

eYs3D Linux SDK v5.0.0.5

Generated by Doxygen 1.8.13

# **Contents**

1	Intro	oduction	1																1
2	Bacl	kground	i																3
3	Bacl	kground	i																9
4	Bacl	kground	i																15
5	Bacl	kground	i																21
6	Clas	s Index																	27
	6.1	Class I	List							 	 	 	 	 		 			 27
7	File	Index																	29
	7.1	File Lis	st							 	•		 29						
8	Clas	ss Docu	mentation	n															31
	8.1	Accele	rationTag	Stru	ıct Re	efere	nce			 			 31						
	8.2	APCIm	nageType \$	Stru	ct Re	fere	nce			 			 31						
	8.3	Compa	assTag Str	ruct	Refer	ence	е.			 			 32						
	8.4	eSPCt	rl_RectLog	gDa	ta Str	uct I	Refe	erenc	е	 			 32						
		8.4.1	Member	Dat	a Do	cume	enta	tion		 			 33						
			8.4.1.1	Ca	amDis	st1				 			 33						
			8.4.1.2	Ca	amDis	st2				 			 33						
			8.4.1.3	Ca	amMa	at1.				 			 33						
			8.4.1.4	Ca	amMa	at2.				 			 33						
			8.4.1.5	Inl	lmgH	eigh	ıt .			 			 33						

ii CONTENTS

	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	OutlmgWidth	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 R	Reference	37
8.6	packet_s Struct F	Reference	37
8.7	PointCloudInfo St	truct Reference	37
8.8	tagAPC_STREAM	M_INFO Struct Reference	38
8.9	tagDEVINFORM/	ATION Struct Reference	38
8.10	tagDEVSEL Struc	ct Reference	38
8.11	tagKEEP_DATA_	_CTRL Struct Reference	38
8.12	tagZDTableInfo S	Struct Reference	39

CONTENTS

9	File	Docum	entation		41
	9.1	eSPDI	/eSPDI.h F	File Reference	41
		9.1.1	Detailed	Description	50
		9.1.2	Function	Documentation	50
			9.1.2.1	APC_ApplyFilters()	50
			9.1.2.2	APC_CloseCmdFiFo()	51
			9.1.2.3	APC_CloseDevice()	51
			9.1.2.4	APC_CloseDeviceEx()	52
			9.1.2.5	APC_CloseDeviceMBL()	52
			9.1.2.6	APC_ColorFormat_to_RGB24()	52
			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]	55
			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()	57
			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()	61
			9.1.2.23	APC_EnableSensorIF()	61
			9.1.2.24	APC_EncryptMP4()	62
			9.1.2.25	APC_EncryptString() [1/2]	62
			9.1.2.26	APC_EncryptString() [2/2]	62

iv CONTENTS

9.1.2.27	APC_FindDevice()	63
9.1.2.28	APC_FlyingDepthCancellation_D11()	63
9.1.2.29	APC_FlyingDepthCancellation_D8()	64
9.1.2.30	APC_GenerateLutFile()	64
9.1.2.31	APC_Get2Image()	65
9.1.2.32	APC_Get_150_mm_depth()	65
9.1.2.33	APC_Get_60_mm_depth()	66
9.1.2.34	APC_Get_Color_30_mm_depth()	66
9.1.2.35	APC_GetAccMeterValue()	67
9.1.2.36	APC_GetAEStatus()	67
9.1.2.37	APC_GetAutoExposureMode()	68
9.1.2.38	APC_GetAWBStatus()	68
9.1.2.39	APC_GetBusInfo()	69
9.1.2.40	APC_GetColorGain()	69
9.1.2.41	APC_GetColorImage()	70
9.1.2.42	APC_GetColorImageWithTimestamp()	70
9.1.2.43	APC_GetControlCounterMode()	71
9.1.2.44	APC_GetCTPropVal()	71
9.1.2.45	APC_GetCTRangeAndStep()	72
9.1.2.46	APC_GetCurrentIRValue()	73
9.1.2.47	APC_GetDepthDataType()	73
9.1.2.48	APC_GetDepthImage()	74
9.1.2.49	APC_GetDepthImageWithTimestamp()	74
9.1.2.50	APC_GetDeviceInfo()	75
9.1.2.51	APC_GetDeviceInfoMBL_15cm()	75
9.1.2.52	APC_GetDeviceNumber()	76
9.1.2.53	APC_GetDeviceResolutionList()	76
9.1.2.54	APC_GetExposureTime()	77
9.1.2.55	APC_GetFlexibleGyroData()	77
9.1.2.56	APC_GetFlexibleGyroLength()	78

CONTENTS

9.1.2.57	APC_GetFWRegister()	78
9.1.2.58	APC_GetFwVersion()	79
9.1.2.59	APC_GetGlobalGain()	79
9.1.2.60	APC_GetHidGyro()	80
9.1.2.61	APC_GetHWRegister()	80
9.1.2.62	APC_GetImage()	81
9.1.2.63	APC_GetImageInterrupt()	81
9.1.2.64	APC_GetInfoHidGyro()	82
9.1.2.65	APC_GetIRMaxValue()	83
9.1.2.66	APC_GetIRMinValue()	83
9.1.2.67	APC_GetIRMode()	84
9.1.2.68	APC_GetLogData()	84
9.1.2.69	APC_GetLutData()	85
9.1.2.70	APC_GetMultiBytesHWRegister()	85
9.1.2.71	APC_GetPidVid()	86
9.1.2.72	APC_GetPointCloud()	86
9.1.2.73	APC_GetPUPropVal()	87
9.1.2.74	APC_GetPURangeAndStep()	88
9.1.2.75	APC_GetRectifyLogData()	88
9.1.2.76	APC_GetRectifyMatLogData()	89
9.1.2.77	APC_GetRectifyTable()	89
9.1.2.78	APC_GetSensorRegister()	90
9.1.2.79	APC_GetSerialNumber()	91
9.1.2.80	APC_GetSRB()	91
9.1.2.81	APC_GetThermalFD()	91
9.1.2.82	APC_GetUACData()	92
9.1.2.83	APC_getUACNAME()	92
9.1.2.84	APC_GetUserData()	93
9.1.2.85	APC_GetYOffset()	93
9.1.2.86	APC_GetZDTable()	94

vi

9.1.2.87 APC_HoleFill()	94
9.1.2.88 APC_HoleFilled()	95
9.1.2.89 APC_ImgMirro() [1/2]	95
9.1.2.90 APC_ImgMirro() [2/2]	96
9.1.2.91 APC_Init()	96
9.1.2.92 APC_InitialCmdFiFo()	97
9.1.2.93 APC_InitialFlexibleGyro()	97
9.1.2.94 APC_InitialHidGyro()	97
9.1.2.95 APC_InitialUAC()	98
9.1.2.96 APC_InitPostProcess()	98
9.1.2.97 APC_InitSRB()	99
9.1.2.98 APC_InjectExtraDataToMp4()	99
9.1.2.99 APC_IsInterleaveDevice()	00
9.1.2.100 APC_IsMLBaseLine()	00
9.1.2.101 APC_OpenDevice()	00
9.1.2.102 APC_OpenDevice2()	<b>)</b> 1
9.1.2.103 APC_OpenDeviceMBL()	ງ2
9.1.2.104 APC_PostProcess()	03
9.1.2.105 APC_PutSRB()	Э4
9.1.2.106 APC_ReadCmdFiFo()	Э4
9.1.2.107 APC_ReadFlashData()	Э4
9.1.2.108 APC_RefreshDevice()	ე6
9.1.2.109 APC_Release()	ე6
9.1.2.110 APC_ReleaseFlexibleGyro()	)7
9.1.2.111 APC_ReleaseHidGyro()	Э7
9.1.2.112 APC_ReleasePostProcess()	Э7
9.1.2.113 APC_ReleaseSwPostProc()	38
9.1.2.114 APC_ReleaseUAC()	38
9.1.2.115 APC_ResetFilters()	38
9.1.2.116 APC_ResizeImgToHalf()	)9

