
| <i>unsigned</i> | char *pDepthMergeFlag [TODO] |
| <i>int</i>      | nDWidth [TODO]               |
| <i>int</i>      | nDHeight [TODO]              |
| <i>float</i>    | fFocus [TODO]                |
| <i>float</i>    | * pBaseline [TODO]           |
| <i>float</i>    | * pWRNear [TODO]             |
| <i>float</i>    | * pWRFar [TODO]              |
| <i>float</i>    | * pWRFusion [TODO]           |
| <i>int</i>      | nMergeNum [TODO]             |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.14 APC\_DisableAE()

```
int APC_DisableAE (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

disable auto exposure(AE) function of ISP

## Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.15 APC\_DisableAWB()

```
int APC_DisableAWB (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

disable auto white balance of ISP

## Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.16 APC\_DoFusion()**

```
int APC_DoFusion (
 unsigned char ** pDepthBufList,
 double * pDepthMerge,
 unsigned char * pDepthMergeFlag,
 int nDWidth,
 int nDHeight,
 double fFocus,
 double * pBaseline,
 double * pWRNear,
 double * pWRFar,
 double * pWRFusion,
 int nMergeNum,
 bool bdepth2Byte11bit,
 int method)
```

Do Fusion Merge.

**Parameters**

|                 |                                                                                                                   |
|-----------------|-------------------------------------------------------------------------------------------------------------------|
| <i>unsigned</i> | char **pDepthBufList Point to Depth Buffer List                                                                   |
| <i>double</i>   | *pDepthMerge Point to Fusion output.                                                                              |
| <i>unsigned</i> | char *pDepthMergeFlag Point to Fusion select fFocus Focus vale                                                    |
| <i>int</i>      | nDWidth Image width                                                                                               |
| <i>int</i>      | nDHeight Image Height                                                                                             |
| <i>double</i>   | *pBaseline Point to baseline array m_baselineDist[0] = 30.0; m_baselineDist[1] = 60.0; m_baselineDist[2] = 150.0; |
| <i>double</i>   | *pWRNear NearWorkingRange Vecror(Container)                                                                       |
| <i>double</i>   | *pWRFar FarWorkingRange Vecror(Container)                                                                         |
| <i>double</i>   | *pWRFusion FusionWorkingRange Vecror(Container)                                                                   |
| <i>int</i>      | nMergeNum Total merges                                                                                            |
| <i>int</i>      | method method select 0: MBLBase 1: MBRbaseV0 2: MBRbaseV1                                                         |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.17 APC\_DoSwPostProc()**

```
int APC_DoSwPostProc (
 void * handle,
```

```

 unsigned char * colorBuf,
 bool isColorRgb24,
 unsigned char * depthBuf,
 unsigned char * outputBuf,
 int width,
 int height)

```

do software post process on a depth buffer

#### Parameters

|                 |                                                   |
|-----------------|---------------------------------------------------|
| <i>void*</i>    | handle handle of this software post process class |
| <i>unsigned</i> | char* colorBuf input color buffer                 |
| <i>bool</i>     | isColorRgb24 is this color buffer RGB888          |
| <i>unsigned</i> | char* depthBuf input depth buffer                 |
| <i>unsigned</i> | char* outputBuf output buffer                     |
| <i>int</i>      | width image width                                 |
| <i>int</i>      | height image height                               |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.18 APC\_EdgePreServingFilter()

```

int APC_EdgePreServingFilter (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * depthBuf,
 int type,
 int width,
 int height,
 int level,
 float sigma,
 float lumda)

```

APC\_EdgePreServingFilter.

#### Parameters

|                    |                                                              |
|--------------------|--------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD the pointer to the initilized EYSD SDK instance |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                   |
| <i>unsigned</i>    | char* depthBuf depth buffer pointer                          |
| <i>int</i>         | bytesPerPixel byte number of one pixel                       |
| <i>int</i>         | width depth width                                            |
| <i>int</i>         | height depth height                                          |
| <i>int</i>         | level [TODO]                                                 |
| <i>float</i>       | sigma [TODO]                                                 |
| <i>float</i>       | lumda [TODO]                                                 |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.19 APC\_EnableAE()**

```
int APC_EnableAE (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

enable auto exposure(AE) function of ISP

**Parameters**

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.20 APC\_EnableAWB()**

```
int APC_EnableAWB (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

enable auto white balance function of ISP

**Parameters**

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.21 APC\_EnableGPUAcceleration()**

```
int APC_EnableGPUAcceleration (
 void * pHandleEYSD,
```

```
PDEVSELINFO pDevSelInfo,
bool enable)
```

APC\_EnableGPUAcceleration.

**Parameters**

|                    |                                                              |
|--------------------|--------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD the pointer to the initilized EYSD SDK instance |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                   |
| <i>bool</i>        | enable enable it or not                                      |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.22 APC\_EnableInterleave()**

```
int APC_EnableInterleave (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 bool enable)
```

enable or disable interleave function

**Parameters**

|                    |                                                                   |
|--------------------|-------------------------------------------------------------------|
| <i>pHandleEYSD</i> | the pointer to the initilized EYSD SDK instance                   |
| <i>pDevSelInfo</i> | pointer of device select index                                    |
| <i>enable</i>      | set true to enable interleave, or set false to disable interleave |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.23 APC\_EnableSensorIF()**

```
int APC_EnableSensorIF (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 bool bIsEnable)
```

enable or disable sensor IF

**Parameters**

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>bool</i>        | bIsEnable true is enable, false is disable |



## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.24 APC\_EncryptMP4()

```
int APC_EncryptMP4 (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 const char * filename)
```

encrypt a H.264 video

## Parameters

|                    |                                                    |
|--------------------|----------------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                                 |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index         |
| <i>const</i>       | char *filename the input video file for encryption |

## Returns

success: APC\_OK, others:see [eSPDI\\_def.h](#)

## 9.1.2.25 APC\_EncryptString() [1/2]

```
int APC_EncryptString (
 const char * src,
 char * dst)
```

APC\_EncryptString.

## Parameters

|              |                               |
|--------------|-------------------------------|
| <i>const</i> | char* src input string        |
| <i>char*</i> | dst output string (encrypted) |

## Returns

success: APC\_OK, others:see [eSPDI\\_def.h](#)

## 9.1.2.26 APC\_EncryptString() [2/2]

```
int APC_EncryptString (
 const char * src1,
```

```
const char * src2,
char * dst)
```

APC\_EncryptString.

#### Parameters

|              |                               |
|--------------|-------------------------------|
| <i>const</i> | char* src1 input string #1    |
| <i>const</i> | char* src2 input string #2    |
| <i>char*</i> | dst output string (encrypted) |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.27 APC\_FindDevice()

```
int APC_FindDevice (
 void * pHandleEYSD)
```

find out all EYSD USB devices by PID, VID and ChipID, also remember device types

#### Parameters

|             |                     |
|-------------|---------------------|
| <i>void</i> | *pHandleEYSD handle |
|-------------|---------------------|

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.28 APC\_FlyingDepthCancellation\_D11()

```
int APC_FlyingDepthCancellation_D11 (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * pdepthD11,
 int width,
 int height)
```

Flying Pixel Depth Cancellation.

#### Parameters

|                    |                                             |
|--------------------|---------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index  |
| <i>unsigned</i>    | char *pdepthD11 point to input depth buffer |
| <i>int</i>         | width depth width                           |
| <i>int</i>         | height depth height                         |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.29 APC\_FlyingDepthCancellation\_D8()**

```
int APC_FlyingDepthCancellation_D8 (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * pdepthD8,
 int width,
 int height)
```

Flying Pixel Depth Cancellation, just for EX8029.

**Parameters**

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>unsigned</i>    | char *pdepthD8 point to input depth buffer |
| <i>int</i>         | width depth width                          |
| <i>int</i>         | height depth height                        |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.30 APC\_GenerateLutFile()**

```
int APC_GenerateLutFile (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 const char * filename)
```

generate look up table(LUT) for spherical display this function reads the camera user data and generate a LUT file using for 360 degree preview

**Parameters**

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>const</i>       | char* filename output LUT file name        |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.31 APC\_Get2Image()**

```
int APC_Get2Image (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * pColorImgBuf,
 BYTE * pDepthImgBuf,
 unsigned long int * pColorImageSize,
 unsigned long int * pDepthImageSize,
 int * pColorSerial = 0,
 int * pDepthSerial = 0,
 int nDepthDataType = 0)
```

get color and/or depth pin images see APC\_GetImage for detailed description

**Parameters**

|                    |                                                                                   |
|--------------------|-----------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                               |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                        |
| <i>BYTE</i>        | *pColorImgBuf buffer to store color image                                         |
| <i>BYTE</i>        | *pDepthImgBuf buffer to store depth image                                         |
| <i>unsigned</i>    | long int *pColorImageSize the actual color buffer size                            |
| <i>unsigned</i>    | long int *pDepthImageSize the actual depth buffer size                            |
| <i>int</i>         | *pColorSerial color serial number                                                 |
| <i>int</i>         | *pDepthSerial depth serial number                                                 |
| <i>int</i>         | nDepthDataType the depth data type, see definition in <a href="#">eSPDI_def.h</a> |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.32 APC\_Get\_150\_mm\_depth()**

```
int APC_Get_150_mm_depth (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * pBuf,
 unsigned long int * pImageSize,
 int * pSerial = 0,
 int nDepthDataType = 0)
```

get color or depth pin image by issuing V4L2's IOCTL to get frame data

## Parameters

|                    |                                                                                   |
|--------------------|-----------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                               |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                        |
| <i>BYTE</i>        | *pDepthImgBuf buffer to store image data                                          |
| <i>unsigned</i>    | long int *pImageSize the actual buffer size getting from device                   |
| <i>int</i>         | *pDepthSerial the serial number for synchronizing depth image                     |
| <i>int</i>         | nDepthDataType the depth data type, see definition in <a href="#">eSPDI_def.h</a> |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.33 APC\_Get\_60\_mm\_depth()

```
int APC_Get_60_mm_depth (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * pBuf,
 unsigned long int * pImageSize,
 int * pSerial = 0,
 int nDepthDataType = 0)
```

get color or depth pin image by issuing V4L2's IOCTL to get frame data

## Parameters

|                    |                                                                                   |
|--------------------|-----------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                               |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                        |
| <i>BYTE</i>        | *pBuf buffer to store image data                                                  |
| <i>unsigned</i>    | long int *pImageSize the actual buffer size getting from device                   |
| <i>int</i>         | *pSerial the serial number for synchronizing color and depth image                |
| <i>int</i>         | nDepthDataType the depth data type, see definition in <a href="#">eSPDI_def.h</a> |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.34 APC\_Get\_Color\_30\_mm\_depth()

```
int APC_Get_Color_30_mm_depth (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * pBuf,
```

```

 unsigned long int * pImageSize,
 int * pSerial = 0,
 int nDepthDataType = 0)

```

get color or depth pin image by issuing V4L2's IOCTL to get frame data

#### Parameters

|                    |                                                                                   |
|--------------------|-----------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                               |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                        |
| <i>BYTE</i>        | *pBuf buffer to store image data                                                  |
| <i>unsigned</i>    | long int *pImageSize the actual buffer size getting from device                   |
| <i>int</i>         | *pSerial the serial number for synchronizing color and depth image                |
| <i>int</i>         | nDepthDataType the depth data type, see definition in <a href="#">eSPDI_def.h</a> |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.35 APC\_GetAccMeterValue()

```

int APC_GetAccMeterValue (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int * pX,
 int * pY,
 int * pZ)

```

get acc meter value

#### Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>int</i>         | *pX X position                             |
| <i>int</i>         | *pY Y position                             |
| <i>int</i>         | *pZ Z position                             |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.36 APC\_GetAESTatus()

```

int APC_GetAESTatus (
 void * pHandleEYSD,

```

```

PDEVSELINFO pDevSelInfo,
PAE_STATUS pAESTatus)

```

get auto exposure(AE) is enabled or disable

#### Parameters

|                    |                                                                              |
|--------------------|------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                          |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                   |
| <i>PAE_STATUS</i>  | pAESTatus see enum definition as to AE_STATUS in <a href="#">eSPDI_def.h</a> |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.37 APC\_GetAutoExposureMode()

```

int APC_GetAutoExposureMode (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short * mode)

```

Get Auto Exposure Mode.

#### Parameters

|                    |                                                                                                         |
|--------------------|---------------------------------------------------------------------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle.                                                                                     |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index.                                                             |
| <i>unsigned</i>    | short* mode pointer of the mode value. 0: Average, 1: Left (or Front) camera, 2: Right (or Back) camera |

#### Returns

success: APC\_OK, others: [eSPDI\\_def.h](#)

#### 9.1.2.38 APC\_GetAWBStatus()

```

int APC_GetAWBStatus (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 PAWB_STATUS pAWBStatus)

```

get auto white balance(AWB) is enabled or disable

## Parameters

|                    |                                                                                |
|--------------------|--------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                            |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                     |
| <i>PAWB_STATUS</i> | pAWBStatus see enum definition as to AWB_STATUS in <a href="#">eSPDI_def.h</a> |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.39 APC\_GetBusInfo()

```
int APC_GetBusInfo (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 char * pszBusInfo,
 int * pActualLength)
```

get the firmware version of device, the version is a string

## Parameters

|                    |                                                      |
|--------------------|------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                  |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index           |
| <i>char</i>        | *pszBusInfo Bus information string                   |
| <i>int</i>         | *pActualLength the actual length of Bus info in byte |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.40 APC\_GetColorGain()

```
int APC_GetColorGain (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nSensorMode,
 float * pfGainR,
 float * pfGainG,
 float * pfGainB)
```

get color gain of ISP setting the target sensor type was set in [APC\\_SetSensorTypeName\(\)](#)



## Parameters

|                    |                                                                                |
|--------------------|--------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                            |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                     |
| <i>int</i>         | nSensorMode which sensor(sensor A, B or Both) to get A is 0, B is 1, Both is 2 |
| <i>float</i>       | *pfGainR pointer of red gain value of ISP setting                              |
| <i>float</i>       | *pfGainG pointer of green gain value of ISP setting                            |
| <i>float</i>       | *pfGainB pointer of blue gain value of ISP setting                             |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.41 APC\_GetColorImage()

```
int APC_GetColorImage (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * pBuf,
 unsigned long int * pImageSize,
 int * pSerial = 0,
 int nDepthDataType = 0)
```

get color image by issuing V4L2's IOCTL to get frame data

## Parameters

|                    |                                                                    |
|--------------------|--------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                         |
| <i>BYTE</i>        | *pBuf buffer to store image data                                   |
| <i>unsigned</i>    | long int *pImageSize the actual buffer size getting from device    |
| <i>int</i>         | *pSerial the serial number for synchronizing color and depth image |
| <i>int</i>         | nDepthDataType reserved, no used.                                  |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.42 APC\_GetColorImageWithTimestamp()

```
int APC_GetColorImageWithTimestamp (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * pBuf,
```

```

 unsigned long int * pImageSize,
 int * pSerial,
 int nDepthDataType,
 int64_t * pcur_tv_sec,
 int64_t * pcur_tv_usec)

```

get color image by issuing V4L2's IOCTL to get frame data

#### Parameters

|                    |                                                                          |
|--------------------|--------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                      |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                               |
| <i>BYTE</i>        | *pBuf buffer to store image data                                         |
| <i>unsigned</i>    | long int *pImageSize the actual buffer size getting from device          |
| <i>int</i>         | *pSerial the serial number for synchronizing color and depth image       |
| <i>int</i>         | nDepthDataType reserved, no used.                                        |
| <i>int64_t</i>     | *pcur_tv_sec seconds in 'v4l2_buffer' timestamp of this image data       |
| <i>int64_t</i>     | *pcur_tv_usec microseconds in 'v4l2_buffer' timestamp of this image data |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.43 APC\_GetControlCounterMode()

```

int APC_GetControlCounterMode (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * nValue)

```

enable or disable interleave function

#### Parameters

|                    |                                                                                     |
|--------------------|-------------------------------------------------------------------------------------|
| <i>pHandleEYSD</i> | the pointer to the initialized EYSD SDK instance                                    |
| <i>pDevSelInfo</i> | pointer of device select index                                                      |
| <i>*nValue</i>     | pointer to frame counter mode value, 0: Frame Counter Mode, 1: Serial Counter Mode, |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.44 APC\_GetCTPropVal()

```

int APC_GetCTPropVal (
 void * pHandleEYSD,

```

```

PDEVSELINFO pDevSelInfo,
int nId,
long int * pValue)

```

get camera terminal(CT) property value By v4l2\_control to get control value of camera terminal

this enumeration contained the following properties: V4L2\_CID\_EXPOSURE\_AUTO; V4L2\_CID\_EXPOSURE\_AUTO\_PRIORITY V4L2\_CID\_EXPOSURE\_ABSOLUTE V4L2\_CID\_EXPOSURE V4L2\_CID\_FOCUS\_ABSOLUTE V4L2\_CID\_FOCUS\_RELATIVE V4L2\_CID\_FOCUS\_AUTO V4L2\_CID\_IRIS\_ABSOLUTE V4L2\_CID\_IRIS\_RELATIVE V4L2\_CID\_ZOOM\_ABSOLUTE V4L2\_CID\_ZOOM\_RELATIVE V4L2\_CID\_PAN\_ABSOLUTE V4L2\_CID\_PAN\_RELATIVE V4L2\_CID\_TILT\_ABSOLUTE V4L2\_CID\_TILT\_RELATIVE V4L2\_CID\_PRIVACY

#### Parameters

|                    |                                                                                                         |
|--------------------|---------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                     |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                              |
| <i>int</i>         | nId specifies the member of the property set, see CT Property ID defined in <a href="#">eSPDI_def.h</a> |
| <i>int</i>         | *pValue pointer of store CT property value                                                              |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.45 APC\_GetCTRangeAndStep()

```

int APC_GetCTRangeAndStep (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nId,
 int * pMax,
 int * pMin,
 int * pStep,
 int * pDefault,
 int * pFlags)