CONTENTS vii

9.1.2.117 APC_RetrieveExtraDataFromMp4()
9.1.2.118 APC_RGB2BMP()
9.1.2.119 APC_RotateImg180() [1/2]
9.1.2.120 APC_RotateImg180() [2/2]
9.1.2.121 APC_RotateImg90() [1/2]
9.1.2.122 APC_RotateImg90() [2/2]
9.1.2.123 APC_SaveLutData()
9.1.2.124 APC_SelectDevice()
9.1.2.125 APC_SetAutoExposureMode()
9.1.2.126 APC_SetColorGain()
9.1.2.127 APC_SetControlCounterMode()
9.1.2.128 APC_SetCTPropVal()
9.1.2.129 APC_SetCurrentIRValue()
9.1.2.130 APC_SetDepthDataType()
9.1.2.131 APC_SetExposureTime()
9.1.2.132 APC_SetFWRegister()
9.1.2.133 APC_SetGlobalGain()
9.1.2.134 APC_SetHWRegister()
9.1.2.135 APC_SetIRMaxValue()
9.1.2.136 APC_SetIRMode()
9.1.2.137 APC_SetLogData()
9.1.2.138 APC_SetMultiBytesHWRegister()
9.1.2.139 APC_SetPidVid()
9.1.2.140 APC_SetPUPropVal()
9.1.2.141 APC_SetRectifyTable()
9.1.2.142 APC_SetSensorRegister()
9.1.2.143 APC_SetSensorTypeName()
9.1.2.144 APC_SetSerialNumber()
9.1.2.145 APC_Setup_v4l2_requestbuffers()
9.1.2.146 APC_SetupBlock()

viii CONTENTS

Index			139
	9.2.1	Detailed Description	138
9.2	eSPDI	/eSPDI_def.h File Reference	130
		9.1.2.158 APC_WriteWaveHeader()	130
		9.1.2.157 APC_WriteWaveEnd()	130
		9.1.2.156 APC_WriteFlashData()	129
		9.1.2.155 APC_WriteCmdFiFo()	129
		9.1.2.154 APC_TemporalFilter()	128
		9.1.2.153 APC_TableToData()	128
		9.1.2.152 APC_SwitchBaseline()	127
		9.1.2.151 APC_SubSample()	126
		9.1.2.150 APC_SetZDTable()	126
		9.1.2.149 APC_SetYOffset()	125
		9.1.2.148 APC_SetUserData()	125
		9.1.2.147 APC_SetupHidGyro()	124

# **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 programm demonstrating how to use the APIs defined in eSPDI.h
cfg	All files	configration files
	eSPDI.h	functions definitions
eSPDI	eSPDI_def.h	error/data type definitions
	eSPDI_←	SDK version declaration header
	version.h	
DMPreview	All files	a sample project demonstrating how to open
		multiple devices in an application

2 Introduction

## **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:

```
#ifdef 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 colorspaces 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\_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:

```
#ifdef 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 libpjeg 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 <a href="http://www.libjpeg-turbo.org/About/SmartScale">http://www.libjpeg-turbo.org/About/SmartScale</a> 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 D CT 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 <code>jpeg\_mem\_src()</code> and <code>jpeg\_mem\_dest()</code> 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 <code>-DWITH\_MEM\_SRCDST=0</code> to <code>cmake</code> prior to building libjpeg-turbo. This will restore the pre-1.3 behavior, in which <code>jpeg\_mem\_src()</code> and <code>jpeg\_mem\_dest()</code> 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 <code>jpeg\_mem\_src()</code> or <code>jpeg\_mem</code> <code>\_dest()</code>, 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 <code>jpeg\_mem\_src()</code> or <code>jpeg\_mem\_dest()</code>. Such is not the case on Windows. If a program is built against the libjpeg-turbo 1.3+ DLL and uses <code>jpeg\_mem\_src()</code> or <code>jpeg\_mem\_dest()</code>, 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 SI  $\leftarrow$  MD extensions be disabled, either by passing an argument of <code>-DWITH\_SIMD=0</code> to <code>cmake</code> when configuring the build or by setting the environment variable <code>JSIMD\_FORCENONE</code> 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:

```
#ifdef 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 colorspaces 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\_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:

```
#ifdef 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 libpjeg 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 <a href="http://www.libjpeg-turbo.org/About/SmartScale">http://www.libjpeg-turbo.org/About/SmartScale</a> 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 D CT 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 <code>jpeg\_mem\_src()</code> and <code>jpeg\_mem\_dest()</code> 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 <code>-DWITH\_MEM\_SRCDST=0</code> to <code>cmake</code> prior to building libjpeg-turbo. This will restore the pre-1.3 behavior, in which <code>jpeg\_mem\_src()</code> and <code>jpeg\_mem\_dest()</code> 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 <code>jpeg\_mem\_src()</code> or <code>jpeg\_mem</code> <code>\_dest()</code>, 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 <code>jpeg\_mem\_src()</code> or <code>jpeg\_mem\_dest()</code>. Such is not the case on Windows. If a program is built against the libjpeg-turbo 1.3+ DLL and uses <code>jpeg\_mem\_src()</code> or <code>jpeg\_mem\_dest()</code>, 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 SI  $\leftarrow$  MD extensions be disabled, either by passing an argument of <code>-DWITH\_SIMD=0</code> to <code>cmake</code> when configuring the build or by setting the environment variable <code>JSIMD\_FORCENONE</code> 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:

```
#ifdef 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 colorspaces 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\_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:

```
#ifdef 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 libpjeg 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 <a href="http://www.libjpeg-turbo.org/About/SmartScale">http://www.libjpeg-turbo.org/About/SmartScale</a> 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 D CT 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 <code>jpeg\_mem\_src()</code> and <code>jpeg\_mem\_dest()</code> 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 <code>-DWITH\_MEM\_SRCDST=0</code> to <code>cmake</code> prior to building libjpeg-turbo. This will restore the pre-1.3 behavior, in which <code>jpeg\_mem\_src()</code> and <code>jpeg\_mem\_dest()</code> 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 <code>jpeg\_mem\_src()</code> or <code>jpeg\_mem</code>—<code>dest()</code>, 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 <code>jpeg\_mem\_src()</code> or <code>jpeg\_mem\_dest()</code>. Such is not the case on Windows. If a program is built against the libjpeg-turbo 1.3+ DLL and uses <code>jpeg\_mem\_src()</code> or <code>jpeg\_mem\_dest()</code>, 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 SI  $\leftarrow$  MD extensions be disabled, either by passing an argument of <code>-DWITH\_SIMD=0</code> to <code>cmake</code> when configuring the build or by setting the environment variable <code>JSIMD\_FORCENONE</code> 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:

```
#ifdef 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 colorspaces 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\_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:

```
#ifdef 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 libpjeg 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 <a href="http://www.libjpeg-turbo.org/About/SmartScale">http://www.libjpeg-turbo.org/About/SmartScale</a> 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 D CT 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 <code>jpeg\_mem\_src()</code> and <code>jpeg\_mem\_dest()</code> 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 <code>-DWITH\_MEM\_SRCDST=0</code> to <code>cmake</code> prior to building libjpeg-turbo. This will restore the pre-1.3 behavior, in which <code>jpeg\_mem\_src()</code> and <code>jpeg\_mem\_dest()</code> 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 <code>jpeg\_mem\_src()</code> or <code>jpeg\_mem</code>—<code>dest()</code>, 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 <code>jpeg\_mem\_src()</code> or <code>jpeg\_mem\_dest()</code>. Such is not the case on Windows. If a program is built against the libjpeg-turbo 1.3+ DLL and uses <code>jpeg\_mem\_src()</code> or <code>jpeg\_mem\_dest()</code>, 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 SI  $\leftarrow$  MD extensions be disabled, either by passing an argument of <code>-DWITH\_SIMD=0</code> to <code>cmake</code> when configuring the build or by setting the environment variable <code>JSIMD\_FORCENONE</code> 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:

elerationTag	31
ClmageType	31
npassTag	32
Ctrl_RectLogData	32
oTag	37
xet_s	37
tCloudInfo	37
NPC_STREAM_INFO	
DEVINFORMATION	38
DEVSEL	38
(EEP_DATA_CTRL	38
'DTableInfo	39

28 Class Index

# **Chapter 7**

# File Index

# 7.1 File List

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

eSPDI/eSPDI.h	
Functions definitions	1
eSPDI/eSPDI_def.h	
Error/data type definitions	C
eSPDI/eSPDI version.h ?	?

30 File Index

# **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_MJPG,
    DEPTH_8BITS = 100, DEPTH_8BITS_0x80, DEPTH_11BITS, DEPTH_14BITS }
```

# **Static Public Member Functions**

- static bool **IslmageColor** (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:

32 Class Documentation

# 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{\bf ::} 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. |[0][1][2]||Xcl||[3][4][5]|*|Ycl|=> cl = left
```

camera coordinate | [6] [7] [8] | |Zcl|

34 Class Documentation

#### 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 rightt 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 OutlmgWidth

```
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
```

# Rectidied image crop column begin

```
8.4.1.15 RECT_Crop_Col_ED_L
unsigned short eSPCtrl_RectLogData::RECT_Crop_Col_ED_L
Rectidied image crop column end
8.4.1.16 RECT_Crop_Row_BG
unsigned short eSPCtrl_RectLogData::RECT_Crop_Row_BG
Rectidied image crop row begin
8.4.1.17 RECT_Crop_Row_ED
unsigned short eSPCtrl_RectLogData::RECT_Crop_Row_ED
Rectidied 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

36 Class Documentation

#### 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]

#### 8.4.1.27 RRotaMat

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

# 8.4.1.28 TranMat

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

Translation vector between the coordinate systems of the cameras. |[0]| |Xcr| |[1]| + |Ycr| => cr = right camera coordinate |[2]| |Zcr|

#### 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:

# 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

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

38 Class Documentation

# 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:

# 8.12 tagZDTableInfo Struct Reference

# **Public Attributes**

- int nlndex
- int nDataType

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

40 Class Documentation

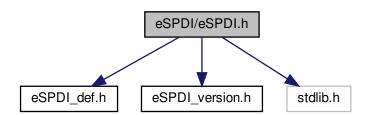
# **Chapter 9**

# **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 oncming 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 \*pDepthMerge
 Flag, 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\_SetControlCounterMode (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned char nValue) enable or disable interleave function
- int APC\_GetControlCounterMode (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned char \*n → Value)

enable or disable interleave function

- int APC\_GetSensorRegister (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, int nld, unsigned short address, unsigned short \*pValue, int flag, SENSORMODE\_INFO SensorMode)
- int APC\_SetSensorRegister (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, int nld, 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 \*pActual
 Length)

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

• int APC\_GetFwVersion (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, char \*pszFwVersion, int n← BufferSize, 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, B

 YTE \*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

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 \*pU

SB\_Port\_Type)

Get Device USB-port-type.

int APC\_GetDeviceResolutionList (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, int nMaxCount, AP←
 C\_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 bEP0MJPG, int nEP1Width, int nEP1Height, DEPTH\_TRANSFER\_CTRL dtc=DEPTH\_IMG\_NON\_T ← RANSFER, bool bIsOutputRGB24=false, void \*phWndNotice=0, int \*pFPS=0, CONTROL\_MODE cm=IM ← AGE SN NONSYNC)

the implement layer to open EYSD camera device by V4L2(https://en.wikipedia.org/wiki/Video4← Linux), 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 bEP0MJPG, int nEP1Width, int nEP1Height, DEPTH\_TRANSFER\_CTRL dtc=DEPTH\_IMG\_NON\_T ← RANSFER, bool bIsOutputRGB24=false, void \*phWndNotice=0, int \*pFPS=0, CONTROL\_MODE cm=IM ← AGE SN NONSYNC)

the implement layer to open EYSD camera device by V4L2(https://en.wikipedia.org/wiki/Video4← Linux), can open color and depth at one time call, do functions as below,

int APC\_OpenDeviceMBL (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, int nEP0Width, int nEP0←
Height, bool bEP0MJPG, 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. $\leftarrow$ 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 \*p↔
ImageSize, 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\_cusec)

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

• int APC\_SetupBlock (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, bool enable)

• 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

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 \*p

DepthImgBuf, 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 \*pColor ← ImgBuf, 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)

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 f
 ExpTimeMS)

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 \*pf
GlobalGain)

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 fGlobal
Gain)

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 \*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 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 nld, 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 nld, long int nValue) set camera terminal property values By v4l2\_control to set
- int APC\_GetPUPropVal (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, int nld, long int \*pValue) get processing unit property value by v4l2\_control to get processing unit(PU) property value
- int APC\_SetPUPropVal (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, int nld, long int nValue) set processing unit property value by v4l2\_control to set processing unit(PU) property value
- int APC\_GetCTRangeAndStep (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, int nld, 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:  $V4L2\_CID\_EXPOSURE\_AUTO\_V4L2\_CID\_EXPOSURE\_AUTO\_PRIORI \hookrightarrow TY V4L2\_CID\_EXPOSURE\_ABSOLUTE\_V4L2\_CID\_EXPOSURE\_V4L2\_CID\_FOCUS\_ABSOLUTE\_V4L2\_CID\_FOCUS\_AUTO\_V4L2\_CID\_IRIS\_ABSOLUTE\_V4L2\_CID\_IRIS\_RELATIVE\_V4L2\_CID\_IRIS\_RELATI$
- int APC\_GetPURangeAndStep (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, int nld, 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\_ $\leftrightarrow$  CID\_CONTRAST V4L2\_CID\_GAIN V4L2\_CID\_POWER\_LINE\_FREQUENCY V4L2\_CID\_HUE V4L2\_CID\_HUE  $\leftrightarrow$  AUTO V4L2\_CID\_SATURATION V4L2\_CID\_SHARPNESS V4L2\_CID\_GAMMA V4L2\_CID\_WHITE\_BALANCE\_ $\leftrightarrow$  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\_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 \*p
   — Data, 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 ldx, 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 Pixcel Depth Cancellation, just for EX8029.

 int APC\_FlyingDepthCancellation\_D11 (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned char \*pdepthD11, int width, int height)

Flying Pixcel 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 \*Gyro
Len)

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 inital 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 BMF

- int APC\_HoleFilled (unsigned short \*pDImgIn, unsigned short \*pDImgOut, int width, int height, int holeFilldiff)

  Hole Filled.
- int APC\_InitialCmdFiFo (const char \*pfifoName, int \*pFileDescrption, bool bRead)

Cmd FiFo Initial function.

int APC CloseCmdFiFo (int FileDescrption)

Cmd FiFo Close function.

• int APC WriteCmdFiFo (int FileDescrption, unsigned char \*pCmd, int len)

Write Cmd FiFo function.

int APC\_ReadCmdFiFo (int FileDescrption, unsigned char \*pBuf, int len)

Read Cmd FiFo function.

int APC\_InitSRB (void \*\*pSmbHandle, int QueueSize, char \*queueName)

Inital 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 \*p← PointCloudRGB, float \*pPointCloudXYZ, float Near, float Far)

get point cloud

• int APC\_ColorFormat\_to\_RGB24 (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned char \*Img← Dst, unsigned char \*ImgSrc, int SrcSize, int width, int height, APCImageType::Value type)

get hardware post processing status

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 bytes←
PerPixel, 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**

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

# Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.2 APC\_CloseCmdFiFo()

Cmd FiFo Close function.

# **Parameters**

int	FileDescrption File Description
-----	---------------------------------

# Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.3 APC\_CloseDevice()

close device and free resource

# **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.4 APC\_CloseDeviceEx()

close device and free resource for warm reset

# **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index

# Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.5 APC\_CloseDeviceMBL()

close Multiple Base Linedevice and free resource

# **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.6 APC\_ColorFormat\_to\_RGB24()

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

# get hardware post processing status

#### **Parameters**

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

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	char *depth_y depth data,
unsigned	char *rgb output data,
int	width image width,
int	height image height,

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

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

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

#### **Parameters**

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

# Returns