```

set camera terminal property values By v4l2\_queryctrl to get control values of camera terminal(CT) this enumeration contained the following properties: V4L2\_CID\_EXPOSURE\_AUTO V4L2\_CID\_EXPOSURE\_AUTO\_PRIORITY V4L2\_CID\_EXPOSURE\_ABSOLUTE V4L2\_CID\_EXPOSURE V4L2\_CID\_FOCUS\_ABSOLUTE V4L2\_CID\_FOCUS\_RELATIVE V4L2\_CID\_FOCUS\_AUTO V4L2\_CID\_IRIS\_ABSOLUTE V4L2\_CID\_IRIS\_RELATIVE V4L2\_CID\_ZOOM\_ABSOLUTE V4L2\_CID\_ZOOM\_RELATIVE V4L2\_CID\_PAN\_ABSOLUTE V4L2\_CID\_PAN\_RELATIVE V4L2\_CID\_TILT\_ABSOLUTE V4L2\_CID\_TILT\_RELATIVE V4L2\_CID\_PRIVACY

#### Parameters

|                    |                                                                                                         |
|--------------------|---------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                     |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                              |
| <i>int</i>         | nId specifies the member of the property set, see CT Property ID defined in <a href="#">eSPDI_def.h</a> |
| <i>long</i>        | int *pMax maximum value, inclusive. This field gives an upper bound for the control                     |
| <i>long</i>        | int *pMin minimum value, inclusive. This field gives a lower bound for the control                      |

## Parameters

|             |                                                                                                                                                                                                                                                                                                                                           |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>long</i> | int *pStep This field gives a step size for the control see enum <a href="https://www.linuxtv.org/downloads/v4l-dvb-apis-old/vidioc-queryctrl.html">https://www.linuxtv.org/downloads/v4l-dvb-apis-old/vidioc-queryctrl.html</a> how the step value is to be used for each possible control type. Note that this an unsigned 32-bit value |
| <i>long</i> | int *pDefault The default value of a V4L2_CTRL_TYPE_INTEGER, _BOOLEAN, _BITMASK, _MENU or _INTEGER_MENU control. Not valid for other types of controls. Note that drivers reset controls to their default value only when the driver is first loaded, never afterwards.                                                                   |
| <i>long</i> | int *pFlags control flags, see <a href="https://www.linuxtv.org/downloads/v4l-dvb-apis-old/vidioc-queryctrl.html">https://www.linuxtv.org/downloads/v4l-dvb-apis-old/vidioc-queryctrl.html</a>                                                                                                                                            |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.46 APC\_GetCurrentIRValue()

```
int APC_GetCurrentIRValue (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short * pValue)
```

get infrared radiation(IR) value of PUMA type IC

## Parameters

|                    |                                               |
|--------------------|-----------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                           |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index    |
| <i>unsigned</i>    | short *pValue current 1 byte IR value setting |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.47 APC\_GetDepthDataType()

```
int APC_GetDepthDataType (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short * pValue)
```

get current depth data type setting

## Parameters

|                    |                                                      |
|--------------------|------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                  |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index           |
| <i>WORD</i>        | *pValue pointer of current depth data type in device |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.48 APC\_GetDepthImage()

```
int APC_GetDepthImage (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * pBuf,
 unsigned long int * pImageSize,
 int * pSerial = 0,
 int nDepthDataType = 0)
```

get depth image by issuing V4L2's IOCTL to get frame data

## Parameters

|                    |                                                                                   |
|--------------------|-----------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                               |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                        |
| <i>BYTE</i>        | *pBuf buffer to store image data                                                  |
| <i>unsigned</i>    | long int *pImageSize the actual buffer size getting from device                   |
| <i>int</i>         | *pSerial the serial number for synchronizing color and depth image                |
| <i>int</i>         | nDepthDataType the depth data type, see definition in <a href="#">eSPDI_def.h</a> |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.49 APC\_GetDepthImageWithTimestamp()

```
int APC_GetDepthImageWithTimestamp (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * pBuf,
 unsigned long int * pImageSize,
 int * pSerial,
 int nDepthDataType,
 int64_t * pcur_tv_sec,
 int64_t * pcur_tv_usec)
```

get color image by issuing V4L2's IOCTL to get frame data

**Parameters**

|                    |                                                                          |
|--------------------|--------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                      |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                               |
| <i>BYTE</i>        | *pBuf buffer to store image data                                         |
| <i>unsigned</i>    | long int *pImageSize the actual buffer size getting from device          |
| <i>int</i>         | *pSerial the serial number for synchronizing color and depth image       |
| <i>int</i>         | nDepthDataType reserved, no used.                                        |
| <i>int64_t</i>     | *pcur_tv_sec seconds in 'v4l2_buffer' timestamp of this image data       |
| <i>int64_t</i>     | *pcur_tv_usec microseconds in 'v4l2_buffer' timestamp of this image data |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.50 APC\_GetDeviceInfo()**

```
int APC_GetDeviceInfo (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 DEVINFORMATION * pdevinfo)
```

get informations of EYSD UVC devices, see DEVINFORMATION

**Parameters**

|                        |                                            |
|------------------------|--------------------------------------------|
| <i>void</i>            | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i>     | pDevSelInfo pointer of device select index |
| <i>DEVINFORMATION*</i> | pdevinfo pointer of device information     |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.51 APC\_GetDeviceInfoMBL\_15cm()**

```
int APC_GetDeviceInfoMBL_15cm (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 DEVINFORMATION * pdevinfo)
```

get informations of EYSD UVC devices, see DEVINFORMATION

## Parameters

|                        |                                            |
|------------------------|--------------------------------------------|
| <i>void</i>            | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i>     | pDevSelInfo pointer of device select index |
| <i>DEVINFORMATION*</i> | pdevinfo pointer of device information     |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.52 APC\_GetDeviceNumber()

```
int APC_GetDeviceNumber (
 void * pHandleEYSD)
```

get EYSD USB device numbers

## Parameters

|             |                     |
|-------------|---------------------|
| <i>void</i> | *pHandleEYSD handle |
|-------------|---------------------|

## Returns

number of EYSD device

## 9.1.2.53 APC\_GetDeviceResolutionList()

```
int APC_GetDeviceResolutionList (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nMaxCount0,
 APC_STREAM_INFO * pStreamInfo0,
 int nMaxCount1,
 APC_STREAM_INFO * pStreamInfo1)
```

get the device resolution list

## Parameters

|                        |                                               |
|------------------------|-----------------------------------------------|
| <i>void</i>            | *pHandleEYSD handle                           |
| <i>PDEVSELINFO</i>     | pDevSelInfo pointer of device select index    |
| <i>int</i>             | nMaxCount0 max count of endpoint1 resolutions |
| <i>APC_STREAM_INFO</i> | *pStreamInfo0 resolution infos of endpoint1   |
| <i>int</i>             | nMaxCount1 max count of endpoint2 resolutions |
| <i>APC_STREAM_INFO</i> | *pStreamInfo1 resolutions infos of endpoint2  |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.54 APC\_GetExposureTime()**

```
int APC_GetExposureTime (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nSensorMode,
 float * pfExpTimeMS)
```

get exposure time of ISP setting in millisecond the target sensor type was set in [APC\\_SetSensorTypeName\(\)](#)

**Parameters**

|                    |                                                                                                                                 |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD pHandleEYSD                                                                                                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                                                      |
| <i>int</i>         | nSensorMode which sensor(sensor A, B or Both) to get A is 0, B is 1, Both is 2                                                  |
| <i>float</i>       | *pfExpTimeMS pointer of getting exposure time in millisecond by pixel clock, pixel per line, exposure line to get exposure time |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.55 APC\_GetFlexibleGyroData()**

```
int APC_GetFlexibleGyroData (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int length,
 unsigned char * pGyroData)
```

getting gyro data function

**Parameters**

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>int</i>         | length Gyro Data Length                    |
| <i>unsigned</i>    | char *pGyroData pointer of Gyro Data.      |



## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.56 APC\_GetFlexibleGyroLength()

```
int APC_GetFlexibleGyroLength (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short * GyroLen)
```

getting length of gyro data function.

## Parameters

|                    |                                             |
|--------------------|---------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                          |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index  |
| <i>unsigned</i>    | short* GyroLen pointer of Gyro Data Lenhth. |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.57 APC\_GetFWRegister()

```
int APC_GetFWRegister (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short address,
 unsigned short * pValue,
 int flag)
```

get firmware register value

## Parameters

|                    |                                                                                                                                                                                                                                             |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                                                                                                                                                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                                                                                                                                                                  |
| <i>unsigned</i>    | short address register address                                                                                                                                                                                                              |
| <i>unsigned</i>    | short *pValue pointer of value got from register address                                                                                                                                                                                    |
| <i>int</i>         | flag address and value data length(2 or 1 byte) ie FG_Address_2Byte   FG_Value_2Byte is 2 byte address and 2 byte value #define FG_Address_1Byte 0x01 #define FG_Address_2Byte 0x02 #define FG_Value_1Byte 0x10 #define FG_Value_2Byte 0x20 |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.58 APC\_GetFwVersion()**

```
int APC_GetFwVersion (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 char * pszFwVersion,
 int nBufferSize,
 int * pActualLength)
```

get the firmware version of device, the version is a string

**Parameters**

|                    |                                                        |
|--------------------|--------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                    |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index             |
| <i>char</i>        | *pszFwVersion firmware version string                  |
| <i>int</i>         | nBufferSize input buffer length to receive FW version  |
| <i>int</i>         | *pActualLength the actual length of FW version in byte |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.59 APC\_GetGlobalGain()**

```
int APC_GetGlobalGain (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nSensorMode,
 float * pfGlobalGain)
```

get global gain of ISP setting the target sensor type was set in [APC\\_SetSensorTypeName\(\)](#)

**Parameters**

|                    |                                                                                |
|--------------------|--------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                            |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                     |
| <i>int</i>         | nSensorMode which sensor(sensor A, B or Both) to get A is 0, B is 1, Both is 2 |
| <i>float</i>       | *pfGlobalGain pointer of global gain value                                     |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.60 APC\_GetHidGyro()

```
int APC_GetHidGyro (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * pBuffer,
 int length)
```

getting gyro data function

## Parameters

|                    |                                              |
|--------------------|----------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                           |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index   |
| <i>unsigned</i>    | char *pGyroData pointer of Gyro Data Buffer. |
| <i>int</i>         | length Input buffer Length, should be >= 24  |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.61 APC\_GetHWRegister()

```
int APC_GetHWRegister (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short address,
 unsigned short * pValue,
 int flag)
```

get hardware register value

## Parameters

|                    |                                                                                                                                                                                                                                             |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                                                                                                                                                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                                                                                                                                                                  |
| <i>unsigned</i>    | short address register address                                                                                                                                                                                                              |
| <i>unsigned</i>    | short *pValue pointer of value got from register address                                                                                                                                                                                    |
| <i>int</i>         | flag address and value data length(2 or 1 byte) ie FG_Address_2Byte   FG_Value_2Byte is 2 byte address and 2 byte value #define FG_Address_1Byte 0x01 #define FG_Address_2Byte 0x02 #define FG_Value_1Byte 0x10 #define FG_Value_2Byte 0x20 |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.62 APC\_GetImage()**

```
int APC_GetImage (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * pBuf,
 unsigned long int * pImageSize,
 int * pSerial = 0,
 int nDepthDataType = 0)
```

get color or depth pin image by issuing V4L2's IOCTL to get frame data

**Parameters**

|                    |                                                                                   |
|--------------------|-----------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                               |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                        |
| <i>BYTE</i>        | *pBuf buffer to store image data                                                  |
| <i>unsigned</i>    | long int *pImageSize the actual buffer size getting from device                   |
| <i>int</i>         | *pSerial the serial number for synchronizing color and depth image                |
| <i>int</i>         | nDepthDataType the depth data type, see definition in <a href="#">eSPDI_def.h</a> |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.63 APC\_GetImageInterrupt()**

```
int APC_GetImageInterrupt (
 void)
```

Get Image interrupt function Get the image interrupt and then read Gyro data.

**Returns**

success: 0, others: not got interrupt

#### 9.1.2.64 APC\_GetInfoHidGyro()

```
int APC_GetInfoHidGyro (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * pCmdBuf,
 int cmdlength,
 unsigned char * pResponseBuf,
 int * resplength)
```

getting gyro data function

**Parameters**

|                    |                                               |
|--------------------|-----------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                            |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index    |
| <i>unsigned</i>    | char *pCmdBuf pointer of Gyro Cmd Buffer.     |
| <i>int</i>         | cmdlength Command Length.                     |
| <i>unsigned</i>    | char *pResponseBuf pointer of ResponseBuffer. |
| <i>int</i>         | resplength Response Length                    |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.65 APC\_GetInterleaveMode()**

```
int APC_GetInterleaveMode (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 bool * pValue)
```

get current depth data type setting

**Parameters**

|                    |                                                    |
|--------------------|----------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index         |
| <i>bool</i>        | *pValue pointer of enable/disable status in device |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.66 APC\_GetIRMaxValue()**

```
int APC_GetIRMaxValue (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short * pValue)
```

get maximum IR value of camera module

**Parameters**

|                    |                                                      |
|--------------------|------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                  |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index           |
| <i>unsigned</i>    | short *pValue the maximum 1 byte IR value can be set |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.67 APC\_GetIRMinValue()**

```
int APC_GetIRMinValue (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short * pValue)
```

get minimum IR value of camera module

**Parameters**

|                    |                                                      |
|--------------------|------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                  |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index           |
| <i>unsigned</i>    | short *pValue the minimum 1 byte IR value can be set |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.68 APC\_GetIRMode()**

```
int APC_GetIRMode (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short * pValue)
```

to check IR is turn on or off

**Parameters**

|                    |                                                                                                                                                                                                                    |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                                                                                                                                |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                                                                                                                                         |
| <i>unsigned</i>    | short *pValue get IR was enabled or not D[7:4]: Reserved D3: Channel 3 D2: Channel 2 D1: Channel 1 D0: Channel 0 1: Enable Channel 0: Disable Channel If want to control ch0 and ch1, ubMode[3:0] must set to 0x03 |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

### 9.1.2.69 APC\_GetLogData()

```
int APC_GetLogData (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * buffer,
 int BufferLength,
 int * pActualLength,
 int index,
 CALIBRATION_LOG_TYPE type)
```

get log data from flash

#### Parameters

|                             |                                                          |
|-----------------------------|----------------------------------------------------------|
| <i>void</i>                 | *pHandleEYSD handle                                      |
| <i>PDEVSELINFO</i>          | pDevSelInfo pointer of device select index               |
| <i>BYTE</i>                 | *buffer buffer to store log data                         |
| <i>int</i>                  | BufferLength input buffer length, must be 4096           |
| <i>int</i>                  | *pActualLength actual length has written to buffer       |
| <i>int</i>                  | index index to identify log data for corresponding depth |
| <i>CALIBRATION_LOG_TYPE</i> | type which calibration log to get                        |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

### 9.1.2.70 APC\_GetLutData()

```
int APC_GetLutData (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * buffer,
 int nSize)
```

Read LUT parameters into the specified buffer.

#### Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>BYTE*</i>       | buffer memory to store LUT data            |
| <i>int</i>         | nSize length of buffer in bytes            |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)



### 9.1.2.71 APC\_GetMultiBytesHWRegister()

```
int APC_GetMultiBytesHWRegister (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short address,
 unsigned char * Data,
 int size,
 int flag)
```

set hardware register

#### Parameters

|                    |                                                                                                                                                                                                                                             |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                                                                                                                                                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                                                                                                                                                                  |
| <i>unsigned</i>    | short address register address                                                                                                                                                                                                              |
| <i>unsigned</i>    | char *Data multiple-bytes register value to set                                                                                                                                                                                             |
| <i>int</i>         | size multiple-bytes register size                                                                                                                                                                                                           |
| <i>int</i>         | flag address and value data length(2 or 1 byte) ie FG_Address_1Byte   FG_Value_1Byte is 1 byte address and 1 byte value #define FG_Address_1Byte 0x01 #define FG_Address_2Byte 0x02 #define FG_Value_1Byte 0x10 #define FG_Value_2Byte 0x20 |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

### 9.1.2.72 APC\_GetPidVid()

```
int APC_GetPidVid (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short * pPidBuf,
 unsigned short * pVidBuf)
```

get PID(product ID) and VID(vendor ID) of device

#### Parameters

|                    |                                                 |
|--------------------|-------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                             |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index      |
| <i>unsigned</i>    | short *pPidBuf 4 byte buffer to store PID value |
| <i>unsigned</i>    | short *pVidBuf 4 byte buffer to store VID value |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

### 9.1.2.73 APC\_GetPointCloud()

```
int APC_GetPointCloud (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * ImgColor,
 int CW,
 int CH,
 unsigned char * ImgDepth,
 int DW,
 int DH,
 PointCloudInfo * pPointCloudInfo,
 unsigned char * pPointCloudRGB,
 float * pPointCloudXYZ,
 float Near,
 float Far)
```

get point cloud

#### Parameters

|                       |                                                              |
|-----------------------|--------------------------------------------------------------|
| <i>void</i>           | *pHandleEYSD the pointer to the initilized EYSD SDK instance |
| <i>PDEVSELINFO</i>    | pDevSelInfo pointer of device select index                   |
| <i>unsigned</i>       | char *ImgColor RGB-buffer                                    |
| <i>int</i>            | CW ImgColor width                                            |
| <i>int</i>            | CH ImgColor height                                           |
| <i>unsigned</i>       | char *ImgDepth depth-buffer                                  |
| <i>int</i>            | DW ImgDepth width                                            |
| <i>int</i>            | DH ImgDepth height                                           |
| <i>PointCloudInfo</i> | *pPointCloudInfo point-cloud information                     |
| <i>unsigned</i>       | char *pPointCloudRGB point-cloud RGB value                   |
| <i>float</i>          | *pPointCloudXYZ point-cloud XYZ value                        |
| <i>float</i>          | Near filter range near dist.                                 |
| <i>float</i>          | Far filter range far dist.                                   |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

### 9.1.2.74 APC\_GetPUPPropVal()

```
int APC_GetPUPPropVal (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nId,
 long int * pValue)
```

get processing unit property value by v4l2\_control to get processing unit(PU) property value

this enumeration contained the following properties: V4L2\_CID\_BACKLIGHT\_COMPENSATION V4L2\_CID\_BR↔  
IGHTNESS V4L2\_CID\_CONTRAST V4L2\_CID\_GAIN V4L2\_CID\_POWER\_LINE\_FREQUENCY V4L2\_CID\_HUE  
V4L2\_CID\_HUE\_AUTO V4L2\_CID\_SATURATION V4L2\_CID\_SHARPNESS V4L2\_CID\_GAMMA V4L2\_CID\_W↔  
HITE\_BALANCE\_TEMPERATURE V4L2\_CID\_AUTO\_WHITE\_BALANCE