```
success: APC_OK, others: see eSPDI_def.h
```

#### 9.1.2.9 APC\_CreateSwPostProc()

create a software post process class

# **Parameters**

int	depthBits depth bit to set
void	**handle handle pointer to this software post process class

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.10 APC\_DecryptMP4()

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

#### **Parameters**

void*	pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
const	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]

# APC\_DecryptString.

#### **Parameters**

const	char* src input string
char*	dst output string (decrypted)

# Returns

success: APC\_OK, others:see eSPDI\_def.h

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

```
char * dst1,
char * dst2 )
```

# APC\_DecryptString.

#### **Parameters**

const	char* src input string
char*	dst1 output string #1 (decrypted)
char*	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**

void	*pHandleEYSD the pointer to the initilized EYSD SDK instance
1010	·
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	char** pDepthBufList [TODO]
float	*pDepthMergeOut [TODO]
unsigned	char *pDepthMergeFlag [TODO]
int	nDWidth [TODO]
int	nDHeight [TODO]
float	fFocus [TODO]
float	* pBaseline [TODO]
float	* pWRNear [TODO]
float	* pWRFar [TODO]
float	* pWRFusion [TODO]
int	nMergeNum [TODO]

# Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.14 APC\_DisableAE()

disable auto exposure(AE) function of ISP

#### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.15 APC\_DisableAWB()

disable auto white balance of ISP

#### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.16 APC\_DoFusion()

```
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**

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

# Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.17 APC\_DoSwPostProc()

# do software post process on a depth buffer

#### **Parameters**

void*	handle handle of this software post process class
-------	---

# **Parameters**

unsigned	char* colorBuf input color buffer
bool	isColorRgb24 is this color buffer RGB888
unsigned	char* depthBuf input depth buffer
unsigned	char* outputBuf output buffer
int	width image width
int	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**

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

# Returns

success: APC\_OK, others: see eSPDI\_def.h

# 9.1.2.19 APC\_EnableAE()

enable auto exposure(AE) function of ISP

# **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.20 APC\_EnableAWB()

enable auto white balance function of ISP

# **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	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 )
```

 ${\sf APC\_EnableGPUAcceleration}.$ 

#### **Parameters**

void	*pHandleEYSD the pointer to the initilized EYSD SDK instance
PDEVSELINFO	pDevSelInfo pointer of device select index
bool	enable enable it or not

# Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.22 APC\_EnableInterleave()

#### enable or disable interleave function

#### **Parameters**

pHandleEYSD	the pointer to the initilized EYSD SDK instance
pDevSelInfo	pointer of device select index
enable	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()

# enable or disable sensor IF

# **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
bool	blsEnable true is enable, false is disable

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.24 APC\_EncryptMP4()

# encrypt a H.264 video

# **Parameters**

void*	pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
const	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]

# APC\_EncryptString.

#### **Parameters**

const	char* src input string
char*	dst output string (encrypted)

# Returns

success: APC\_OK, others:see eSPDI\_def.h

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

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

## APC\_EncryptString.

### **Parameters**

const	char* src1 input string #1
const	char* src2 input string #2
char*	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**

```
void *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 Pixcel Depth Cancellation.

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	char *pdepthD11 point toinput depth buffer
int Generated by Doxygen	width depth width
int	height depth height

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.29 APC\_FlyingDepthCancellation\_D8()

Flying Pixcel Depth Cancellation, just for EX8029.

#### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	char *pdepthD8 point toinput depth buffer
int	width depth width
int	height depth height

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.30 APC\_GenerateLutFile()

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

void*	pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
const	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**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
BYTE	*pColorImgBuf buffer to store color image
BYTE	*pDepthImgBuf buffer to store depth image
unsigned	long int *pColorImageSize the actual color buffer size
unsigned	long int *pDepthImageSize the actual depth buffer size
int	*pColorSerial color serial number
int	*pDepthSerial depth serial number
int	nDepthDataType the depth data type, see definition in eSPDI_def.h

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

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

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

### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
BYTE	*pDepthImgBuf buffer to store image data
unsigned	long int *pImageSize the actual buffer size getting from device
int	*pDepthSerial the serial number for synchronizing depth image
int	nDepthDataType the depth data type, see definition in eSPDI_def.h

### Returns

success: APC\_OK, others: see eSPDI\_def.h

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

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

## **Parameters**

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

#### Returns

success: APC\_OK, others: see eSPDI\_def.h

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

```
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**

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

### Returns

success: APC\_OK, others: see eSPDI\_def.h

### 9.1.2.35 APC\_GetAccMeterValue()

get acc meter value

### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
int	*pX X posiztion
int	*pY Y posiztion
int	*pZ Z posiztion

# Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.36 APC\_GetAEStatus()

```
PDEVSELINFO pDevSelInfo,
PAE_STATUS pAEStatus )
```

get auto exposure(AE) is enabled or disable

### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
PAE_STATUS	pAEStatus see enum definition as to AE_STATUS in eSPDI_def.h

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

### 9.1.2.37 APC\_GetAutoExposureMode()

Get Auto Exposure Mode.

## **Parameters**

void*	pHandleEYSD handle.
PDEVSELINFO	pDevSelInfo pointer of device select index.
unsigned	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()

get auto white balance(AWB) is enabled or disable

### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
PAWB_STATUS	pAWBStatus see enum definition as to AWB_STATUS in eSPDI_def.h

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

### 9.1.2.39 APC\_GetBusInfo()

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

#### **Parameters**

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

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

### 9.1.2.40 APC\_GetColorGain()

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

### **Parameters**

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

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

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.42 APC\_GetColorImageWithTimestamp()

```
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**

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

### enable or disable interleave function

### **Parameters**

pHandleEYSD	the pointer to the initilized EYSD SDK instance
pDevSelInfo	pointer of device select index
*nValue	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()

```
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\_A⇔
UTO\_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\_REL↔
ATIVE 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**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
int	nld specifies the member of the property set, see CT Property ID defined in eSPDI_def.h
int	*pValue pointer of store CT property value

#### Returns

success: APC\_OK, others: see eSPDI\_def.h

### 9.1.2.45 APC\_GetCTRangeAndStep()

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\_PRIOR ITY V4L2\_CID\_EXPOSURE\_ABSOLUTE V4L2\_CID\_EXPOSURE V4L2\_CID\_FOCUS\_ABSOLUTE V4L2\_CID\_FOCUS\_AUTO V4L2\_CID\_IRIS\_ABSOLUTE V4L2\_CID\_IRIS\_RELATIVE V4L2\_CID\_IRIS\_RELATIVE V4L2\_CID\_ZOOM\_ABSOLUTE V4L2\_CID\_ZOOM\_RELATIVE V4L2\_CID\_PAN\_ABSOLUTE V4L2\_CID\_PAN\_RE LATIVE V4L2\_CID\_TILT\_ABSOLUTE V4L2\_CID\_TILT\_RELATIVE V4L2\_CID\_PRIVACY

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
int	nld specifies the member of the property set, see CT Property ID defined in eSPDI_def.h
long	int *pMax maximum value, inclusive. This field gives an upper bound for the control
long	int *pMin minimum value, inclusive. This field gives a lower bound for the control

### **Parameters**

long	int *pStep This field gives a step size for the control see enum https://www.← linuxtv.org/downloads/v4l-dvb-apis-old/vidioc-queryctrl.html how the step value is to be used for each possible control type. Note that this an unsigned 32-bit value
long	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.
long	<pre>int *pFlags control flags, see https://www.linuxtv.↔ org/downloads/v4l-dvb-apis-old/vidioc-queryctrl.html</pre>

## Returns

success: APC\_OK, others: see eSPDI\_def.h

### 9.1.2.46 APC\_GetCurrentlRValue()

get infrared radiation(IR) value of PUMA type IC

## **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short *pValue current 1 byte IR value setting

### Returns

success: APC\_OK, others: see eSPDI\_def.h

# 9.1.2.47 APC\_GetDepthDataType()

get current depth data type setting

### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
WORD	*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**

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

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.49 APC\_GetDepthImageWithTimestamp()

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

### **Parameters**

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

get informations of EYSD UVC devices, see DEVINFORMATION

## **Parameters**

void *pHandleEYSD handle	
PDEVSELINFO	pDevSelInfo pointer of device select index
DEVINFORMATION*	pdevinfo pointer of device information

### Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.51 APC\_GetDeviceInfoMBL\_15cm()

get informations of EYSD UVC devices, see DEVINFORMATION

### **Parameters**

void *pHandleEYSD handle	
PDEVSELINFO	pDevSelInfo pointer of device select index
DEVINFORMATION*	pdevinfo pointer of device information

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.52 APC\_GetDeviceNumber()

# get EYSD USB device numbers

### **Parameters**

void	*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

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

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.54 APC\_GetExposureTime()

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

### **Parameters**

void	*pHandleEYSD pHandleEYSD
PDEVSELINFO	pDevSelInfo pointer of device select index
int	nSensorMode which sensor(sensor A, B or Both) to get A is 0, B is 1, Both is 2
float	*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()

### getting gyro data function

void*	pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
int	length Gyro Data Length
unsigned	char *pGyroData pointer of Gyro Data.

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.56 APC\_GetFlexibleGyroLength()

getting length of gyro data function.

### **Parameters**

void*	pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short* GyroLen pointer of Gyro Data Lenhth.

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.57 APC\_GetFWRegister()

## get firmware register value

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short address register address
unsigned	short *pValue pointer of value got from register address
int	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()

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

#### **Parameters**

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

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

### 9.1.2.59 APC\_GetGlobalGain()

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

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

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.60 APC\_GetHidGyro()

## getting gyro data function

### **Parameters**

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

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.61 APC\_GetHWRegister()

## get hardware register value

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short address register address
unsigned	short *pValue pointer of value got from register address
int	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**

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

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.63 APC\_GetImageInterrupt()

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**

void*	pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	char *pCmdBuf pointer of Gyro Cmd Buffer.
int	cmdlength Command Lehgth.
unsigned	char *pResponseBuf pointer of ResponseBuffer.
int	resplength Response Length

## Returns

success: APC\_OK, others: see eSPDI\_def.h

# 9.1.2.65 APC\_GetIRMaxValue()

get maximum IR value of camera module

## **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short *pValue the maximum 1 byte IR value can be set

### Returns

success: APC\_OK, others: see eSPDI\_def.h

### 9.1.2.66 APC\_GetIRMinValue()

get minimum IR value of camera module

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned Generated by Doxygen	short *pValue the minimum 1 byte IR value can be set

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.67 APC\_GetIRMode()

## to check IR is turn on or off

### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	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
```

# 9.1.2.68 APC\_GetLogData()

## get log data from flash

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

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.69 APC\_GetLutData()

Read LUT parameters into the specified buffer.