## Parameters

|                    |                                                                                                        |
|--------------------|--------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                    |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                             |
| <i>int</i>         | nId specifies the member of the property set see PU property ID defined in <a href="#">eSPDI_def.h</a> |
| <i>long</i>        | int *pValue pointer of store PU property value                                                         |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.75 APC\_GetPURangeAndStep()

```
int APC_GetPURangeAndStep (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nId,
 int * pMax,
 int * pMin,
 int * pStep,
 int * pDefault,
 int * pFlags)
```

get processing unit property value By v4l2\_queryctrl to get property values of processing unit(PU) this enumeration contained the following properties: V4L2\_CID\_BACKLIGHT\_COMPENSATION V4L2\_CID\_BRIGHTNESS V4L2\_CID\_CONTRAST V4L2\_CID\_GAIN V4L2\_CID\_POWER\_LINE\_FREQUENCY V4L2\_CID\_HUE V4L2\_CID\_HUE\_AUTO V4L2\_CID\_SATURATION V4L2\_CID\_SHARPNESS V4L2\_CID\_GAMMA V4L2\_CID\_WHITE\_BALANCE\_TEMPERATURE V4L2\_CID\_AUTO\_WHITE\_BALANCE

## Parameters

|                    |                                                                                                                                                                                                                                                                                                                                           |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                                                                                                                                                                                                                                                       |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                                                                                                                                                                                                                                                                |
| <i>int</i>         | nId specifies the member of the property set, see CT Property ID defined in <a href="#">eSPDI_def.h</a>                                                                                                                                                                                                                                   |
| <i>long</i>        | int *pMax maximum value, inclusive. This field gives an upper bound for the control                                                                                                                                                                                                                                                       |
| <i>long</i>        | int *pMin minimum value, inclusive. This field gives a lower bound for the control                                                                                                                                                                                                                                                        |
| <i>long</i>        | int *pStep This field gives a step size for the control see enum <a href="https://www.linuxtv.org/downloads/v4l-dvb-apis-old/vidioc-queryctrl.html">https://www.linuxtv.org/downloads/v4l-dvb-apis-old/vidioc-queryctrl.html</a> how the step value is to be used for each possible control type. Note that this an unsigned 32-bit value |
| <i>long</i>        | int *pDefault The default value of a V4L2_CTRL_TYPE_INTEGER, _BOOLEAN, _BITMASK, _MENU or _INTEGER_MENU control. Not valid for other types of controls. Note that drivers reset controls to their default value only when the driver is first loaded, never afterwards.                                                                   |
| <i>long</i>        | int *pFlags control flags, see <a href="https://www.linuxtv.org/downloads/v4l-dvb-apis-old/vidioc-queryctrl.html">https://www.linuxtv.org/downloads/v4l-dvb-apis-old/vidioc-queryctrl.html</a>                                                                                                                                            |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.76 APC\_GetRectifyLogData()**

```
int APC_GetRectifyLogData (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 eSPCtrl_RectLogData * pData,
 int index)
```

get rectify log data from flash, just for AXES1 device type

**Parameters**

|                                     |                                                                                                     |
|-------------------------------------|-----------------------------------------------------------------------------------------------------|
| <i>void</i>                         | *pHandleEYSD handle                                                                                 |
| <i>PDEVSELINFO</i>                  | pDevSelInfo pointer of device select index                                                          |
| <a href="#">eSPCtrl_RectLogData</a> | *pData 4096 bytes of rectify log data, see <a href="#">eSPCtrl_RectLogData</a> for detailed members |
| <i>index,user</i>                   | data section from 0 ~ 9                                                                             |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.77 APC\_GetRectifyMatLogData()**

```
int APC_GetRectifyMatLogData (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 eSPCtrl_RectLogData * pData,
 int index)
```

get rectify log data from flash, just for PUMA device type

**Parameters**

|                                     |                                                                                                     |
|-------------------------------------|-----------------------------------------------------------------------------------------------------|
| <i>void</i>                         | *pHandleEYSD handle                                                                                 |
| <i>PDEVSELINFO</i>                  | pDevSelInfo pointer of device select index                                                          |
| <a href="#">eSPCtrl_RectLogData</a> | *pData 4096 bytes of rectify log data, see <a href="#">eSPCtrl_RectLogData</a> for detailed members |
| <i>index,user</i>                   | data section from 0 ~ 9                                                                             |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.78 APC\_GetRectifyTable()

```
int APC_GetRectifyTable (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * buffer,
 int BufferLength,
 int * pActualLength,
 int index)
```

get rectify values (file ID 40+) from flash

## Parameters

|                    |                                                                           |
|--------------------|---------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                       |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                |
| <i>BYTE</i>        | *buffer buffer to store rectify table data                                |
| <i>int</i>         | BufferLength input buffer length, must be 1024                            |
| <i>int</i>         | *pActualLength actual length has written to buffer                        |
| <i>int</i>         | index index(from 0 ~ 9) to identify rectify table for corresponding depth |

## Returns

success:APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.79 APC\_GetSensorRegister()

```
int APC_GetSensorRegister (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nId,
 unsigned short address,
 unsigned short * pValue,
 int flag,
 SENSORMODE_INFO SensorMode)
```

get value from sensor register

## Parameters

|                        |                                                                                                                                                                                                                                             |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>            | *pHandleEYSD handle                                                                                                                                                                                                                         |
| <i>PDEVSELINFO</i>     | pDevSelInfo pointer of device select index                                                                                                                                                                                                  |
| <i>int</i>             | nId sensor slave address see Videodevice.h for sensor slave address setting                                                                                                                                                                 |
| <i>unsigned</i>        | short address register address                                                                                                                                                                                                              |
| <i>unsigned</i>        | short *pValue pointer of value got from register address                                                                                                                                                                                    |
| <i>int</i>             | flag address and value data length(2 or 1 byte) ie FG_Address_2Byte   FG_Value_2Byte is 2 byte address and 2 byte value #define FG_Address_1Byte 0x01 #define FG_Address_2Byte 0x02 #define FG_Value_1Byte 0x10 #define FG_Value_2Byte 0x20 |
| <i>SENSORMODE_INFO</i> | SensorMode sensor mode(sensor A, B or Both) A is 0, B is 1, Both is 2                                                                                                                                                                       |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.80 APC\_GetSerialNumber()**

```
int APC_GetSerialNumber (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * pData,
 int nbufferSize,
 int * pLen)
```

get device serial number

**Parameters**

|                    |                                                                     |
|--------------------|---------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                 |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                          |
| <i>BYTE</i>        | *pData output buffer to store serial number string                  |
| <i>int</i>         | nbufferSize pData buffer length in byte, 2 byte(WideChar) is a unit |
| <i>int</i>         | *pLen pointer of actual serial number length                        |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.81 APC\_GetSRB()**

```
int APC_GetSRB (
 void * pSrbHandle,
 srb_packet_s * pPacket)
```

Get Packet from SRB.

**Parameters**

|                                 |                                  |
|---------------------------------|----------------------------------|
| <i>void</i>                     | *pSrbHandle pointer to SRB class |
| <i><a href="#">packet_s</a></i> | *pPacket Input Packet            |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.82 APC\_GetThermalFD()

```
int APC_GetThermalFD (
 void * pHandleEYSD,
 int * p_FD)
```

get file description of thermal device

##### Parameters

|             |                                          |
|-------------|------------------------------------------|
| <i>void</i> | *pHandleEYSD handle                      |
| <i>int</i>  | *p_FD file description of thermal device |

##### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.83 APC\_GetUACData()

```
int APC_GetUACData (
 unsigned char * buffer,
 int length)
```

UAC initial function.

##### Parameters

|                 |                                    |
|-----------------|------------------------------------|
| <i>unsigned</i> | char *buffer pointer of UAC buffer |
| <i>int</i>      | length UAC buffer length           |

##### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.84 APC\_getUACNAME()

```
int APC_getUACNAME (
 char * input,
 char * output)
```

Get EYSD UAC Name.

##### Parameters

|             |                                 |
|-------------|---------------------------------|
| <i>char</i> | *input Point to device Address. |
| <i>char</i> | *output Point to device Name.   |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.85 APC\_GetUserData()**

```
int APC_GetUserData (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * buffer,
 int BufferLength,
 USERDATA_SECTION_INDEX usi)
```

get user data from flash

**Parameters**

|                               |                                            |
|-------------------------------|--------------------------------------------|
| <i>void</i>                   | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i>            | pDevSelInfo pointer of device select index |
| <i>BYTE</i>                   | *buffer buffer to store user data          |
| <i>int</i>                    | BufferLength input buffer length           |
| <i>USERDATA_SECTION_INDEX</i> | usi which user index data to select        |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.86 APC\_GetYOffset()**

```
int APC_GetYOffset (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * buffer,
 int BufferLength,
 int * pActualLength,
 int index)
```

get Y offset (file ID 30+) value

**Parameters**

|                    |                                                 |
|--------------------|-------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                             |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index      |
| <i>BYTE</i>        | *buffer buffer to store Y offset values         |
| <i>int</i>         | BufferLength must be 256                        |
| <i>int</i>         | *pActualLength the buffer length, always be 256 |
| <i>int</i>         | index index value to file ID 30                 |



**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.87 APC\_GetZDTable()**

```
int APC_GetZDTable (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * buffer,
 int BufferLength,
 int * pActualLength,
 PZDTABLEINFO pZDTableInfo)
```

get disparity and Z values from flash

**Parameters**

|                     |                                                                               |
|---------------------|-------------------------------------------------------------------------------|
| <i>void</i>         | *pHandleEYSD handle                                                           |
| <i>PDEVSELINFO</i>  | pDevSelInfo pointer of device select index                                    |
| <i>BYTE</i>         | *buffer bufer to store ZD table                                               |
| <i>int</i>          | BufferLength input buffer length                                              |
| <i>int</i>          | *pActualLength actual length has written to buffer                            |
| <i>PZDTABLEINFO</i> | pZDTableInfo index to identify ZD table and data type for corresponding depth |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.88 APC\_HoleFill()**

```
int APC_HoleFill (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * depthBuf,
 int bytesPerPixel,
 int kernel_size,
 int width,
 int height,
 int level,
 bool horizontal)
```

APC\_HoleFill.

**Parameters**

|             |                                                              |
|-------------|--------------------------------------------------------------|
| <i>void</i> | *pHandleEYSD the pointer to the initilized EYSD SDK instance |
|-------------|--------------------------------------------------------------|

## Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>unsigned</i>    | char* depthBuf depth buffer pointer        |
| <i>int</i>         | bytesPerPixel byte number of one pixel     |
| <i>int</i>         | kernel_size [TODO]                         |
| <i>int</i>         | width depth width                          |
| <i>int</i>         | height depth height                        |
| <i>int</i>         | level [TODO]                               |
| <i>bool</i>        | horizontal [TODO]                          |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.89 APC\_HoleFilled()

```
int APC_HoleFilled (
 unsigned short * pDImgIn,
 unsigned short * pDImgOut,
 int width,
 int height,
 int holeFilldiff)
```

Hole Filled.

## Parameters

|                 |                                                          |
|-----------------|----------------------------------------------------------|
| <i>unsigned</i> | short *pDImgIn Image Input                               |
| <i>unsigned</i> | short *pDImgOut Image Output                             |
| <i>int</i>      | width image width                                        |
| <i>int</i>      | height image height                                      |
| <i>int</i>      | holeFilldiff Hole filled strength, value from 0 to 2047. |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.90 APC\_ImgMirro() [1/2]

```
int APC_ImgMirro (
 APCImageType::Value imgType,
 int width,
 int height,
```

```
unsigned char * src,
unsigned char * dstBuf)
```

Make the image to Mirro.

## Parameters

|                            |                                |
|----------------------------|--------------------------------|
| <i>APCImageType::Value</i> | imgType Image Type             |
| <i>int</i>                 | width image width              |
| <i>int</i>                 | height image height            |
| <i>unsigned</i>            | char *src image source         |
| <i>unsigned</i>            | char *dstBuf image desteration |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.91 APC\_ImgMirro() [2/2]

```
int APC_ImgMirro (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 APCImageType::Value imgType,
 int width,
 int height,
 unsigned char * src,
 unsigned char * dstBuf)
```

Make the image to Mirro.

## Parameters

|                            |                                                              |
|----------------------------|--------------------------------------------------------------|
| <i>void</i>                | *pHandleEYSD the pointer to the initilized EYSD SDK instance |
| <i>PDEVSELINFO</i>         | pDevSelInfo pointer of device select index                   |
| <i>APCImageType::Value</i> | imgType Image Type                                           |
| <i>int</i>                 | width image width                                            |
| <i>int</i>                 | height image height                                          |
| <i>unsigned</i>            | char *src image source                                       |
| <i>unsigned</i>            | char *dstBuf image desteration                               |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.92 APC\_Init()

```
int APC_Init (
 void ** ppHandleEYSD,
 bool bIsLogEnabled)
```

entry point of EYSD camera SDK including 1.create a CEYSD class for accessing oncming APIs 2.find out EYSD devices 3.create a CVideoDevice class for video streaming and hardware access

## Parameters

|                       |                                            |
|-----------------------|--------------------------------------------|
| <i>**ppHandleEYSD</i> | a pointer of pointer to access CEYSD class |
| <i>blsLogEnabled</i>  | generate log or not                        |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.93 APC\_InitialCmdFiFo()

```
int APC_InitialCmdFiFo (
 const char * pfifoName,
 int * pFileDescription,
 bool bRead)
```

Cmd FiFo Initial function.

## Parameters

|              |                                                 |
|--------------|-------------------------------------------------|
| <i>const</i> | char *pfifoName Point to the cmd fifo name      |
| <i>int</i>   | *pFileDescription Point to the file description |
| <i>bRead</i> | Indicate Read or Write Cmd fifo                 |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.94 APC\_InitialFlexibleGyro()

```
int APC_InitialFlexibleGyro (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

gyro sensor initial function

## Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.95 APC\_InitialHidGyro()

```
int APC_InitialHidGyro (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

gyro sensor initial function

##### Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |

##### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.96 APC\_InitialUAC()

```
int APC_InitialUAC (
 char * deviceName)
```

UAC initial function.

##### Parameters

|             |                                   |
|-------------|-----------------------------------|
| <i>char</i> | *deviceName Point to device Name. |
|-------------|-----------------------------------|

##### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.97 APC\_InitPostProcess()

```
APC_InitPostProcess (
 void ** ppPostProcessHandle,
 unsigned int nWidth,
 unsigned int nHeight,
 APCImageType::Value imageType)
```

APC\_InitPostProcess.

## Parameters

|                            |                              |
|----------------------------|------------------------------|
| <i>void</i>                | **ppPostProcessHandle [TODO] |
| <i>unsigned</i>            | int nWidth [TODO]            |
| <i>unsigned</i>            | int nHeight [TODO]           |
| <i>APCImageType::Value</i> | imageType [TODO]             |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.98 APC\_InitSRB()

```
int APC_InitSRB (
 void ** pSrbHandle,
 int QueueSize,
 char * queueName)
```

Initial the SRB(Share Ring Buffering)

## Parameters

|             |                                                |
|-------------|------------------------------------------------|
| <i>void</i> | **pSrbHandle a pointer of pointer to SRB class |
| <i>int</i>  | QueueSize                                      |
| <i>char</i> | srbName SRM Name                               |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.99 APC\_InjectExtraDataToMp4()

```
int APC_InjectExtraDataToMp4 (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 const char * filename,
 const char * data,
 int dataLen)
```

APC\_InjectExtraDataToMp4.

## Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>const</i>       | char *filename input video file name       |
| <i>const</i>       | char *data video data                      |
| <i>const</i>       | int dataLen video data length              |

**Returns**

success: APC\_OK, others:see [eSPDI\\_def.h](#)

**9.1.2.100 APC\_IsInterleaveDevice()**

```
bool APC_IsInterleaveDevice (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

check module support interleave function or not

**Parameters**

|                    |                                                 |
|--------------------|-------------------------------------------------|
| <i>pHandleEYSD</i> | the pointer to the initilized EYSD SDK instance |
| <i>pDevSelInfo</i> | pointer of device select index                  |

**Returns**

true: support interleave, false: not support

**9.1.2.101 APC\_IsMLBaseLine()**

```
bool APC_IsMLBaseLine (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

Check the device is multiple baseline device.

**Parameters**

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |

**Returns**

true: multiplies baseline device, false: normally device.

**9.1.2.102 APC\_OpenDevice()**

```
int APC_OpenDevice (
 void * pHandleEYSD,
```



```

PDEVSELINFO pDevSelInfo,
int nEP0Width,
int nEP0Height,
bool bEP0MJPG,
int nEP1Width,
int nEP1Height,
DEPTH_TRANSFER_CTRL dtc = DEPTH_IMG_NON_TRANSFER,
bool bIsOutputRGB24 = false,
void * phWndNotice = 0,
int * pFPS = 0,
CONTROL_MODE cm = IMAGE_SN_NONSYNC)

```

the implement layer to open EYSD camera device by V4L2(<https://en.wikipedia.org/wiki/V4L2>), can open color and depth at one time call, do functions as below,

1. initialize the USB device by V4L2 protocol 1.1 query device v4l2 capability 1.2 must have video capability 1.3 must have streaming capability 1.4 issue resolution mode to UVC driver and check result 1.5 initialize memory buffer mapping from kernel to user mode
2. enumerate frame interval to set frame rate
3. start video capture processes

#### Parameters

|                    |                                                  |
|--------------------|--------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                              |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index       |
| <i>int</i>         | nEP0Width width of endpoint1(color) resolution   |
| <i>int</i>         | nEP0Height height of endpoint1(color) resolution |
| <i>bool</i>        | bEP0MJPG endpoint1 output is MJPEG ?             |
| <i>int</i>         | *pFPS input frame rate setting                   |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.103 APC\_OpenDevice2()

```

int APC_OpenDevice2 (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nEP0Width,
 int nEP0Height,
 bool bEP0MJPG,
 int nEP1Width,
 int nEP1Height,
 DEPTH_TRANSFER_CTRL dtc = DEPTH_IMG_NON_TRANSFER,
 bool bIsOutputRGB24 = false,
 void * phWndNotice = 0,
 int * pFPS = 0,
 CONTROL_MODE cm = IMAGE_SN_NONSYNC)

```

the implement layer to open EYSD camera device by V4L2(<https://en.wikipedia.org/wiki/V4L2>), can open color and depth at one time call, do functions as below,

1. initialize the USB device by V4L2 protocol 1.1 query device v4l2 capability 1.2 must have video capability 1.3 must have streaming capability 1.4 issue resolution mode to UVC driver and check result 1.5 initialize memory buffer mapping from kernel to user mode
2. enumerate frame interval to set frame rate
3. start video capture processes

#### Parameters

|                            |                                                  |
|----------------------------|--------------------------------------------------|
| <i>void</i>                | *pHandleEYSD handle                              |
| <i>PDEVSELINFO</i>         | pDevSelInfo pointer of device select index       |
| <i>int</i>                 | nEP0Width width of endpoint1(color) resolution   |
| <i>int</i>                 | nEP0Height height of endpoint1(color) resolution |
| <i>bool</i>                | bEP0MJPEG endpoint1 output is MJPEG ?            |
| <i>int</i>                 | nEP1Width width of endpoint2(depth) resolution   |
| <i>int</i>                 | nEP1Height height of endpoint2(depth) resolution |
| <i>DEPTH_TRANSFER_CTRL</i> | dtc depth image output transfer                  |

1. default is transferred to color(DEPTH\_IMG\_COLORFUL\_TRANSFER) by calling from [APC\\_OpenDevice\(\)](#)
2. DEPTH\_IMG\_GRAY\_TRANSFER : transfer to gray
3. DEPTH\_IMG\_NON\_TRANSFER : no transfer

#### Parameters

|                     |                                                 |
|---------------------|-------------------------------------------------|
| <i>bool</i>         | bIsOutputRGB24 output color image is RGB format |
| <i>void</i>         | *phWndNotice reserved, not use                  |
| <i>int</i>          | *pFPS input frame rate setting                  |
| <i>CONTROL_MODE</i> | cm reserved, not use                            |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.104 APC\_OpenDeviceMBL()

```
int APC_OpenDeviceMBL (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nEP0Width,
 int nEP0Height,
 bool bEP0MJPEG,
 int nEP1Width,
 int nEP1Height,
 DEPTH_TRANSFER_CTRL dtc = DEPTH_IMG_NON_TRANSFER,
 bool bIsOutputRGB24 = false,
 void * phWndNotice = 0,
 int * pFPS = 0,
 CONTROL_MODE cm = IMAGE_SN_NONSYNC)
```

the implement layer to open Multiple Base Line EYSD camera device by V4L2(<https://en.wikipedia.org/wiki/Video4Linux>), can open color and depth at one time call, do functions as below,

1. initialize the USB device by V4L2 protocol 1.1 query device v4l2 capability 1.2 must have video capability 1.3 must have streaming capability 1.4 issue resolution mode to UVC driver and check result 1.5 initialize memory buffer mapping from kernel to user mode
2. enumerate frame interval to set frame rate
3. start video capture processes

#### Parameters

|                            |                                                  |
|----------------------------|--------------------------------------------------|
| <i>void</i>                | *pHandleEYSD handle                              |
| <i>PDEVSELINFO</i>         | pDevSelInfo pointer of device select index       |
| <i>int</i>                 | nEP0Width width of endpoint1(color) resolution   |
| <i>int</i>                 | nEP0Height height of endpoint1(color) resolution |
| <i>bool</i>                | bEP0MJPG endpoint1 output is MJPEG ?             |
| <i>int</i>                 | nEP1Width width of endpoint2(depth) resolution   |
| <i>int</i>                 | nEP1Height height of endpoint2(depth) resolution |
| <i>DEPTH_TRANSFER_CTRL</i> | dtc depth image output transfer                  |

1. default is transferred to color(DEPTH\_IMG\_COLORFUL\_TRANSFER) by calling from [APC\\_OpenDevice\(\)](#)
2. DEPTH\_IMG\_GRAY\_TRANSFER : transfer to gray
3. DEPTH\_IMG\_NON\_TRANSFER : no transfer

#### Parameters

|                     |                                                 |
|---------------------|-------------------------------------------------|
| <i>bool</i>         | bIsOutputRGB24 output color image is RGB format |
| <i>void</i>         | *pHwndNotice reserved, not use                  |
| <i>int</i>          | *pFPS input frame rate setting                  |
| <i>CONTROL_MODE</i> | cm reserved, not use                            |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.105 APC\_PostProcess()

```
APC_PostProcess (
 void * pPostProcessHandle,
 unsigned char * pDepthData)
```

APC\_PostProcess.

#### Parameters

|                 |                             |
|-----------------|-----------------------------|
| <i>void</i>     | *ppPostProcessHandle [TODO] |
| <i>unsigned</i> | char *pDepthData [TODO]     |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.106 APC\_PutSRB()**

```
int APC_PutSRB (
 void * pSrbHandle,
 srb_packet_s * pPacket)
```

Put Packet to SRB.

**Parameters**

|                          |                                  |
|--------------------------|----------------------------------|
| <i>void</i>              | *pSrbHandle pointer to SRB class |
| <a href="#">packet_s</a> | *pPacket Input Packet            |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.107 APC\_ReadCmdFiFo()**

```
APC_ReadCmdFiFo (
 int FileDescription,
 unsigned char * pBuf,
 int len)
```

Read Cmd FiFo function.

**Parameters**

|                 |                                    |
|-----------------|------------------------------------|
| <i>int</i>      | FileDescription File description   |
| <i>unsigned</i> | char *pCmd Point to the cmd buffer |
| <i>int</i>      | lenIndicate the cmd length.        |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.108 APC\_ReadFlashData()**

```
int APC_ReadFlashData (
 void * pHandleEYSD,
```

```

PDEVSELINFO pDevSelInfo,
FLASH_DATA_TYPE fdt,
BYTE * pBuffer,
unsigned long int BufferLength,
unsigned long int * pActualLength)

```

read firmware code(.bin) form flash The firmware code is the combination of boot loader, firmware body and plug-in data. This input buffer length has to match with the flash data type

#### Parameters

|                        |                                                              |
|------------------------|--------------------------------------------------------------|
| <i>void</i>            | *pHandleEYSD handle                                          |
| <i>PDEVSELINFO</i>     | pDevSelInfo pointer of device select index                   |
| <i>FLASH_DATA_TYPE</i> | fdt segment type of flash be read                            |
| <i>BYTE</i>            | *pBuffer buffer to store firmware code                       |
| <i>unsigned</i>        | long int BufferLength input buffer length                    |
| <i>unsigned</i>        | long int *pActualLength actual length has written to pBuffer |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.109 APC\_RefreshDevice()

```

int APC_RefreshDevice (
 void * pHandleEYSD)

```

refresh all EYSD UVC devices

#### Parameters

|             |                     |
|-------------|---------------------|
| <i>void</i> | *pHandleEYSD handle |
|-------------|---------------------|

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.110 APC\_Release()

```

void APC_Release (
 void ** ppHandleEYSD)

```

release resource that APC\_Init had allocated

**Parameters**

|             |                                              |
|-------------|----------------------------------------------|
| <i>void</i> | **ppHandleEYSD array of CEYSD class handlers |
|-------------|----------------------------------------------|

**Returns**

none

**9.1.2.111 APC\_ReleaseFlexibleGyro()**

```
int APC_ReleaseFlexibleGyro (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

gyro sensor release function

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.112 APC\_ReleaseHidGyro()**

```
int APC_ReleaseHidGyro (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

gyro sensor release function

**Parameters**

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.113 APC\_ReleasePostProcess()**

```
APC_ReleasePostProcess (
 void * pPostProcessHandle)
```

APC\_ReleasePostProcess.

## Parameters

|             |                                    |
|-------------|------------------------------------|
| <i>void</i> | <i>*ppPostProcessHandle</i> [TODO] |
|-------------|------------------------------------|

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.114 APC\_ReleaseSwPostProc()**

```
int APC_ReleaseSwPostProc (
 void ** handle)
```

release a software post process class

## Parameters

|               |                                                           |
|---------------|-----------------------------------------------------------|
| <i>void**</i> | handle handle pointer to this software post process class |
|---------------|-----------------------------------------------------------|

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.115 APC\_ReleaseUAC()**

```
int APC_ReleaseUAC (
 void)
```

UAC initial function.

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.116 APC\_ResetFilters()**

```
int APC_ResetFilters (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo)
```

APC\_ResetFilters.

## Parameters

|                    |                                                              |
|--------------------|--------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD the pointer to the initilized EYSD SDK instance |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                   |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.117 APC\_ResizeImgToHalf()

```
int APC_ResizeImgToHalf (
 APCImageType::Value imgType,
 int width,
 int height,
 unsigned char * src,
 unsigned char * dst,
 int len)
```

Resize the image to half.

## Parameters

|                            |                               |
|----------------------------|-------------------------------|
| <i>APCImageType::Value</i> | mgType Image Type             |
| <i>int</i>                 | width image width             |
| <i>int</i>                 | height image height           |
| <i>unsigned</i>            | char *src image source        |
| <i>unsigned</i>            | char *dst image desteration   |
| <i>int</i>                 | len desteration buffer length |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.118 APC\_RetrieveExtraDataFromMp4()

```
int APC_RetrieveExtraDataFromMp4 (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 const char * filename,
 char * data,
 int * dataLen)
```

APC\_RetrieveExtraDataFromMp4.



## Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>const</i>       | char *filename input video file name       |
| <i>const</i>       | char *data video data                      |
| <i>const</i>       | int dataLen video data length              |

## Returns

success: APC\_OK, others:see [eSPDI\\_def.h](#)

## 9.1.2.119 APC\_RGB2BMP()

```
int APC_RGB2BMP (
 char * filename,
 int width,
 int height,
 unsigned char * data)
```

RGB to BMP.

## Parameters

|                  |                     |
|------------------|---------------------|
| <i>*filename</i> | Ouput BMP file name |
| <i>int</i>       | width image width   |
| <i>int</i>       | height image height |
| <i>*data</i>     | input RGB buffer.   |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.120 APC\_RotateImg180() [1/2]

```
int APC_RotateImg180 (
 APCImageType::Value imgType,
 int width,
 int height,
 unsigned char * src,
 unsigned char * dstBuf,
 int len)
```

Rotate the image to 180 degree.

## Parameters

|                            |                                |
|----------------------------|--------------------------------|
| <i>APCImageType::Value</i> | mgType Image Type              |
| <i>int</i>                 | width image width              |
| <i>int</i>                 | height image height            |
| <i>unsigned</i>            | char *src image source         |
| <i>unsigned</i>            | char *dstBuf image desteration |
| <i>int</i>                 | len desteration buffer length  |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

9.1.2.121 **APC\_RotateImg180()** [2/2]

```
int APC_RotateImg180 (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 APCImageType::Value imgType,
 int width,
 int height,
 unsigned char * src,
 unsigned char * dstBuf,
 int len)
```

Rotate the image to 180 degree.

## Parameters

|                            |                                                               |
|----------------------------|---------------------------------------------------------------|
| <i>void</i>                | * pHandleEYSD the pointer to the initilized EYSD SDK instance |
| <i>PDEVSELINFO</i>         | pDevSelInfo pointer of device select index                    |
| <i>APCImageType::Value</i> | mgType Image Type                                             |
| <i>int</i>                 | width image width                                             |
| <i>int</i>                 | height image height                                           |
| <i>unsigned</i>            | char *src image source                                        |
| <i>unsigned</i>            | char *dstBuf image desteration                                |
| <i>int</i>                 | len desteration buffer length                                 |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

9.1.2.122 **APC\_RotateImg90()** [1/2]

```
int APC_RotateImg90 (
 APCImageType::Value imgType,
```

```

 int width,
 int height,
 unsigned char * src,
 unsigned char * dstBuf,
 int len,
 bool clockwise)

```

Rotate the image to 90 degree.

#### Parameters

|                            |                                |
|----------------------------|--------------------------------|
| <i>APCImageType::Value</i> | mgType Image Type              |
| <i>int</i>                 | width image width              |
| <i>int</i>                 | height image height            |
| <i>unsigned</i>            | char *src image source         |
| <i>unsigned</i>            | char *dstBuf image desteration |
| <i>int</i>                 | len desteration buffer length  |
| <i>bClockwise, false</i>   | not supported.                 |
| <i>bOpencv</i>             | useage, not supported.         |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.123 APC\_RotateImg90() [2/2]

```

int APC_RotateImg90 (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 APCImageType::Value imgType,
 int width,
 int height,
 unsigned char * src,
 unsigned char * dstBuf,
 int len,
 bool clockwise)

```

Make the image to rotate.

#### Parameters

|                            |                                                               |
|----------------------------|---------------------------------------------------------------|
| <i>void</i>                | * pHandleEYSD the pointer to the initilized EYSD SDK instance |
| <i>PDEVSELINFO</i>         | pDevSelInfo pointer of device select index                    |
| <i>APCImageType::Value</i> | imgType Image Type                                            |
| <i>int</i>                 | width image width                                             |
| <i>int</i>                 | height image height                                           |
| <i>unsigned</i>            | char *src image source                                        |
| <i>unsigned</i>            | char *dstBuf image desteration                                |
| <i>bool</i>                | clockwise clockwise rotate or not                             |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.124 APC\_SaveLutData()**

```
int APC_SaveLutData (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 const char * filename)
```

Save LUT parameters in the specified file.

**Parameters**

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>const</i>       | char* filename output LUT file name        |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.125 APC\_SelectDevice()**

```
int APC_SelectDevice (
 void * pHandleEYSD,
 int dev_index)
```

do not support currently

**Returns**

APC\_NotSupport

**9.1.2.126 APC\_SetAutoExposureMode()**

```
int APC_SetAutoExposureMode (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short mode)
```

Setup Auto Exposure Mode.

## Parameters

|                    |                                                                                                   |
|--------------------|---------------------------------------------------------------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle.                                                                               |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index.                                                       |
| <i>unsigned</i>    | short mode The setup mode value. 0: Average, 1: Left (or Front) camera, 2: Right (or Back) camera |

## Returns

success: APC\_OK, others: [eSPDI\\_def.h](#)

## 9.1.2.127 APC\_SetColorGain()

```
int APC_SetColorGain (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nSensorMode,
 float fGainR,
 float fGainG,
 float fGainB)
```

set color gain of ISP

## Parameters

|                    |                                                                                |
|--------------------|--------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                            |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                     |
| <i>int</i>         | nSensorMode which sensor(sensor A, B or Both) to get A is 0, B is 1, Both is 2 |
| <i>float</i>       | fGainR Red channel color gain value                                            |
| <i>float</i>       | fGainG Green channel color gain value                                          |
| <i>float</i>       | fGainB Blue channel color gain value                                           |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.128 APC\_SetControlCounterMode()

```
int APC_SetControlCounterMode (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char nValue)
```

enable or disable interleave function

## Parameters

|                    |                                                 |
|--------------------|-------------------------------------------------|
| <i>pHandleEYSD</i> | the pointer to the initilized EYSD SDK instance |
| <i>pDevSelInfo</i> | pointer of device select index                  |
| <i>nValue</i>      | 0: Frame Counter Mode, 1: Serial Counter Mode,  |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.129 APC\_SetCTPropVal()

```
int APC_SetCTPropVal (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nId,
 long int nValue)
```

set camera terminal property values By v4l2\_control to set

## Parameters

|                    |                                                                                                        |
|--------------------|--------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                    |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                             |
| <i>int</i>         | nId specifies the member of the property set see CT Property ID defined in <a href="#">eSPDI_def.h</a> |
| <i>long</i>        | int nValue CT property value to set                                                                    |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.130 APC\_SetCurrentIRValue()

```
t APC_SetCurrentIRValue (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short nValue)
```

set infrared radiation(IR) value of PUMA type IC

## Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>unsigned</i>    | short nValue 1 byte IR value to set        |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.131 APC\_SetDepthDataType()

```
APC_SetDepthDataType (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short nValue)
```

set depth data type, 11 bit for disparity data, 14 bit for Z data notice: only PUMA type IC can support this setting

## Parameters

|                    |                                                                                                     |
|--------------------|-----------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                 |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                          |
| <i>unsigned</i>    | short nValue depth data type you want to set, see APC_DEPTH_DATA_xxx in <a href="#">eSPDI_def.h</a> |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.132 APC\_SetExposureTime()

```
int APC_SetExposureTime (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nSensorMode,
 float fExpTimeMS)
```

set exposure time of ISP sensor setting the target sensor type was set in [APC\\_SetSensorTypeName\(\)](#)

APC\_SetExposureTime( void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, int nSensorMode, float fExpTimeMS)

## Parameters

|                    |                                                                                                                                                                   |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                                                                               |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                                                                                        |
| <i>int</i>         | nSensorMode which sensor(sensor A, B or Both) to set A is 0, B is 1, Both is 2                                                                                    |
| <i>float</i>       | fExpTimeMS pointer of setting exposure time in millisecond check sensor spec for detailed setting, we need pixel clock, pixel per line, V blank and exposure line |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.133 APC\_SetFWRegister()**

```
int APC_SetFWRegister (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short address,
 unsigned short nValue,
 int flag)
```

set firmware register value

**Parameters**

|                    |                                                                                                                                                                                                                                             |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                                                                                                                                                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                                                                                                                                                                  |
| <i>unsigned</i>    | short address register address                                                                                                                                                                                                              |
| <i>unsigned</i>    | short nValue register value to set                                                                                                                                                                                                          |
| <i>int</i>         | flag address and value data length(2 or 1 byte) ie FG_Address_1Byte   FG_Value_1Byte is 1 byte address and 1 byte value #define FG_Address_1Byte 0x01 #define FG_Address_2Byte 0x02 #define FG_Value_1Byte 0x10 #define FG_Value_2Byte 0x20 |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.134 APC\_SetGlobalGain()**

```
int APC_SetGlobalGain (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nSensorMode,
 float fGlobalGain)
```

set global gain of ISP sensor setting the target sensor type was set in [APC\\_SetSensorTypeName\(\)](#)