### **Parameters**

void*	pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
BYTE*	buffer memory to store LUT data
int	nSize length of buffer in bytes

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.70 APC\_GetMultiBytesHWRegister()

## set hardware register

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short address register address
unsigned	char *Data multiple-bytes regigster value to set
int	size multiple-bytes regigster size
int	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
```

## 9.1.2.71 APC\_GetPidVid()

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

### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short *pPidBuf 4 byte buffer to store PID value
unsigned	short *pVidBuf 4 byte buffer to store VID value

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.72 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

void	*pHandleEYSD the pointer to the initilized EYSD SDK instance
PDEVSELINFO	pDevSelInfo pointer of device select index

### **Parameters**

unsigned	char ∗ImgColor RGB-buffer
int	CW ImgColor width
int	CH ImgColor height
unsigned	char *ImgDepth depth-buffer
int	DW ImgDepth width
int	DH ImgDepth height
PointCloudInfo	*pPointCloudInfo point-cloud information
unsigned	char *pPointCloudRGB point-cloud RGB value
float	*pPointCloudXYZ point-cloud XYZ value
float	Near filter range near dist.
float	Far filter range far dist.

### Returns

success: APC\_OK, others: see eSPDI\_def.h

### 9.1.2.73 APC\_GetPUPropVal()

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**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
int	nld specifies the member of the property set see PU property ID defined in eSPDI_def.h
long	int ∗pValue pointer of store PU property value

### Returns

success: APC\_OK, others: see eSPDI\_def.h

### 9.1.2.74 APC\_GetPURangeAndStep()

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\_HU CE\_AUTO V4L2\_CID\_SATURATION V4L2\_CID\_SHARPNESS V4L2\_CID\_GAMMA V4L2\_CID\_WHITE\_BALAN CE\_TEMPERATURE V4L2\_CID\_AUTO\_WHITE\_BALANCE

### **Parameters**

void	*pHandleEYSD handle	
PDEVSELINFO	pDevSelInfo pointer of device select index	
int	nld nld specifies the member of the property set, see CT Property ID defined in eSPDI_def.h	
long	int *pMax maximum value, inclusive. This field gives an upper bound for the control	
long	int *pMin minimum value, inclusive. This field gives a lower bound for the control	
long	int *pStep This field gives a step size for the control see enum https://www.← linuxtv.org/downloads/v4l-dvb-apis-old/vidioc-queryctrl.html how the step value is to be used for each possible control type. Note that this an unsigned 32-bit value	
long	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.	
long	<pre>int *pFlags control flags, see https://www.linuxtv.↔ org/downloads/v4l-dvb-apis-old/vidioc-queryctrl.html</pre>	

### Returns

success: APC\_OK, others: see eSPDI\_def.h

# 9.1.2.75 APC\_GetRectifyLogData()

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

### **Parameters**

void	*pHandleEYSD handle	
PDEVSELINFO	pDevSelInfo pointer of device select index	
eSPCtrl_RectLogData	*pData 4096 bytes of rectify log data, see eSPCtrl_RectLogData for detailed members	
index,user	data section from 0 $\sim$ 9	

### Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.76 APC\_GetRectifyMatLogData()

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

### **Parameters**

void	*pHandleEYSD handle	
PDEVSELINFO	pDevSelInfo pointer of device select index	
eSPCtrl_RectLogData	*pData 4096 bytes of rectify log data, see eSPCtrl_RectLogData for detailed members	
index,user	data section from 0 $\sim$ 9	

### Returns

success: APC\_OK, others: see eSPDI\_def.h

### 9.1.2.77 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**

void	*pHandleEYSD handle	
PDEVSELINFO	pDevSelInfo pointer of device select index	
BYTE	*buffer buffer to store rectify table data	
int	BufferLength input buffer length, must be 1024	
int	*pActualLength actual length has written to buffer	
int	index index(from 0 $\sim$ 9) to identify rectify table for corresponding depth	

### Returns

```
success:APC_OK, others: see eSPDI_def.h
```

## 9.1.2.78 APC\_GetSensorRegister()

## get value from sensor register

### **Parameters**

void	*pHandleEYSD handle	
PDEVSELINFO	pDevSelInfo pointer of device select index	
int	nld sensor slave address see Videodevice.h for sensor slave address setting	
unsigned	short address register address	
unsigned	short *pValue pointer of value got from register address	
int	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	
SENSORMODE_INFO	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

## 9.1.2.79 APC\_GetSerialNumber()

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

### get device serial number

### **Parameters**

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

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

### 9.1.2.80 APC\_GetSRB()

### Get Packet from SRB.

## Parameters

void	*pSrbHandle pointer to SRB class
packet⊷	*pPacket Input Packet
_ <b>s</b>	

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.81 APC\_GetThermalFD()

# get file description of thermal device

### **Parameters**

void	oid *pHandleEYSD handle	
int	*p_FD file description of thermal device	

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.82 APC\_GetUACData()

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

UAC inital function.

### **Parameters**

unsigned	char *buffer pointer of UAC buffer
int	length UAC buffer length

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.83 APC\_getUACNAME()

## Get EYSD UAC Name.

### **Parameters**

char	*input Point to device Address.	
char	*output Point to device Name.	

### Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.84 APC\_GetUserData()

### get user data from flash

### **Parameters**

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

### Returns

success: APC\_OK, others: see eSPDI\_def.h

### 9.1.2.85 APC\_GetYOffset()

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

#### **Parameters**

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

## Returns

success: APC\_OK, others: see eSPDI\_def.h

### 9.1.2.86 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**

void	*pHandleEYSD handle	
PDEVSELINFO	pDevSelInfo pointer of device select index	
BYTE	*buffer bufer to store ZD table	
int	BufferLength input buffer length	
int	*pActualLength actual length has written to buffer	
PZDTABLEINFO pZDTableInfo index to identify ZD table and data type for corrresponding de		

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.87 APC\_HoleFill()

## APC\_HoleFill.

void	*pHandleEYSD the pointer to the initilized EYSD SDK instance
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	char∗ depthBuf depth buffer pointer
int	bytesPerPixel byte number of one pixel
int	kernel_size [TODO]
int	width depth width
int	height depth height
int	level [TODO]
bool	horizontal [TODO]

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.88 APC\_HoleFilled()

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

### Hole Filled.

### **Parameters**

unsigned	short *pDImgIn Image Input	
unsigned	short *pDImgOut Image Output	
int	width image width	
int	height image height	
int holeFilldiff Hole filled strangth, value from 0 to 2047		

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## **9.1.2.89** APC\_ImgMirro() [1/2]

# Make the image to Mirro.

APCImageType::Value	imgType Image Type
int	width image width
int	height image height
unsigned	char *src image source
unsigned	char *dstBuf image desteration

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

Make the image to Mirro.

#### **Parameters**

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

### Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.91 APC\_Init()

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

**ppHandleEYSD	a pointer of pointer to access CEYSD class	
blsLogEnabled	generate log or not	

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.92 APC\_InitialCmdFiFo()

Cmd FiFo Initial function.

#### **Parameters**

const	char *pfifoName Point to the cmd fifo name *pFileDescrption Point to the file description	
int		
bRead Indicate Read or Write Cmd fifo		

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.93 APC\_InitialFlexibleGyro()

## gyro sensor inital function

### **Parameters**

void*	pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

# 9.1.2.94 APC\_InitialHidGyro()

## gyro sensor inital function

### **Parameters**

void*	pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index

### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.95 APC\_InitialUAC()

### UAC inital function.

### **Parameters**

	char	*deviceName Point to device Name.	
--	------	-----------------------------------	--

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.96 APC\_InitPostProcess()

## $APC\_InitPostProcess.$

void	**ppPostProcessHandle [TODO]
unsigned	int nWidth [TODO]
unsigned	int nHeight [TODO]
APCImageType::Value	imageType [TODO]

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.97 APC\_InitSRB()

## Inital the SRB(Share Ring Buffering)

## **Parameters**

void	**pSrbHandle a pointer of pointer to SRB class
int	QueueSize
char	srbName SRM Name

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.98 APC\_InjectExtraDataToMp4()

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

## $APC\_Inject ExtraData To Mp4.$

#### **Parameters**

void*	pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
const	char *filename input video file name
const	char *data video data
const	int dataLen video data length

## Returns