**Parameters**

|                    |                                                                                |
|--------------------|--------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                            |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                     |
| <i>int</i>         | nSensorMode which sensor(sensor A, B or Both) to get A is 0, B is 1, Both is 2 |
| <i>float</i>       | fGlobalGain pointer of global gain value                                       |



## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.135 APC\_SetHWRegister()

```
int APC_SetHWRegister (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short address,
 unsigned short nValue,
 int flag)
```

set hardware register

## Parameters

|                    |                                                                                                                                                                                                                                             |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                                                                                                                                                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                                                                                                                                                                  |
| <i>unsigned</i>    | short address register address                                                                                                                                                                                                              |
| <i>unsigned</i>    | short nValue register value to set                                                                                                                                                                                                          |
| <i>int</i>         | flag address and value data length(2 or 1 byte) ie FG_Address_1Byte   FG_Value_1Byte is 1 byte address and 1 byte value #define FG_Address_1Byte 0x01 #define FG_Address_2Byte 0x02 #define FG_Value_1Byte 0x10 #define FG_Value_2Byte 0x20 |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.136 APC\_SetInterleaveMode()

```
APC_SetInterleaveMode (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 bool enable)
```

set depth data type, 11 bit for disparity data, 14 bit for Z data notice: only PUMA type IC can support this setting

## Parameters

|                    |                                                                                             |
|--------------------|---------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                  |
| <i>bool</i>        | enable enable/disable interleave mode see APC_DEPTH_DATA_xxx in <a href="#">eSPDI_def.h</a> |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.137 APC\_SetIRMaxValue()**

```
int APC_SetIRMaxValue (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short nValue)
```

get maximum IR value of camera module

**Parameters**

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>unsigned</i>    | short nValue the IR maximum setting value  |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.138 APC\_SetIRMode()**

```
APC_SetIRMode (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short nValue)
```

enable or disable IRs

**Parameters**

|                    |                                                                                                                                                                                                                                     |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                                                                                                                                                 |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                                                                                                                                                          |
| <i>unsigned</i>    | short nValue 8 bit definition as below to turn on/off IR D[7:4]: Reserved D3: Channel 3 D2: Channel 2 D1: Channel 1 D0: Channel 0 1: Enable Channel 0: Disable Channel If want to control ch0 and ch1, ubMode[3:0] must set to 0x03 |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.139 APC\_SetLogData()

```
int APC_SetLogData (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * buffer,
 int BufferLength,
 int * pActualLength,
 int index)
```

set log data to flash

## Parameters

|                    |                                                          |
|--------------------|----------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                      |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index               |
| <i>BYTE</i>        | *buffer log data to set                                  |
| <i>int</i>         | BufferLength buffer length, must be 4096                 |
| <i>int</i>         | *pActualLength always return 4096                        |
| <i>int</i>         | index index to identify log data for corresponding depth |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.140 APC\_SetMultiBytesHWRegister()

```
int APC_SetMultiBytesHWRegister (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short address,
 unsigned char * Data,
 int size,
 int flag)
```

set hardware register

## Parameters

|                    |                                                                                                                                                                                                                                             |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                                                                                                                                                         |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                                                                                                                                                                  |
| <i>unsigned</i>    | short address register address                                                                                                                                                                                                              |
| <i>unsigned</i>    | char *Data multiple-bytes register value to set                                                                                                                                                                                             |
| <i>int</i>         | size multiple-bytes register size                                                                                                                                                                                                           |
| <i>int</i>         | flag address and value data length(2 or 1 byte) ie FG_Address_1Byte   FG_Value_1Byte is 1 byte address and 1 byte value #define FG_Address_1Byte 0x01 #define FG_Address_2Byte 0x02 #define FG_Value_1Byte 0x10 #define FG_Value_2Byte 0x20 |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.141 APC\_SetPidVid()**

```
int APC_SetPidVid (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned short * pPidBuf,
 unsigned short * pVidBuf)
```

set PID and VID to device

**Parameters**

|                    |                                               |
|--------------------|-----------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                           |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index    |
| <i>unsigned</i>    | short *pPidBuf 4 byte PID value buffer to set |
| <i>unsigned</i>    | short *pVidBuf 4 byte VID value buffer to set |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.142 APC\_SetPUPPropVal()**

```
int APC_SetPUPPropVal (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nId,
 long int nValue)
```

set processing unit property value by v4l2\_control to set processing unit(PU) property value

**Parameters**

|                    |                                                                                                        |
|--------------------|--------------------------------------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                                                    |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                                             |
| <i>int</i>         | nId specifies the member of the property set see PU Property ID defined in <a href="#">eSPDI_def.h</a> |
| <i>int</i>         | nValue value to set                                                                                    |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.143 APC\_SetRectifyTable()

```
int APC_SetRectifyTable (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * buffer,
 int BufferLength,
 int * pActualLength,
 int index)
```

set rectify values to flash

## Parameters

|                    |                                                                           |
|--------------------|---------------------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                                       |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                                |
| <i>BYTE</i>        | *buffer rectify values to set                                             |
| <i>int</i>         | BufferLength bufer length, must be 1024                                   |
| <i>int</i>         | *pActualLength always return 1024                                         |
| <i>int</i>         | index index(from 0 ~ 9) to identify rectify table for corresponding depth |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.144 APC\_SetSensorRegister()

```
int APC_SetSensorRegister (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int nId,
 unsigned short address,
 unsigned short nValue,
 int flag,
 SENSORMODE_INFO SensorMode)
```

set sensor register value

## Parameters

|                        |                                                                                                                                                                                                                                             |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>            | *pHandleEYSD handle                                                                                                                                                                                                                         |
| <i>PDEVSELINFO</i>     | pDevSelInfo pointer of device select index                                                                                                                                                                                                  |
| <i>int</i>             | nId sensor slave address see Videodevice.h for sensor slave address setting                                                                                                                                                                 |
| <i>unsigned</i>        | short address register address                                                                                                                                                                                                              |
| <i>unsigned</i>        | short nValue value to set                                                                                                                                                                                                                   |
| <i>int</i>             | flag address and value data length(2 or 1 byte) ie FG_Address_1Byte   FG_Value_1Byte is 1 byte address and 1 byte value #define FG_Address_1Byte 0x01 #define FG_Address_2Byte 0x02 #define FG_Value_1Byte 0x10 #define FG_Value_2Byte 0x20 |
| <i>SENSORMODE_INFO</i> | SensorMode sensor mode(sensor A, B or Both) A is 0, B is 1, Both is 2                                                                                                                                                                       |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.145 APC\_SetSensorTypeName()**

```
int APC_SetSensorTypeName (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 SENSOR_TYPE_NAME stn)
```

set the sensor type you want to work on

**Parameters**

|                         |                                            |
|-------------------------|--------------------------------------------|
| <i>void</i>             | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i>      | pDevSelInfo pointer of device select index |
| <i>SENSOR_TYPE_NAME</i> | stn which sensor you want to work on       |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.146 APC\_SetSerialNumber()**

```
int APC_SetSerialNumber (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * pData,
 int nLen)
```

set serial number to device

**Parameters**

|                    |                                                                 |
|--------------------|-----------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                                             |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                      |
| <i>BYTE</i>        | *pData pointer of buffer to store serial number, it is WildChar |
| <i>int</i>         | nLen pData length in byte                                       |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.147 APC\_Setup\_v4l2\_requestbuffers()

```
int APC_Setup_v4l2_requestbuffers (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int cnt)
```

Setup v4l2 request buffers, default = 4.

##### Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>int</i>         | cnt Should be >= 0                         |

##### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.148 APC\_SetupBlock()

```
int APC_SetupBlock (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 bool enable)
```

get color or depth pin image by issuing V4L2's IOCTL to get frame data

##### Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>bool</i>        | enable Enable the Blocking mode or not)    |

##### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.149 APC\_SetupHidGyro()

```
int APC_SetupHidGyro (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * pCmdBuf,
 int cmdlength)
```

getting gyro data function

## Parameters

|                    |                                              |
|--------------------|----------------------------------------------|
| <i>void*</i>       | pHandleEYSD handle                           |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index   |
| <i>unsigned</i>    | char *pGyroData pointer of Gyro Data Buffer. |
| <i>int</i>         | length Input buffer Length, shoul            |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.150 APC\_SetUserData()

```
int APC_SetUserData (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * buffer,
 int BufferLength,
 USERDATA_SECTION_INDEX usi)
```

set user data to flash

## Parameters

|                               |                                            |
|-------------------------------|--------------------------------------------|
| <i>void</i>                   | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i>            | pDevSelInfo pointer of device select index |
| <i>BYTE</i>                   | *buffer user buffer data to set            |
| <i>int</i>                    | BufferLength buffer length to write        |
| <i>USERDATA_SECTION_INDEX</i> | usi which user section data to set         |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.151 APC\_SetYOffset()

```
int APC_SetYOffset (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * buffer,
 int BufferLength,
 int * pActualLength,
 int index)
```

set Y offset values



## Parameters

|                    |                                            |
|--------------------|--------------------------------------------|
| <i>void</i>        | *pHandleEYSD handle                        |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index |
| <i>BYTE</i>        | *buffer buffer data to set                 |
| <i>int</i>         | BufferLength buffer length                 |
| <i>int</i>         | *pActualLength always return 256           |
| <i>int</i>         | index index value to file ID 30            |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.152 APC\_SetZDTable()

```
int APC_SetZDTable (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 BYTE * buffer,
 int BufferLength,
 int * pActualLength,
 PZDTABLEINFO pZDTableInfo)
```

set disparity and Z values to flash

## Parameters

|                     |                                                                               |
|---------------------|-------------------------------------------------------------------------------|
| <i>void</i>         | *pHandleEYSD handle                                                           |
| <i>PDEVSELINFO</i>  | pDevSelInfo pointer of device select index                                    |
| <i>BYTE</i>         | *buffer ZD values to set                                                      |
| <i>int</i>          | BufferLength corresponding length of ZD table in buffer                       |
| <i>int</i>          | *pActualLength buffer length written to flash, should be same as BufferLength |
| <i>PZDTABLEINFO</i> | pZDTableInfo index and depth type of this ZD                                  |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.153 APC\_SubSample()

```
int APC_SubSample (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char ** SubSample,
```

```

 unsigned char * depthBuf,
 int bytesPerPixel,
 int width,
 int height,
 int & new_width,
 int & new_height,
 int mode = 0,
 int factor = 3)

```

APC\_SubSample.

#### Parameters

|                    |                                                              |
|--------------------|--------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD the pointer to the initilized EYSD SDK instance |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                   |
| <i>unsigned</i>    | char **SubSample [TODO]                                      |
| <i>unsigned</i>    | char *depthBuf depth buffer pointer                          |
| <i>int</i>         | bytesPerPixel byte number of one pixel                       |
| <i>int</i>         | width depth width                                            |
| <i>int</i>         | height depth height                                          |
| <i>int&amp;</i>    | new_width new depth width                                    |
| <i>int&amp;</i>    | new_height new depth height                                  |
| <i>int</i>         | mode [TODO]                                                  |
| <i>int</i>         | factor [TODO]                                                |

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

#### 9.1.2.154 APC\_SwitchBaseline()

```

int APC_SwitchBaseline (
 int index)

```

Swich the baseline index.

#### Parameters

|            |                                                  |
|------------|--------------------------------------------------|
| <i>int</i> | index Baseline index 1: 30 mm 2: 60 mm 3: 150 mm |
|------------|--------------------------------------------------|

#### Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.155 APC\_TableToData()

```
int APC_TableToData (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 int width,
 int height,
 int TableSize,
 unsigned short * Table,
 unsigned short * Src,
 unsigned short * Dst)
```

transfer Src to Dst by Table

## Parameters

|                    |                                                              |
|--------------------|--------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD the pointer to the initilized EYSD SDK instance |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                   |
| <i>int</i>         | width input image width                                      |
| <i>int</i>         | height input image height                                    |
| <i>int</i>         | TableSize input Table size in bytes                          |
| <i>unsigned</i>    | short *Table input Table buffer                              |
| <i>unsigned</i>    | short *Src input Src buffer                                  |
| <i>unsigned</i>    | short *Dst output Dst buffer                                 |

## Returns

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.1.2.156 APC\_TemporalFilter()

```
int APC_TemporalFilter (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 unsigned char * depthBuf,
 int bytesPerPixel,
 int width,
 int height,
 float alpha,
 int history)
```

APC\_TemporalFilter.

## Parameters

|                    |                                                              |
|--------------------|--------------------------------------------------------------|
| <i>void</i>        | *pHandleEYSD the pointer to the initilized EYSD SDK instance |
| <i>PDEVSELINFO</i> | pDevSelInfo pointer of device select index                   |
| <i>unsigned</i>    | char* depthBuf depth buffer pointer                          |
| <i>int</i>         | bytesPerPixel byte number of one pixel                       |
| <i>int</i>         | width depth width                                            |
| <i>int</i>         | height depth height                                          |
| <i>float</i>       | alpha [TODO]                                                 |
| <i>int</i>         | history [TODO]                                               |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.157 APC\_WriteCmdFiFo()**

```
int APC_WriteCmdFiFo (
 int FileDescription,
 unsigned char * pCmd,
 int len)
```

Write Cmd FiFo function.

**Parameters**

|                 |                                    |
|-----------------|------------------------------------|
| <i>int</i>      | FileDescription File description   |
| <i>unsigned</i> | char *pCmd Point to the cmd buffer |
| <i>int</i>      | lenIndicate the cmd length.        |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.158 APC\_WriteFlashData()**

```
int APC_WriteFlashData (
 void * pHandleEYSD,
 PDEVSELINFO pDevSelInfo,
 FLASH_DATA_TYPE fdt,
 BYTE * pBuffer,
 unsigned long int BufferLength,
 bool bIsDataVerify,
 KEEP_DATA_CTRL kdc)
```

write firmware code(.bin) to flash The firmware code is the combination of boot loader, firmware body and plug-in data, also can keep original functions(Serial Number, Sensor Position, RectificationTable, ZD Table and CalibrationLog) on camera flash by KEEP\_DATA\_CTRL control

**Parameters**

|                        |                                                                                                                         |
|------------------------|-------------------------------------------------------------------------------------------------------------------------|
| <i>void</i>            | *pHandleEYSD CEronDI class                                                                                              |
| <i>PDEVSELINFO</i>     | pDevSelInfo pointer of device select index                                                                              |
| <i>FLASH_DATA_TYPE</i> | fdt segment type of flash be wrote                                                                                      |
| <i>BYTE</i>            | *pBuffer buffer of firmware code                                                                                        |
| <i>unsigned</i>        | long int BufferLength Buffer length to be wrote                                                                         |
| <i>BOOL</i>            | blsDataVerify write data verification flag, if true this function will read data again and do a byte to byte comparison |
| <i>KEEP_DATA_CTRL</i>  | kdc keep function flags                                                                                                 |

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.159 APC\_WriteWaveEnd()**

```
int APC_WriteWaveEnd (
 int fd,
 size_t length)
```

Modified Wave Header.

**Parameters**

|            |                        |
|------------|------------------------|
| <i>int</i> | fd wave file descript. |
|------------|------------------------|

**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

**9.1.2.160 APC\_WriteWaveHeader()**

```
int APC_WriteWaveHeader (
 int fd)
```

Write Wave Header.

**Parameters**

|            |                        |
|------------|------------------------|
| <i>int</i> | fd wave file descript. |
|------------|------------------------|

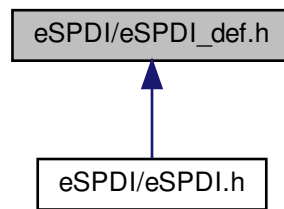
**Returns**

success: APC\_OK, others: see [eSPDI\\_def.h](#)

## 9.2 eSPDI/eSPDI\_def.h File Reference

error/data type definitions

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



## Classes

- struct [packet\\_s](#)
- struct [tagDEVINFORMATION](#)
- struct [tagDEVSEL](#)
- struct [tagAPC\\_STREAM\\_INFO](#)
- struct [tagZDTableInfo](#)
- struct [tagKEEP\\_DATA\\_CTRL](#)
- struct [eSPCtrl\\_RectLogData](#)
- struct [GyroTag](#)
- struct [AccelerationTag](#)
- struct [CompassTag](#)
- struct [APCImageType](#)
- struct [PointCloudInfo](#)

## Macros

- #define **APC\_OK** 0
- #define **APC\_NoDevice** -1
- #define **APC\_NullPtr** -2
- #define **APC\_ErrBufLen** -3
- #define **APC\_Init\_Fail** -4
- #define **APC\_NoZDTable** -5
- #define **APC\_READFLASHFAIL** -6
- #define **APC\_WRITEFLASHFAIL** -7
- #define **APC\_VERIFY\_DATA\_FAIL** -8
- #define **APC\_KEEP\_DATA\_FAIL** -9
- #define **APC\_RECT\_DATA\_LEN\_FAIL** -10
- #define **APC\_RECT\_DATA\_PARSING\_FAIL** -11
- #define **APC\_RET\_BAD\_PARAM** -12
- #define **APC\_RET\_OPEN\_FILE\_FAIL** -13
- #define **APC\_NO\_CALIBRATION\_LOG** -14
- #define **APC\_POSTPROCESS\_INIT\_FAIL** -15
- #define **APC\_POSTPROCESS\_NOT\_INIT** -16
- #define **APC\_POSTPROCESS\_FRAME\_FAIL** -17
- #define **APC\_NotSupport** -18