```
success: APC_OK, others:see eSPDI_def.h
```

## 9.1.2.99 APC\_IsInterleaveDevice()

check module support interleave function or not

#### **Parameters**

pHandleEYSD	the pointer to the initilized EYSD SDK instance
pDevSelInfo	pointer of device select index

#### Returns

true: support interleave, false: not support

## 9.1.2.100 APC\_IsMLBaseLine()

Check the device is multiple baseline device.

## **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index

## Returns

true: multiplies baseline device, false: normally device.

## 9.1.2.101 APC\_OpenDevice()

```
int APC_OpenDevice (
    void * pHandleEYSD,
    PDEVSELINFO pDevSelInfo,
    int nEPOWidth,
    int nEPOHeight,
    bool bEPOMJPG,
    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/ $\leftarrow$  Video4Linux), can open color and depth at one time call, do functions as below,

- initialize the USB device by V4L2 protocol 1.1 query device v4l2 capability 1.2 must have video capability
   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**

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

#### **Returns**

success: APC\_OK, others: see eSPDI\_def.h

#### 9.1.2.102 APC\_OpenDevice2()

```
int APC_OpenDevice2 (
    void * pHandleEYSD,
    PDEVSELINFO pDevSelInfo,
    int nEPOWidth,
    int nEPOHeight,
    bool bEPOMJPG,
    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/ $\leftarrow$  Video4Linux), can open color and depth at one time call, do functions as below,

- initialize the USB device by V4L2 protocol 1.1 query device v4l2 capability 1.2 must have video capability
   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**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
int	nEP0Width width of endpoint1(color) resolution
int	nEP0Height height of endpoint1(color) resolution
bool	bEP0MJPG endpoint1 output is MJPEG?
int	nEP1Width width of endpoint2(depth) resolution
int	nEP1Height height of endpoint2(depth) resolution
DEPTH_TRANSFER_CTRL	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**

bool	blsOutputRGB24 output color image is RGB format
void	*phWndNotice reserved, not use
int	*pFPS input frame rate setting
CONTROL_MODE	cm reserved, not use

#### Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.103 APC\_OpenDeviceMBL()

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

- initialize the USB device by V4L2 protocol 1.1 query device v4l2 capability 1.2 must have video capability
   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**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
int	nEP0Width width of endpoint1(color) resolution
int	nEP0Height height of endpoint1(color) resolution
bool	bEP0MJPG endpoint1 output is MJPEG?
int	nEP1Width width of endpoint2(depth) resolution
int	nEP1Height height of endpoint2(depth) resolution
DEPTH_TRANSFER_CTRL	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**

bool	blsOutputRGB24 output color image is RGB format
void	*phWndNotice reserved, not use
int	*pFPS input frame rate setting
CONTROL_MODE	cm reserved, not use

#### Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.104 APC\_PostProcess()

## APC\_PostProcess.

## **Parameters**

l	void	*ppPostProcessHandle [TODO]
	unsigned	char *pDepthData [TODO]

## Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.105 APC\_PutSRB()

## Put Packet to SRB.

#### **Parameters**

void	*pSrbHandle pointer to SRB class
packet←	*pPacket Input Packet
_ <b>s</b>	

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.106 APC\_ReadCmdFiFo()

## Read Cmd FiFo function.

## Parameters

int	FileDescrption File description
unsigned	char *pCmd Point to the cmd buffer
int	lenIndicate the cmd lemgth.

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.107 APC\_ReadFlashData()

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**

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

#### Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.108 APC\_RefreshDevice()

```
int APC_RefreshDevice (
     void * pHandleEYSD )
```

refresh all EYSD UVC devices

#### **Parameters**

void *pHandleEYSD har	ıdle
-----------------------	------

## Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.109 APC\_Release()

```
void APC_Release (
     void ** ppHandleEYSD )
```

release resource that APC\_Init had allocated

## **Parameters**

void | \*\*ppHandleEYSD array of CEYSD class handlers

#### Returns

none

## 9.1.2.110 APC\_ReleaseFlexibleGyro()

gyro sensor release function

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.111 APC\_ReleaseHidGyro()

gyro sensor release function

#### **Parameters**

void*	pHandleEYSD handle	
PDEVSELINFO	pDevSelInfo pointer of device select index	

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.112 APC\_ReleasePostProcess()

```
\label{eq:apc_releasePostProcess} \mbox{ APC\_ReleasePostProcess} \mbox{ (} \\ \mbox{void} \ * \ pPostProcessHandle \ )
```

APC\_ReleasePostProcess.

## **Parameters**

void	*ppPostProcessHandle [TODO]
------	-----------------------------

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.113 APC\_ReleaseSwPostProc()

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

release a software post process class

## **Parameters**

void**	handle handle pointer to this software post process class
--------	---

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.114 APC\_ReleaseUAC()

```
int APC_ReleaseUAC (
     void )
```

UAC inital function.

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.115 APC\_ResetFilters()

APC\_ResetFilters.

## **Parameters**

void	*pHandleEYSD the pointer to the initilized EYSD SDK instance	
PDEVSELINFO	pDevSelInfo pointer of device select index	

## Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.116 APC\_ResizeImgToHalf()

## Resize the image to half.

#### **Parameters**

APCImageType::Value	mgType Image Type
int	width image width
int	height image height
unsigned	char *src image source
unsigned	char *dst image desteration
int	len desteration buffer length

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.117 APC\_RetrieveExtraDataFromMp4()

## $APC\_Retrieve ExtraData From Mp4.$

## **Parameters**

void*	pHandleEYSD handle	
PDEVSELINFO	pDevSelInfo pointer of device select index	
const	char *filename input video file name	
const	char *data video data	
const	int dataLen video data length	

## Returns

success: APC\_OK, others:see eSPDI\_def.h

## 9.1.2.118 APC\_RGB2BMP()

## RGB to BMP.

## **Parameters**

*filename	Ouput BMP file name
int	width image width
int	height image height
*data	input RGB buffer.

## Returns

success: APC\_OK, others: see eSPDI\_def.h

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

Rotate the image to 180 degree.

## **Parameters**

APCImageType::Value	mgType Image Type
int	width image width
int	height image height
unsigned	char *src image source
unsigned	char *dstBuf image desteration
int	len desteration buffer length

## Returns

success: APC\_OK, others: see eSPDI\_def.h

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

Rotate the image to 180 degree.

#### **Parameters**

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

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## **9.1.2.121** APC\_RotateImg90() [1/2]

Rotate the image to 90 degree.

## **Parameters**

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

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

bool clockwise )

Make the image to rotate.

#### **Parameters**

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

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.123 APC\_SaveLutData()

Save LUT parameters in the specified file.

#### **Parameters**

void*	pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
const	char* filename output LUT file name

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.124 APC\_SelectDevice()

do not support currently

## Returns

APC\_NotSupport

## 9.1.2.125 APC\_SetAutoExposureMode()

Setup Auto Exposure Mode.

#### **Parameters**

void*	pHandleEYSD handle.
PDEVSELINFO	pDevSelInfo pointer of device select index.
unsigned	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
```