- #define **APC\_GET\_RES\_LIST\_FAIL** -19
- #define **APC\_READ\_REG\_FAIL** -20
- #define **APC\_WRITE\_REG\_FAIL** -21
- #define **APC\_SET\_FPS\_FAIL** -22
- #define **APC\_VIDEO\_RENDER\_FAIL** -23
- #define **APC\_OPEN\_DEVICE\_FAIL** -24
- #define **APC\_FIND\_DEVICE\_FAIL** -25
- #define **APC\_GET\_IMAGE\_FAIL** -26
- #define **APC\_NOT\_SUPPORT\_RES** -27
- #define **APC\_CALLBACK\_REGISTER\_FAIL** -28
- #define **APC\_CLOSE\_DEVICE\_FAIL** -29
- #define **APC\_GET\_CALIBRATIONLOG\_FAIL** -30
- #define **APC\_SET\_CALIBRATIONLOG\_FAIL** -31
- #define **APC\_DEVICE\_NOT\_SUPPORT** -32
- #define **APC\_DEVICE\_BUSY** -33
- #define **APC\_DEVICE\_TIMEOUT** -34
- #define **APC\_IO\_SELECT\_EINTR** -35
- #define **APC\_IO\_SELECT\_ERROR** -36
- #define **APC\_ILLEGAL\_ANGLE** -40
- #define **APC\_ILLEGAL\_STEP** -41
- #define **APC\_ILLEGAL\_TIMEPERSTEP** -42
- #define **APC\_MOTOR\_RUNNING** -43
- #define **APC\_GETSENSORREG\_FAIL** -44
- #define **APC\_SETSENSORREG\_FAIL** -45
- #define **APC\_READ\_X\_AXIS\_FAIL** -46
- #define **APC\_READ\_Y\_AXIS\_FAIL** -47
- #define **APC\_READ\_Z\_AXIS\_FAIL** -48
- #define **APC\_READ\_PRESS\_DATA\_FAIL** -49
- #define **APC\_READ\_TEMPERATURE\_FAIL** -50
- #define **APC\_RETURNHOME\_RUNNING** -51
- #define **APC\_MOTOTSTOP\_BY\_HOME\_INDEX** -52
- #define **APC\_MOTOTSTOP\_BY\_PROTECT\_SCHEME** -53
- #define **APC\_MOTOTSTOP\_BY\_NORMAL** -54
- #define **APC\_ILLEGAL\_FIRMWARE\_VERSION** -55
- #define **APC\_ILLEGAL\_STEPPERTIME** -56
- #define **APC\_GET\_PU\_PROP\_VAL\_FAIL** -60
- #define **APC\_SET\_PU\_PROP\_VAL\_FAIL** -61
- #define **APC\_GET\_CT\_PROP\_VAL\_FAIL** -62
- #define **APC\_SET\_CT\_PROP\_VAL\_FAIL** -63
- #define **APC\_GET\_CT\_PROP\_RANGE\_STEP\_FAIL** -64
- #define **APC\_GET\_PU\_PROP\_RANGE\_STEP\_FAIL** -65
- #define **APC\_INVALID\_USERDATA** -70
- #define **APC\_MAP\_LUT\_FAIL** -71
- #define **APC\_APPEND\_TO\_FILE\_FRONT\_FAIL** -72
- #define **APC\_TOO\_MANY\_DEVICE** -80
- #define **APC\_ACCESS\_MP4\_EXTRA\_DATA\_FAIL** -81
- #define **BIT\_SET**(a, b) ((a) |= (1<<(b)))
- #define **BIT\_CLEAR**(a, b) ((a) &= ~(1<<(b)))
- #define **BIT\_FLIP**(a, b) ((a) ^= (1<<(b)))
- #define **BIT\_CHECK**(a, b) ((a) & (1<<(b)))
- #define **FG\_Address\_1Byte** 0x01
- #define **FG\_Address\_2Byte** 0x02
- #define **FG\_Value\_1Byte** 0x10
- #define **FG\_Value\_2Byte** 0x20
- #define **EVENT\_BUFFER\_SHM\_COLOR** "/shm\_ring\_buffer\_color"

- #define **EVENT\_BUFFER\_SHM\_DEPTH** "/shm\_ring\_buffer\_depth"
- #define **EVENT\_BUFFER\_SHM** "/shm\_ring\_buffer"
- #define **CMD\_FIFO\_PATH** "/tmp/cmdfifo"
- #define **ZD\_PATH** "/tmp/zd\_addr"
- #define **RECTIFY\_LOG\_PATH** "/tmp/rectifylog\_addr"
- #define **SRB\_LENGTH** 10
- #define **CHIPID\_ADDR** 0xf014
- #define **SERIAL\_2BIT\_ADDR** 0xf0fe
- #define **APC\_DEPTH\_DATA\_OFF\_RAW** 0 /\* raw (depth off, only raw color) \*/
- #define **APC\_DEPTH\_DATA\_DEFAULT** APC\_DEPTH\_DATA\_OFF\_RAW /\* raw (depth off, only gray raw color) \*/
- #define **APC\_DEPTH\_DATA\_8\_BITS** 1 /\* rectify, 1 byte per pixel \*/
- #define **APC\_DEPTH\_DATA\_14\_BITS** 2 /\* rectify, 2 byte per pixel \*/
- #define **APC\_DEPTH\_DATA\_8\_BITS\_x80** 3 /\* rectify, 2 byte per pixel but using 1 byte only \*/
- #define **APC\_DEPTH\_DATA\_11\_BITS** 4 /\* rectify, 2 byte per pixel but using 11 bit only \*/
- #define **APC\_DEPTH\_DATA\_OFF\_RECTIFY** 5 /\* rectify (depth off, only rectify raw color) \*/
- #define **APC\_DEPTH\_DATA\_8\_BITS\_RAW** 6 /\* raw \*/
- #define **APC\_DEPTH\_DATA\_14\_BITS\_RAW** 7 /\* raw \*/
- #define **APC\_DEPTH\_DATA\_8\_BITS\_x80\_RAW** 8 /\* raw \*/
- #define **APC\_DEPTH\_DATA\_11\_BITS\_RAW** 9 /\* raw \*/
- #define **APC\_DEPTH\_DATA\_14\_BITS\_COMBINED\_RECTIFY** 11
- #define **APC\_DEPTH\_DATA\_11\_BITS\_COMBINED\_RECTIFY** 13
- #define **APC\_DEPTH\_DATA\_OFF\_BAYER\_RAW** 14
- #define **APC\_DEPTH\_DATA\_INTERLEAVE\_MODE\_OFFSET** 16
- #define **APC\_DEPTH\_DATA\_ILM\_OFF\_RAW** APC\_DEPTH\_DATA\_OFF\_RAW + APC\_DEPTH\_DATA\_↵  
INTERLEAVE\_MODE\_OFFSET /\* raw (depth off, only raw color) \*/
- #define **APC\_DEPTH\_DATA\_ILM\_DEFAULT** APC\_DEPTH\_DATA\_DEFAULT + APC\_DEPTH\_DATA\_I↵  
NTERLEAVE\_MODE\_OFFSET /\* raw (depth off, only raw color) \*/
- #define **APC\_DEPTH\_DATA\_ILM\_8\_BITS** APC\_DEPTH\_DATA\_8\_BITS + APC\_DEPTH\_DATA\_INTER↵  
LEAVE\_MODE\_OFFSET /\* rectify, 1 byte per pixel \*/
- #define **APC\_DEPTH\_DATA\_ILM\_14\_BITS** APC\_DEPTH\_DATA\_14\_BITS + APC\_DEPTH\_DATA\_INT↵  
ERLEAVE\_MODE\_OFFSET /\* rectify, 2 byte per pixel \*/
- #define **APC\_DEPTH\_DATA\_ILM\_8\_BITS\_x80** APC\_DEPTH\_DATA\_8\_BITS\_x80 + APC\_DEPTH\_DAT↵  
A\_INTERLEAVE\_MODE\_OFFSET /\* rectify, 2 byte per pixel but using 1 byte only \*/
- #define **APC\_DEPTH\_DATA\_ILM\_11\_BITS** APC\_DEPTH\_DATA\_11\_BITS + APC\_DEPTH\_DATA\_INT↵  
ERLEAVE\_MODE\_OFFSET /\* rectify, 2 byte per pixel but using 11 bit only \*/
- #define **APC\_DEPTH\_DATA\_ILM\_OFF\_RECTIFY** APC\_DEPTH\_DATA\_OFF\_RECTIFY + APC\_DEPTH↵  
\_DATA\_INTERLEAVE\_MODE\_OFFSET /\* rectify (depth off, only rectify color) \*/
- #define **APC\_DEPTH\_DATA\_ILM\_8\_BITS\_RAW** APC\_DEPTH\_DATA\_8\_BITS\_RAW + APC\_DEPTH\_D↵  
ATA\_INTERLEAVE\_MODE\_OFFSET /\* raw \*/
- #define **APC\_DEPTH\_DATA\_ILM\_14\_BITS\_RAW** APC\_DEPTH\_DATA\_14\_BITS\_RAW + APC\_DEPTH↵  
\_DATA\_INTERLEAVE\_MODE\_OFFSET /\* raw \*/
- #define **APC\_DEPTH\_DATA\_ILM\_8\_BITS\_x80\_RAW** APC\_DEPTH\_DATA\_8\_BITS\_x80\_RAW + APC\_↵  
DEPTH\_DATA\_INTERLEAVE\_MODE\_OFFSET /\* raw \*/
- #define **APC\_DEPTH\_DATA\_ILM\_11\_BITS\_RAW** APC\_DEPTH\_DATA\_11\_BITS\_RAW + APC\_DEPTH↵  
\_DATA\_INTERLEAVE\_MODE\_OFFSET /\* raw \*/
- #define **APC\_DEPTH\_DATA\_ILM\_14\_BITS\_COMBINED\_RECTIFY** APC\_DEPTH\_DATA\_14\_BITS\_CO↵  
MBINED\_RECTIFY + APC\_DEPTH\_DATA\_INTERLEAVE\_MODE\_OFFSET
- #define **APC\_DEPTH\_DATA\_ILM\_11\_BITS\_COMBINED\_RECTIFY** APC\_DEPTH\_DATA\_11\_BITS\_CO↵  
MBINED\_RECTIFY + APC\_DEPTH\_DATA\_INTERLEAVE\_MODE\_OFFSET
- #define **APC\_DEPTH\_DATA\_SCALE\_DOWN\_MODE\_OFFSET** 32
- #define **APC\_DEPTH\_DATA\_SCALE\_DOWN\_OFF\_RAW** (APC\_DEPTH\_DATA\_OFF\_RAW + APC\_DE↵  
PTH\_DATA\_SCALE\_DOWN\_MODE\_OFFSET)/\* raw (depth off, only raw color) \*/
- #define **APC\_DEPTH\_DATA\_SCALE\_DOWN\_DEFAULT** (APC\_DEPTH\_DATA\_DEFAULT + APC\_DEP↵  
TH\_DATA\_SCALE\_DOWN\_MODE\_OFFSET) /\* raw (depth off, only raw color) \*/



- `#define APC_DEPTH_DATA_SCALE_DOWN_8_BITS (APC_DEPTH_DATA_8_BITS + APC_DEPTH_DATA_SCALE_DOWN_MODE_OFFSET)/* rectify, 1 byte per pixel */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_14_BITS (APC_DEPTH_DATA_14_BITS + APC_DEPTH_DATA_SCALE_DOWN_MODE_OFFSET) /* rectify, 2 byte per pixel */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_8_BITS_x80 (APC_DEPTH_DATA_8_BITS_x80 + APC_DEPTH_DATA_SCALE_DOWN_MODE_OFFSET) /* rectify, 2 byte per pixel but using 1 byte only */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_11_BITS (APC_DEPTH_DATA_11_BITS + APC_DEPTH_DATA_SCALE_DOWN_MODE_OFFSET)/* rectify, 2 byte per pixel but using 11 bit only */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_OFF_RECTIFY (APC_DEPTH_DATA_OFF_RECTIFY + APC_DEPTH_DATA_SCALE_DOWN_MODE_OFFSET) /* Rule 0.4b Reserved unused in any firmware*/`
- `#define APC_DEPTH_DATA_SCALE_DOWN_8_BITS_RAW (APC_DEPTH_DATA_8_BITS_RAW + APC_DEPTH_DATA_SCALE_DOWN_MODE_OFFSET) /* raw */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_14_BITS_RAW (APC_DEPTH_DATA_14_BITS_RAW + APC_DEPTH_DATA_SCALE_DOWN_MODE_OFFSET) /* raw */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_8_BITS_x80_RAW (APC_DEPTH_DATA_8_BITS_x80_RAW + APC_DEPTH_DATA_SCALE_DOWN_MODE_OFFSET) /* raw */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_11_BITS_RAW (APC_DEPTH_DATA_11_BITS_RAW + APC_DEPTH_DATA_SCALE_DOWN_MODE_OFFSET) /* raw */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_14_BITS_COMBINED_RECTIFY (APC_DEPTH_DATA_14_BITS_COMBINED_RECTIFY + APC_DEPTH_DATA_SCALE_DOWN_MODE_OFFSET) /* Rule 0.4b Reserved unused in any firmware*/`
- `#define APC_DEPTH_DATA_SCALE_DOWN_11_BITS_COMBINED_RECTIFY (APC_DEPTH_DATA_11_BITS_COMBINED_RECTIFY + APC_DEPTH_DATA_SCALE_DOWN_MODE_OFFSET) /* Rule 0.4b Reserved unused in any firmware*/`
- `#define APC_DEPTH_DATA_SCALE_DOWN_ILM_OFF_RAW (APC_DEPTH_DATA_SCALE_DOWN_OFF_RAW + APC_DEPTH_DATA_INTERLEAVE_MODE_OFFSET) /* raw (depth off, only raw color) */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_ILM_DEFAULT (APC_DEPTH_DATA_SCALE_DOWN_DEFAULT + APC_DEPTH_DATA_INTERLEAVE_MODE_OFFSET) /* raw (depth off, only raw color) */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_ILM_8_BITS (APC_DEPTH_DATA_SCALE_DOWN_8_BITS + APC_DEPTH_DATA_INTERLEAVE_MODE_OFFSET) /* rectify, 1 byte per pixel */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_ILM_14_BITS (APC_DEPTH_DATA_SCALE_DOWN_14_BITS + APC_DEPTH_DATA_INTERLEAVE_MODE_OFFSET) /* rectify, 2 byte per pixel */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_ILM_8_BITS_x80 (APC_DEPTH_DATA_SCALE_DOWN_8_BITS_x80 + APC_DEPTH_DATA_INTERLEAVE_MODE_OFFSET) /* rectify, 2 byte per pixel but using 1 byte only */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_ILM_11_BITS (APC_DEPTH_DATA_SCALE_DOWN_11_BITS + APC_DEPTH_DATA_INTERLEAVE_MODE_OFFSET) /* rectify, 2 byte per pixel but using 11 bit only */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_ILM_OFF_RECTIFY (APC_DEPTH_DATA_SCALE_DOWN_OFF_RECTIFY + APC_DEPTH_DATA_INTERLEAVE_MODE_OFFSET) /* rectify (depth off, only rectify color) */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_ILM_8_BITS_RAW (APC_DEPTH_DATA_SCALE_DOWN_8_BITS_RAW + APC_DEPTH_DATA_INTERLEAVE_MODE_OFFSET) /* raw */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_ILM_14_BITS_RAW (APC_DEPTH_DATA_SCALE_DOWN_14_BITS_RAW + APC_DEPTH_DATA_INTERLEAVE_MODE_OFFSET) /* raw */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_ILM_8_BITS_x80_RAW (APC_DEPTH_DATA_SCALE_DOWN_8_BITS_x80_RAW + APC_DEPTH_DATA_INTERLEAVE_MODE_OFFSET) /* raw */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_ILM_11_BITS_RAW (APC_DEPTH_DATA_SCALE_DOWN_11_BITS_RAW + APC_DEPTH_DATA_INTERLEAVE_MODE_OFFSET) /* raw */`
- `#define APC_DEPTH_DATA_SCALE_DOWN_ILM_14_BITS_COMBINED_RECTIFY (APC_DEPTH_DATA_SCALE_DOWN_14_BITS_COMBINED_RECTIFY + APC_DEPTH_DATA_INTERLEAVE_MODE_OFFSET)`
- `#define APC_DEPTH_DATA_SCALE_DOWN_ILM_11_BITS_COMBINED_RECTIFY (APC_DEPTH_DATA_SCALE_DOWN_11_BITS_COMBINED_RECTIFY + APC_DEPTH_DATA_INTERLEAVE_MODE_OFFSET)`
- `#define APC_READ_FLASH_TOTAL_SIZE 128`
- `#define APC_READ_FLASH_FW_PLUGIN_SIZE 104`

- `#define APC_WRITE_FLASH_TOTAL_SIZE 128`
- `#define APC_Y_OFFSET_FILE_ID_0 30`
- `#define APC_Y_OFFSET_FILE_SIZE 256`
- `#define APC_RECTIFY_FILE_ID_0 40`
- `#define APC_RECTIFY_FILE_SIZE 1024`
- `#define APC_ZD_TABLE_FILE_ID_0 50`
- `#define APC_ZD_TABLE_FILE_SIZE_8_BITS 512`
- `#define APC_ZD_TABLE_FILE_SIZE_11_BITS 4096`
- `#define APC_CALIB_LOG_FILE_ID_0 240`
- `#define APC_CALIB_LOG_FILE_SIZE 4096`
- `#define APC_USER_DATA_FILE_ID_0 200`
- `#define APC_USER_DATA_FILE_SIZE_0 1024`
- `#define APC_USER_DATA_FILE_SIZE_1 4096`
- `#define APC_PID_8029 0x0568`
- `#define APC_PID_8030 APC_PID_8029`
- `#define APC_PID_8039 APC_PID_8029`
- `#define APC_PID_8031 0x0117`
- `#define APC_PID_8032 0x0118`
- `#define APC_PID_8036 0x0120`
- `#define APC_PID_8037 0x0121`
- `#define APC_PID_8038 0x0124`
- `#define APC_PID_8038_M0 APC_PID_8038`
- `#define APC_PID_8038_M1 0x0147`
- `#define APC_PID_8040W 0x0130`
- `#define APC_PID_8040S 0x0131`
- `#define APC_PID_8040S_K 0x0149`
- `#define APC_PID_8041 0x0126`
- `#define APC_PID_8042 0x0127`
- `#define APC_PID_8043 0x0128`
- `#define APC_PID_8044 0x0129`
- `#define APC_PID_8045K 0x0134`
- `#define APC_PID_8046K 0x0135`
- `#define APC_PID_8051 0x0136`
- `#define APC_PID_8052 0x0137`
- `#define APC_PID_8053 0x0138`
- `#define APC_PID_8054 0x0139`
- `#define APC_PID_8054_K 0x0143`
- `#define APC_PID_8059 0x0146`
- `#define APC_PID_8060 0x0152`
- `#define APC_PID_8060_K 0x0150`
- `#define APC_PID_8060_T 0x0151`
- `#define APC_PID_AMBER 0x0112`
- `#define APC_PID_SALLY 0x0158`
- `#define APC_PID_HYPATIA 0x0160`
- `#define APC_PID_8062 0x0162`
- `#define APC_PID_GRAP 0x0179`
- `#define APC_PID_GRAP_K 0x0183`
- `#define APC_PID_GRAP_SLAVE 0x0279`
- `#define APC_PID_GRAP_SLAVE_K 0x0283`
- `#define APC_PID_SANDRA 0x0167`
- `#define APC_PID_NORA 0x0168`
- `#define APC_PID_HELEN 0x0171`
- `#define APC_PID_GRAP_THERMAL 0xf9f9`
- `#define APC_PID_GRAP_THERMAL2 0xf8f8`
- `#define APC_VID_GRAP_THERMAL 0x04b4`

- #define **APC\_VID\_2170** 0x0110
- #define **CT\_PROPERTY\_ID\_AUTO\_EXPOSURE\_MODE\_CTRL** 0
- #define **CT\_PROPERTY\_ID\_AUTO\_EXPOSURE\_PRIORITY\_CTRL** 1
- #define **CT\_PROPERTY\_ID\_EXPOSURE\_TIME\_ABSOLUTE\_CTRL** 2
- #define **CT\_PROPERTY\_ID\_EXPOSURE\_TIME\_RELATIVE\_CTRL** 3
- #define **CT\_PROPERTY\_ID\_FOCUS\_ABSOLUTE\_CTRL** 4
- #define **CT\_PROPERTY\_ID\_FOCUS\_RELATIVE\_CTRL** 5
- #define **CT\_PROPERTY\_ID\_FOCUS\_AUTO\_CTRL** 6
- #define **CT\_PROPERTY\_ID\_IRIS\_ABSOLUTE\_CTRL** 7
- #define **CT\_PROPERTY\_ID\_IRIS\_RELATIVE\_CTRL** 8
- #define **CT\_PROPERTY\_ID\_ZOOM\_ABSOLUTE\_CTRL** 9
- #define **CT\_PROPERTY\_ID\_ZOOM\_RELATIVE\_CTRL** 10
- #define **CT\_PROPERTY\_ID\_PAN\_ABSOLUTE\_CTRL** 11
- #define **CT\_PROPERTY\_ID\_PAN\_RELATIVE\_CTRL** 12
- #define **CT\_PROPERTY\_ID\_TILT\_ABSOLUTE\_CTRL** 13
- #define **CT\_PROPERTY\_ID\_TILT\_RELATIVE\_CTRL** 14
- #define **CT\_PROPERTY\_ID\_PRIVACY\_CTRL** 15
- #define **PU\_PROPERTY\_ID\_BACKLIGHT\_COMPENSATION\_CTRL** 0
- #define **PU\_PROPERTY\_ID\_BRIGHTNESS\_CTRL** 1
- #define **PU\_PROPERTY\_ID\_CONTRAST\_CTRL** 2
- #define **PU\_PROPERTY\_ID\_GAIN\_CTRL** 3
- #define **PU\_PROPERTY\_ID\_POWER\_LINE\_FREQUENCY\_CTRL** 4
- #define **PU\_PROPERTY\_ID\_HUE\_CTRL** 5
- #define **PU\_PROPERTY\_ID\_HUE\_AUTO\_CTRL** 6
- #define **PU\_PROPERTY\_ID\_SATURATION\_CTRL** 7
- #define **PU\_PROPERTY\_ID\_SHARPNESS\_CTRL** 8
- #define **PU\_PROPERTY\_ID\_GAMMA\_CTRL** 9
- #define **PU\_PROPERTY\_ID\_WHITE\_BALANCE\_CTRL** 10
- #define **PU\_PROPERTY\_ID\_WHITE\_BALANCE\_AUTO\_CTRL** 11
- #define **POSTPAR\_HR\_MODE** 5
- #define **POSTPAR\_HR\_CURVE\_0** 6
- #define **POSTPAR\_HR\_CURVE\_1** 7
- #define **POSTPAR\_HR\_CURVE\_2** 8
- #define **POSTPAR\_HR\_CURVE\_3** 9
- #define **POSTPAR\_HR\_CURVE\_4** 10
- #define **POSTPAR\_HR\_CURVE\_5** 11
- #define **POSTPAR\_HR\_CURVE\_6** 12
- #define **POSTPAR\_HR\_CURVE\_7** 13
- #define **POSTPAR\_HR\_CURVE\_8** 14
- #define **POSTPAR\_HF\_MODE** 17
- #define **POSTPAR\_DC\_MODE** 20
- #define **POSTPAR\_DC\_CNT\_THD** 21
- #define **POSTPAR\_DC\_GRAD\_THD** 22
- #define **POSTPAR\_SEG\_MODE** 23
- #define **POSTPAR\_SEG\_THD\_SUB** 24
- #define **POSTPAR\_SEG\_THD\_SLP** 25
- #define **POSTPAR\_SEG\_THD\_MAX** 26
- #define **POSTPAR\_SEG\_THD\_MIN** 27
- #define **POSTPAR\_SEG\_FILL\_MODE** 28
- #define **POSTPAR\_HF2\_MODE** 31
- #define **POSTPAR\_GRAD\_MODE** 34
- #define **POSTPAR\_TEMP0\_MODE** 37
- #define **POSTPAR\_TEMP0\_THD** 38
- #define **POSTPAR\_TEMP1\_MODE** 41
- #define **POSTPAR\_TEMP1\_LEVEL** 42

- `#define POSTPAR_TEMP1_THD 43`
- `#define POSTPAR_FC_MODE 46`
- `#define POSTPAR_FC_EDGE_THD 47`
- `#define POSTPAR_FC_AREA_THD 48`
- `#define POSTPAR_MF_MODE 51`
- `#define POSTPAR_ZM_MODE 52`
- `#define POSTPAR_RF_MODE 53`
- `#define POSTPAR_RF_LEVEL 54`

## Typedefs

- typedef unsigned char **BYTE**
- typedef signed int **BOOL**
- typedef unsigned short **WORD**
- typedef struct [packet\\_s](#) **srb\_packet\_s**
- typedef struct [tagDEVINFORMATION](#) **DEVINFORMATION**
- typedef struct [tagDEVINFORMATION](#) \* **PDEVINFORMATION**
- typedef struct [tagDEVSEL](#) **DEVSELINFO**
- typedef struct [tagDEVSEL](#) \* **PDEVSELINFO**
- typedef struct [tagAPC\\_STREAM\\_INFO](#) **APC\_STREAM\_INFO**
- typedef struct [tagAPC\\_STREAM\\_INFO](#) \* **PAPC\_STREAM\_INFO**
- typedef struct [tagZDTableInfo](#) **ZDTABLEINFO**
- typedef struct [tagZDTableInfo](#) \* **PZDTABLEINFO**
- typedef struct [tagKEEP\\_DATA\\_CTRL](#) **KEEP\_DATA\_CTRL**
- typedef enum **AE\_STATUS** \* **PAE\_STATUS**
- typedef enum **AWB\_STATUS** \* **PAWB\_STATUS**
- typedef struct [eSPCtrl\\_RectLogData](#) **eSPCtrl\_RectLogData**
- typedef struct [GyroTag](#) **GYRO\_ANGULAR\_RATE\_DATA**
- typedef struct [AccelerationTag](#) **ACCELERATION\_DATA**
- typedef struct [CompassTag](#) **COMPASS\_DATA**

## Enumerations

- enum **SENSORMODE\_INFO** {  
**SENSOR\_A** = 0, **SENSOR\_B**, **SENSOR\_BOTH**, **SENSOR\_C**,  
**SENSOR\_D** }
- enum **PIXEL\_FMT** {  
**YUV22\_YUYV\_PIXEL\_FMT** = 0, **YUV22\_UYVY\_PIXEL\_FMT**, **RAW10\_GBRG\_PIXEL\_FMT**, **RAW10\_BGRG\_PIXEL\_FMT**,  
**RAW10\_RGBG\_PIXEL\_FMT**, **RAW10\_GRBG\_PIXEL\_FMT**, **MJPEG\_PIXEL\_FMT**, **UNKNOWN\_PIXEL\_FMT** = 0xffff }
- enum **DEVICE\_TYPE** {  
**OTHERS** = 0, **AXES1**, **PUMA**, **KIWI**,  
**UNKNOWN\_DEVICE\_TYPE** = 0xffff }
- enum **FLASH\_DATA\_TYPE** {  
**Total** = 0, **FW\_PLUGIN**, **BOOTLOADER\_ONLY**, **FW\_ONLY**,  
**PLUGIN\_ONLY** }
- enum **USERDATA\_SECTION\_INDEX** {  
**USERDATA\_SECTION\_0** = 0, **USERDATA\_SECTION\_1**, **USERDATA\_SECTION\_2**, **USERDATA\_SECTION\_3**,  
**USERDATA\_SECTION\_4**, **USERDATA\_SECTION\_5**, **USERDATA\_SECTION\_6**, **USERDATA\_SECTION\_7**,  
**USERDATA\_SECTION\_8**, **USERDATA\_SECTION\_9** }

- enum **CALIBRATION\_LOG\_TYPE** {  
**ALL\_LOG** = 0, **SERIAL\_NUMBER**, **PRJFILE\_LOG**, **STAGE\_TIME\_RESULT\_LOG**,  
**SENSOR\_OFFSET**, **AUTO\_ADJUST\_LOG**, **RECTIFY\_LOG**, **ZD\_LOG**,  
**DEPTHMAP\_KOG** }
- enum **CONTROL\_MODE** { **IMAGE\_SN\_NONSYNC** = 0, **IMAGE\_SN\_SYNC** }
- enum **DEPTH\_TRANSFER\_CTRL** { **DEPTH\_IMG\_NON\_TRANSFER**, **DEPTH\_IMG\_GRAY\_TRANSFER**,  
**DEPTH\_IMG\_COLORFUL\_TRANSFER** }
- enum **SENSOR\_TYPE\_NAME** {  
**APC\_SENSOR\_TYPE\_H22** = 0, **APC\_SENSOR\_TYPE\_H65** = 1, **APC\_SENSOR\_TYPE\_OV7740** = 2, **APC\_SENSOR\_TYPE\_AR0134** = 3,  
**APC\_SENSOR\_TYPE\_AR0135** = 4, **APC\_SENSOR\_TYPE\_AR0144** = 5, **APC\_SENSOR\_TYPE\_AR0330**  
= 6, **APC\_SENSOR\_TYPE\_AR0522** = 7,  
**APC\_SENSOR\_TYPE\_AR1335** = 8, **APC\_SENSOR\_TYPE\_OV9714** = 9, **APC\_SENSOR\_TYPE\_OV9282**  
= 10, **APC\_SENSOR\_TYPE\_H68** = 11,  
**APC\_SENSOR\_TYPE\_OV2740** = 12, **APC\_SENSOR\_TYPE\_OC0SA10** = 13, **APC\_SENSOR\_TYPE\_UNKNOWN** = 0xffff }
- enum **AE\_STATUS** { **AE\_ENABLE** = 0, **AE\_DISABLE** }
- enum **AWB\_STATUS** { **AWB\_ENABLE** = 0, **AWB\_DISABLE** }
- enum **USB\_PORT\_TYPE** { **USB\_PORT\_TYPE\_2\_0** = 2, **USB\_PORT\_TYPE\_3\_0**, **USB\_PORT\_TYPE\_UNKNOWN** }
- enum **SENSITIVITY\_LEVEL\_L3G** { **DPS\_245** = 0, **DPS\_500**, **DPS\_2000** }
- enum **SENSITIVITY\_LEVEL\_LSM** {  
**\_2G** = 0, **\_4G**, **\_6G**, **\_8G**,  
**\_16G** }
- enum **OUTPUT\_DATA\_RATE** {  
**One\_Shot** = 0, **\_1\_HZ\_1\_HZ**, **\_7\_HZ\_1\_HZ**, **\_12\_5\_HZ\_1HZ**,  
**\_25\_HZ\_1\_HZ**, **\_7\_HZ\_7\_HZ**, **\_12\_5\_HZ\_12\_5\_HZ**, **\_25\_HZ\_25\_HZ** }
- enum **POWER\_STATE** { **POWER\_ON** = 0, **POWER\_OFF** }
- enum **BRIGHTNESS\_LEVEL** {  
**LEVEL\_0** = 0, **LEVEL\_1**, **LEVEL\_2**, **LEVEL\_3**,  
**LEVEL\_4**, **LEVEL\_5**, **LEVEL\_6**, **LEVEL\_7**,  
**LEVEL\_8**, **LEVEL\_9**, **LEVEL\_10**, **LEVEL\_11**,  
**LEVEL\_12**, **LEVEL\_13**, **LEVEL\_14**, **LEVEL\_15** }

### 9.2.1 Detailed Description

error/data type definitions

Copyright:

This file copyright (C) 2021 by eYs3D Microelectronics, Co.

An unpublished work. All rights reserved.

This file is proprietary information, and may not be disclosed or copied without the prior permission of eYs3D Microelectronics, Co.



# Index

APC\_ApplyFilters  
    eSPDI.h, [50](#)

APC\_CloseCmdFiFo  
    eSPDI.h, [51](#)

APC\_CloseDevice  
    eSPDI.h, [51](#)

APC\_CloseDeviceEx  
    eSPDI.h, [52](#)

APC\_CloseDeviceMBL  
    eSPDI.h, [52](#)

APC\_ColorFormat\_to\_RGB24  
    eSPDI.h, [53](#)

APC\_Convert\_Depth\_Y\_To\_Buffer  
    eSPDI.h, [53](#)

APC\_Convert\_Depth\_Y\_To\_Buffer\_offset  
    eSPDI.h, [54](#)

APC\_CreateSwPostProc  
    eSPDI.h, [54](#)

APC\_DecryptMP4  
    eSPDI.h, [55](#)

APC\_DecryptString  
    eSPDI.h, [55](#), [56](#)

APC\_DepthMerge  
    eSPDI.h, [56](#)

APC\_DisableAWB  
    eSPDI.h, [57](#)

APC\_DisableAE  
    eSPDI.h, [57](#)

APC\_DoFusion  
    eSPDI.h, [58](#)

APC\_DoSwPostProc  
    eSPDI.h, [58](#)

APC\_EdgePreServingFilter  
    eSPDI.h, [59](#)

APC\_EnableAWB  
    eSPDI.h, [60](#)

APC\_EnableAE  
    eSPDI.h, [60](#)

APC\_EnableGPUAcceleration  
    eSPDI.h, [60](#)

APC\_EnableInterleave  
    eSPDI.h, [62](#)

APC\_EnableSensorIF  
    eSPDI.h, [62](#)

APC\_EncryptMP4  
    eSPDI.h, [63](#)

APC\_EncryptString  
    eSPDI.h, [63](#)

APC\_FindDevice  
    eSPDI.h, [64](#)

APC\_FlyingDepthCancellation\_D11  
    eSPDI.h, [64](#)

APC\_FlyingDepthCancellation\_D8  
    eSPDI.h, [65](#)

APC\_GenerateLutFile  
    eSPDI.h, [65](#)

APC\_Get2Image  
    eSPDI.h, [66](#)

APC\_Get\_150\_mm\_depth  
    eSPDI.h, [66](#)

APC\_Get\_60\_mm\_depth  
    eSPDI.h, [67](#)

APC\_Get\_Color\_30\_mm\_depth  
    eSPDI.h, [67](#)

APC\_GetAESTatus  
    eSPDI.h, [68](#)

APC\_GetAWBStatus  
    eSPDI.h, [69](#)

APC\_GetAccMeterValue  
    eSPDI.h, [68](#)

APC\_GetAutoExposureMode  
    eSPDI.h, [69](#)

APC\_GetBusInfo  
    eSPDI.h, [70](#)

APC\_GetCTPropVal  
    eSPDI.h, [72](#)

APC\_GetCTRangeAndStep  
    eSPDI.h, [73](#)

APC\_GetColorGain  
    eSPDI.h, [70](#)

APC\_GetColorImage  
    eSPDI.h, [71](#)

APC\_GetColorImageWithTimestamp  
    eSPDI.h, [71](#)

APC\_GetControlCounterMode  
    eSPDI.h, [72](#)

APC\_GetCurrentIRValue  
    eSPDI.h, [74](#)

APC\_GetDepthDataType  
    eSPDI.h, [74](#)

APC\_GetDepthImage  
    eSPDI.h, [75](#)

APC\_GetDepthImageWithTimestamp  
    eSPDI.h, [75](#)

APC\_GetDeviceInfo  
    eSPDI.h, [76](#)

APC\_GetDeviceInfoMBL\_15cm  
    eSPDI.h, [76](#)

APC\_GetDeviceNumber  
eSPDI.h, [77](#)

APC\_GetDeviceResolutionList  
eSPDI.h, [77](#)

APC\_GetExposureTime  
eSPDI.h, [78](#)

APC\_GetFWRegister  
eSPDI.h, [79](#)

APC\_GetFlexibleGyroData  
eSPDI.h, [78](#)

APC\_GetFlexibleGyroLength  
eSPDI.h, [79](#)

APC\_GetFwVersion  
eSPDI.h, [80](#)

APC\_GetGlobalGain  
eSPDI.h, [80](#)

APC\_GetHWRegister  
eSPDI.h, [81](#)

APC\_GetHidGyro  
eSPDI.h, [81](#)

APC\_GetIRMaxValue  
eSPDI.h, [84](#)

APC\_GetIRMinValue  
eSPDI.h, [85](#)

APC\_GetIRMode  
eSPDI.h, [85](#)

APC\_GetImage  
eSPDI.h, [82](#)

APC\_GetImageInterrupt  
eSPDI.h, [82](#)

APC\_GetInfoHidGyro  
eSPDI.h, [82](#)

APC\_GetInterleaveMode  
eSPDI.h, [84](#)

APC\_GetLogData  
eSPDI.h, [85](#)

APC\_GetLutData  
eSPDI.h, [86](#)

APC\_GetMultiBytesHWRegister  
eSPDI.h, [86](#)

APC\_GetPUPPropVal  
eSPDI.h, [88](#)

APC\_GetPURangeAndStep  
eSPDI.h, [89](#)

APC\_GetPidVid  
eSPDI.h, [87](#)

APC\_GetPointCloud  
eSPDI.h, [87](#)

APC\_GetRectifyLogData  
eSPDI.h, [90](#)

APC\_GetRectifyMatLogData  
eSPDI.h, [90](#)

APC\_GetRectifyTable  
eSPDI.h, [90](#)

APC\_GetSRB  
eSPDI.h, [92](#)

APC\_GetSensorRegister  
eSPDI.h, [91](#)

APC\_GetSerialNumber  
eSPDI.h, [92](#)

APC\_GetThermalFD  
eSPDI.h, [92](#)

APC\_GetUACData  
eSPDI.h, [93](#)

APC\_GetUserData  
eSPDI.h, [94](#)

APC\_GetYOffset  
eSPDI.h, [94](#)

APC\_GetZDTable  
eSPDI.h, [95](#)

APC\_HoleFill  
eSPDI.h, [95](#)

APC\_HoleFilled  
eSPDI.h, [96](#)

APC\_ImgMirro  
eSPDI.h, [96](#), [98](#)

APC\_Init  
eSPDI.h, [98](#)

APC\_InitPostProcess  
eSPDI.h, [100](#)

APC\_InitSRB  
eSPDI.h, [101](#)

APC\_InitialCmdFiFo  
eSPDI.h, [99](#)

APC\_InitialFlexibleGyro  
eSPDI.h, [99](#)

APC\_InitialHidGyro  
eSPDI.h, [100](#)

APC\_InitialUAC  
eSPDI.h, [100](#)

APC\_InjectExtraDataToMp4  
eSPDI.h, [101](#)

APC\_IsInterleaveDevice  
eSPDI.h, [102](#)

APC\_IsMLBaseLine  
eSPDI.h, [102](#)

APC\_OpenDevice  
eSPDI.h, [102](#)

APC\_OpenDevice2  
eSPDI.h, [103](#)

APC\_OpenDeviceMBL  
eSPDI.h, [104](#)

APC\_PostProcess  
eSPDI.h, [105](#)

APC\_PutSRB  
eSPDI.h, [106](#)

APC\_RGB2BMP  
eSPDI.h, [111](#)

APC\_ReadCmdFiFo  
eSPDI.h, [106](#)

APC\_ReadFlashData  
eSPDI.h, [106](#)

APC\_RefreshDevice  
eSPDI.h, [107](#)

APC\_Release  
eSPDI.h, [107](#)



APC\_ReleaseFlexibleGyro  
     eSPDI.h, [108](#)  
 APC\_ReleaseHidGyro  
     eSPDI.h, [108](#)  
 APC\_ReleasePostProcess  
     eSPDI.h, [108](#)  
 APC\_ReleaseSwPostProc  
     eSPDI.h, [109](#)  
 APC\_ReleaseUAC  
     eSPDI.h, [109](#)  
 APC\_ResetFilters  
     eSPDI.h, [109](#)  
 APC\_ResizeImgToHalf  
     eSPDI.h, [110](#)  
 APC\_RetrieveExtraDataFromMp4  
     eSPDI.h, [110](#)  
 APC\_RotateImg180  
     eSPDI.h, [111](#), [112](#)  
 APC\_RotateImg90  
     eSPDI.h, [112](#), [113](#)  
 APC\_SaveLutData  
     eSPDI.h, [114](#)  
 APC\_SelectDevice  
     eSPDI.h, [114](#)  
 APC\_SetAutoExposureMode  
     eSPDI.h, [114](#)  
 APC\_SetCTPropVal  
     eSPDI.h, [116](#)  
 APC\_SetColorGain  
     eSPDI.h, [115](#)  
 APC\_SetControlCounterMode  
     eSPDI.h, [115](#)  
 APC\_SetCurrentIRValue  
     eSPDI.h, [116](#)  
 APC\_SetDepthDataType  
     eSPDI.h, [117](#)  
 APC\_SetExposureTime  
     eSPDI.h, [117](#)  
 APC\_SetFWRegister  
     eSPDI.h, [118](#)  
 APC\_SetGlobalGain  
     eSPDI.h, [118](#)  
 APC\_SetHWRegister  
     eSPDI.h, [119](#)  
 APC\_SetIRMaxValue  
     eSPDI.h, [120](#)  
 APC\_SetIRMode  
     eSPDI.h, [120](#)  
 APC\_SetInterleaveMode  
     eSPDI.h, [119](#)  
 APC\_SetLogData  
     eSPDI.h, [120](#)  
 APC\_SetMultiBytesHWRegister  
     eSPDI.h, [121](#)  
 APC\_SetPUPPropVal  
     eSPDI.h, [122](#)  
 APC\_SetPidVid  
     eSPDI.h, [122](#)  
 APC\_SetRectifyTable  
     eSPDI.h, [122](#)  
 APC\_SetSensorRegister  
     eSPDI.h, [123](#)  
 APC\_SetSensorTypeName  
     eSPDI.h, [124](#)  
 APC\_SetSerialNumber  
     eSPDI.h, [124](#)  
 APC\_SetUserData  
     eSPDI.h, [126](#)  
 APC\_SetYOffset  
     eSPDI.h, [126](#)  
 APC\_SetZDTable  
     eSPDI.h, [127](#)  
 APC\_Setup\_v4l2\_requestbuffers  
     eSPDI.h, [124](#)  
 APC\_SetupBlock  
     eSPDI.h, [125](#)  
 APC\_SetupHidGyro  
     eSPDI.h, [125](#)  
 APC\_SubSample  
     eSPDI.h, [127](#)  
 APC\_SwitchBaseline  
     eSPDI.h, [128](#)  
 APC\_TableToData  
     eSPDI.h, [128](#)  
 APC\_TemporalFilter  
     eSPDI.h, [129](#)  
 APC\_WriteCmdFiFo  
     eSPDI.h, [130](#)  
 APC\_WriteFlashData  
     eSPDI.h, [130](#)  
 APC\_WriteWaveEnd  
     eSPDI.h, [131](#)  
 APC\_WriteWaveHeader  
     eSPDI.h, [131](#)  
 APC\_getUACNAME  
     eSPDI.h, [93](#)  
 APCImageType, [31](#)  
 AccelerationTag, [31](#)  
  
 CamDist1  
     eSPCtrl\_RectLogData, [33](#)  
 CamDist2  
     eSPCtrl\_RectLogData, [33](#)  
 CamMat1  
     eSPCtrl\_RectLogData, [33](#)  
 CamMat2  
     eSPCtrl\_RectLogData, [33](#)  
 CompassTag, [32](#)  
  
 eSPCtrl\_RectLogData, [32](#)  
     CamDist1, [33](#)  
     CamDist2, [33](#)  
     CamMat1, [33](#)  
     CamMat2, [33](#)  
     InImgHeight, [33](#)  
     InImgWidth, [33](#)  
     LRotaMat, [33](#)

- nLineBuffers, [34](#)
- NewCamMat1, [33](#)
- NewCamMat2, [34](#)
- OutImgHeight, [34](#)
- OutImgWidth, [34](#)
- RECT\_AvgErr, [34](#)
- RECT\_Crop\_Col\_BG\_L, [34](#)
- RECT\_Crop\_Col\_ED\_L, [34](#)
- RECT\_Crop\_Row\_BG, [35](#)
- RECT\_Crop\_Row\_ED, [35](#)
- RECT\_CropEnable, [35](#)
- RECT\_Scale\_Col\_M, [35](#)
- RECT\_Scale\_Col\_N, [35](#)
- RECT\_Scale\_Row\_M, [35](#)
- RECT\_Scale\_Row\_N, [35](#)
- RECT\_ScaleEnable, [35](#)
- RECT\_ScaleHeight, [36](#)
- RECT\_ScaleWidth, [36](#)
- RRotaMat, [36](#)
- RotaMat, [36](#)
- TranMat, [36](#)
- uByteArray, [36](#)
- eSPDI.h
  - APC\_ApplyFilters, [50](#)
  - APC\_CloseCmdFiFo, [51](#)
  - APC\_CloseDevice, [51](#)
  - APC\_CloseDeviceEx, [52](#)
  - APC\_CloseDeviceMBL, [52](#)
  - APC\_ColorFormat\_to\_RGB24, [53](#)
  - APC\_Convert\_Depth\_Y\_To\_Buffer, [53](#)
  - APC\_Convert\_Depth\_Y\_To\_Buffer\_offset, [54](#)
  - APC\_CreateSwPostProc, [54](#)
  - APC\_DecryptMP4, [55](#)
  - APC\_DecryptString, [55, 56](#)
  - APC\_DepthMerge, [56](#)
  - APC\_DisableAWB, [57](#)
  - APC\_DisableAE, [57](#)
  - APC\_DoFusion, [58](#)
  - APC\_DoSwPostProc, [58](#)
  - APC\_EdgePreServingFilter, [59](#)
  - APC\_EnableAWB, [60](#)
  - APC\_EnableAE, [60](#)
  - APC\_EnableGPUAcceleration, [60](#)
  - APC\_EnableInterleave, [62](#)
  - APC\_EnableSensorIF, [62](#)
  - APC\_EncryptMP4, [63](#)
  - APC\_EncryptString, [63](#)
  - APC\_FindDevice, [64](#)
  - APC\_FlyingDepthCancellation\_D11, [64](#)
  - APC\_FlyingDepthCancellation\_D8, [65](#)
  - APC\_GenerateLutFile, [65](#)
  - APC\_Get2Image, [66](#)
  - APC\_Get\_150\_mm\_depth, [66](#)
  - APC\_Get\_60\_mm\_depth, [67](#)
  - APC\_Get\_Color\_30\_mm\_depth, [67](#)
  - APC\_GetAESTatus, [68](#)
  - APC\_GetAWBStatus, [69](#)
  - APC\_GetAccMeterValue, [68](#)
  - APC\_GetAutoExposureMode, [69](#)
  - APC\_GetBusInfo, [70](#)
  - APC\_GetCTPropVal, [72](#)
  - APC\_GetCTRangeAndStep, [73](#)
  - APC\_GetColorGain, [70](#)
  - APC\_GetColorImage, [71](#)
  - APC\_GetColorImageWithTimestamp, [71](#)
  - APC\_GetControlCounterMode, [72](#)
  - APC\_GetCurrentIRValue, [74](#)
  - APC\_GetDepthDataType, [74](#)
  - APC\_GetDepthImage, [75](#)
  - APC\_GetDepthImageWithTimestamp, [75](#)
  - APC\_GetDeviceInfo, [76](#)
  - APC\_GetDeviceInfoMBL\_15cm, [76](#)
  - APC\_GetDeviceNumber, [77](#)
  - APC\_GetDeviceResolutionList, [77](#)
  - APC\_GetExposureTime, [78](#)
  - APC\_GetFWRegister, [79](#)
  - APC\_GetFlexibleGyroData, [78](#)
  - APC\_GetFlexibleGyroLength, [79](#)
  - APC\_GetFwVersion, [80](#)
  - APC\_GetGlobalGain, [80](#)
  - APC\_GetHWRegister, [81](#)
  - APC\_GetHidGyro, [81](#)
  - APC\_GetIRMaxValue, [84](#)
  - APC\_GetIRMinValue, [85](#)
  - APC\_GetIRMode, [85](#)
  - APC\_GetImage, [82](#)
  - APC\_GetImageInterrupt, [82](#)
  - APC\_GetInfoHidGyro, [82](#)
  - APC\_GetInterleaveMode, [84](#)
  - APC\_GetLogData, [85](#)
  - APC\_GetLutData, [86](#)
  - APC\_GetMultiBytesHWRegister, [86](#)
  - APC\_GetPUPropVal, [88](#)
  - APC\_GetPURangeAndStep, [89](#)
  - APC\_GetPidVid, [87](#)
  - APC\_GetPointCloud, [87](#)
  - APC\_GetRectifyLogData, [90](#)
  - APC\_GetRectifyMatLogData, [90](#)
  - APC\_GetRectifyTable, [90](#)
  - APC\_GetSRB, [92](#)
  - APC\_GetSensorRegister, [91](#)
  - APC\_GetSerialNumber, [92](#)
  - APC\_GetThermalFD, [92](#)
  - APC\_GetUACData, [93](#)
  - APC\_GetUserData, [94](#)
  - APC\_GetYOffset, [94](#)
  - APC\_GetZDTable, [95](#)
  - APC\_HoleFill, [95](#)
  - APC\_HoleFilled, [96](#)
  - APC\_ImgMirro, [96, 98](#)
  - APC\_Init, [98](#)
  - APC\_InitPostProcess, [100](#)
  - APC\_InitSRB, [101](#)
  - APC\_InitialCmdFiFo, [99](#)
  - APC\_InitialFlexibleGyro, [99](#)
  - APC\_InitialHidGyro, [100](#)

- APC\_InitialUAC, [100](#)
- APC\_InjectExtraDataToMp4, [101](#)
- APC\_IsInterleaveDevice, [102](#)
- APC\_IsMLBaseLine, [102](#)
- APC\_OpenDevice, [102](#)
- APC\_OpenDevice2, [103](#)
- APC\_OpenDeviceMBL, [104](#)
- APC\_PostProcess, [105](#)
- APC\_PutSRB, [106](#)
- APC\_RGB2BMP, [111](#)
- APC\_ReadCmdFiFo, [106](#)
- APC\_ReadFlashData, [106](#)
- APC\_RefreshDevice, [107](#)
- APC\_Release, [107](#)
- APC\_ReleaseFlexibleGyro, [108](#)
- APC\_ReleaseHidGyro, [108](#)
- APC\_ReleasePostProcess, [108](#)
- APC\_ReleaseSwPostProc, [109](#)
- APC\_ReleaseUAC, [109](#)
- APC\_ResetFilters, [109](#)
- APC\_ResizeImgToHalf, [110](#)
- APC\_RetrieveExtraDataFromMp4, [110](#)
- APC\_RotateImg180, [111](#), [112](#)
- APC\_RotateImg90, [112](#), [113](#)
- APC\_SaveLutData, [114](#)
- APC\_SelectDevice, [114](#)
- APC\_SetAutoExposureMode, [114](#)
- APC\_SetCTPropVal, [116](#)
- APC\_SetColorGain, [115](#)
- APC\_SetControlCounterMode, [115](#)
- APC\_SetCurrentIRValue, [116](#)
- APC\_SetDepthDataType, [117](#)
- APC\_SetExposureTime, [117](#)
- APC\_SetFWRegister, [118](#)
- APC\_SetGlobalGain, [118](#)
- APC\_SetHWRegister, [119](#)
- APC\_SetIRMaxValue, [120](#)
- APC\_SetIRMode, [120](#)
- APC\_SetInterleaveMode, [119](#)
- APC\_SetLogData, [120](#)
- APC\_SetMultiBytesHWRegister, [121](#)
- APC\_SetPUPPropVal, [122](#)
- APC\_SetPidVid, [122](#)
- APC\_SetRectifyTable, [122](#)
- APC\_SetSensorRegister, [123](#)
- APC\_SetSensorTypeName, [124](#)
- APC\_SetSerialNumber, [124](#)
- APC\_SetUserData, [126](#)
- APC\_SetYOffset, [126](#)
- APC\_SetZDTable, [127](#)
- APC\_Setup\_v4l2\_requestbuffers, [124](#)
- APC\_SetupBlock, [125](#)
- APC\_SetupHidGyro, [125](#)
- APC\_SubSample, [127](#)
- APC\_SwitchBaseline, [128](#)
- APC\_TableToData, [128](#)
- APC\_TemporalFilter, [129](#)
- APC\_WriteCmdFiFo, [130](#)
- APC\_WriteFlashData, [130](#)
- APC\_WriteWaveEnd, [131](#)
- APC\_WriteWaveHeader, [131](#)
- APC\_getUACNAME, [93](#)
- eSPDI/eSPDI.h, [41](#)
- eSPDI/eSPDI\_def.h, [131](#)
- GyroTag, [37](#)
- InImgHeight
  - eSPCtrl\_RectLogData, [33](#)
- InImgWidth
  - eSPCtrl\_RectLogData, [33](#)
- LRotaMat
  - eSPCtrl\_RectLogData, [33](#)
- nLineBuffers
  - eSPCtrl\_RectLogData, [34](#)
- NewCamMat1
  - eSPCtrl\_RectLogData, [33](#)
- NewCamMat2
  - eSPCtrl\_RectLogData, [34](#)
- OutImgHeight
  - eSPCtrl\_RectLogData, [34](#)
- OutImgWidth
  - eSPCtrl\_RectLogData, [34](#)
- packet\_s, [37](#)
- PointCloudInfo, [37](#)
- RECT\_AvgErr
  - eSPCtrl\_RectLogData, [34](#)
- RECT\_Crop\_Col\_BG\_L
  - eSPCtrl\_RectLogData, [34](#)
- RECT\_Crop\_Col\_ED\_L
  - eSPCtrl\_RectLogData, [34](#)
- RECT\_Crop\_Row\_BG
  - eSPCtrl\_RectLogData, [35](#)
- RECT\_Crop\_Row\_ED
  - eSPCtrl\_RectLogData, [35](#)
- RECT\_CropEnable
  - eSPCtrl\_RectLogData, [35](#)
- RECT\_Scale\_Col\_M
  - eSPCtrl\_RectLogData, [35](#)
- RECT\_Scale\_Col\_N
  - eSPCtrl\_RectLogData, [35](#)
- RECT\_Scale\_Row\_M
  - eSPCtrl\_RectLogData, [35](#)
- RECT\_Scale\_Row\_N
  - eSPCtrl\_RectLogData, [35](#)
- RECT\_ScaleEnable
  - eSPCtrl\_RectLogData, [35](#)
- RECT\_ScaleHeight
  - eSPCtrl\_RectLogData, [36](#)
- RECT\_ScaleWidth
  - eSPCtrl\_RectLogData, [36](#)
- RRotaMat
  - eSPCtrl\_RectLogData, [36](#)

## RotaMat

eSPCtrl\_RectLogData, [36](#)

tagAPC\_STREAM\_INFO, [38](#)

tagDEVINFORMATION, [38](#)

tagDEVSEL, [38](#)

tagKEEP\_DATA\_CTRL, [38](#)

tagZDTableInfo, [39](#)

## TranMat

eSPCtrl\_RectLogData, [36](#)

## uByteArray

eSPCtrl\_RectLogData, [36](#)