## 9.1.2.126 APC\_SetColorGain()

```
float fGainG,
float fGainB )
```

set color gain of ISP

## **Parameters**

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

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.127 APC\_SetControlCounterMode()

#### enable or disable interleave function

#### **Parameters**

pHandleEYSD	the pointer to the initilized EYSD SDK instance	
pDevSelInfo	pointer of device select index	
nValue	0: Frame Counter Mode, 1: Serial Counter Mode,	

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.128 APC\_SetCTPropVal()

set camera terminal property values By v4l2\_control to set

## **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
Geppprated by Doxygen	nld specifies the member of the property set see CT Property ID defined in eSPDI_def.h
long	int nValue CT property value to set

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.129 APC\_SetCurrentIRValue()

set infrared radiation(IR) value of PUMA type IC

## **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short nValue 1 byte IR value to set

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.130 APC\_SetDepthDataType()

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

## **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short nValue depth data type you want to set, see APC_DEPTH_DATA_xxx in eSPDI_def.h

## Returns

success: APC\_OK, others: see eSPDI\_def.h

#### 9.1.2.131 APC\_SetExposureTime()

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**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
int	nSensorMode which sensor(sensor A, B or Both) to set A is 0, B is 1, Both is 2
float	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
```

## 9.1.2.132 APC\_SetFWRegister()

set firmware register value

#### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short address register address
unsigned	short nValue register value to set
int	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
```

## 9.1.2.133 APC\_SetGlobalGain()

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

#### **Parameters**

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

#### Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.134 APC\_SetHWRegister()

## set hardware register

#### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short address register address
unsigned	short nValue register value to set
int	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

## 9.1.2.135 APC\_SetIRMaxValue()

#### get maximum IR value of camera module

#### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short nValue the IR maximum setting value

#### Returns

success: APC\_OK, others: see eSPDI\_def.h

#### 9.1.2.136 APC\_SetIRMode()

## enable or disable IRs

## **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	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

## 9.1.2.137 APC\_SetLogData()

```
BYTE * buffer,
int BufferLength,
int * pActualLength,
int index )
```

## set log data to flash

## **Parameters**

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

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.138 APC\_SetMultiBytesHWRegister()

## set hardware register

#### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short address register address
unsigned	char *Data multiple-bytes regigster value to set
int	size multiple-bytes regigster size
int	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

## 9.1.2.139 APC\_SetPidVid()

#### set PID and VID to device

#### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	short *pPidBuf 4 byte PID value buffer to set
unsigned	short *pVidBuf 4 byte VID value buffer to set

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.140 APC\_SetPUPropVal()

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

#### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
int	nld specifies the member of the property set see PU Property ID defined in eSPDI_def.h
int	nValue value to set

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.141 APC\_SetRectifyTable()

```
PDEVSELINFO pDevSelInfo,
BYTE * buffer,
int BufferLength,
int * pActualLength,
int index )
```

## set rectify values to flash

#### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
BYTE	*buffer rectify values to set
int	BufferLength bufer length, must be 1024
int	*pActualLength always return 1024
int	index index(from 0 $\sim$ 9) to identify rectify table for corresponding depth

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.142 APC\_SetSensorRegister()

## set sensor register value

## **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
int	nld sensor slave address see Videodevice.h for sensor slave address setting
unsigned	short address register address
unsigned	short nValue value to set
int	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
SENSORMODE_INFO	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
```

## 9.1.2.143 APC\_SetSensorTypeName()

set the sensor type you want to work on

## **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
SENSOR_TYPE_NAME	stn which sensor you want to work on

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.144 APC\_SetSerialNumber()

set serial number to device

## **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
BYTE	*pData pointer of buffer to store serial number, it is WildChar
int	nLen pData length in byte

## Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.145 APC\_Setup\_v4l2\_requestbuffers()

Setup v4l2 request buffers, default = 4.

#### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
int	cnt Should be >= 0

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.146 APC\_SetupBlock()

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

#### **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
bool	enable Enable the Blocking mode or not)

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.147 APC\_SetupHidGyro()

#### getting gyro data function

#### **Parameters**

void*	pHandleEYSD handle	
PDEVSELINFO	pDevSelInfo pointer of device select index	
unsigned	char *pGyroData pointer of Gyro Data Buffer.	
int	length Input buffer Length, shoul	

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.148 APC\_SetUserData()

#### set user data to flash

## **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
BYTE	*buffer user buffer data to set
int	BufferLength buffer length to write
USERDATA_SECTION_INDEX	usi which user section data to set

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.149 APC\_SetYOffset()

#### set Y offset values

## **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
BYTE	*buffer buffer data to set
int	BufferLength buffer length
int	*pActualLength always return 256
int	index index value to file ID 30

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.150 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**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
BYTE	*buffer ZD values to set
int	BufferLength corresponding length of ZD table in buffer
int	*pActualLength buffer lenth written to flash, should be same as BufferLength
PZDTABLEINFO	pZDTableInfo index and depth type of this ZD

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.151 APC\_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**

void	*pHandleEYSD the pointer to the initilized EYSD SDK instance
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	char **SubSample [TODO]
unsigned	char *depthBuf depth buffer pointer
int	bytesPerPixel byte number of one pixel
int	width depth width
int	height depth height
int&	new_width new depth width
int&	new_height new depth height
int	mode [TODO]
int	factor [TODO]

## Returns

success: APC OK, others: see eSPDI def.h

## 9.1.2.152 APC\_SwitchBaseline()

Swich the baseline index.

## **Parameters**

```
int index Baseline index 1: 30 mm 2: 60 mm 3: 150 mm
```

## Returns

success: APC\_OK, others: see eSPDI\_def.h

## 9.1.2.153 APC\_TableToData()

## transfer Src to Dst by Table

#### **Parameters**

void	*pHandleEYSD the pointer to the initilized EYSD SDK instance
PDEVSELINFO	pDevSelInfo pointer of device select index
int	width input image width
int	height input image height
int	TableSize input Table size in bytes
unsigned	short *Table input Table buffer
unsigned	short *Src input Src buffer
unsigned	short *Dst output Dst buffer

## Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.154 APC\_TemporalFilter()

```
int APC_TemporalFilter (
    void * pHandleEYSD,
    PDEVSELINFO pDevSelInfo,
    unsigned char * depthBuf,
    int bytesPerPixel,
    int width,
    int height,
    float alpha,
    int history )
```

## $APC\_Temporal Filter.$

## **Parameters**

void	*pHandleEYSD the pointer to the initilized EYSD SDK instance
PDEVSELINFO	pDevSelInfo pointer of device select index
unsigned	char∗ depthBuf depth buffer pointer
int	bytesPerPixel byte number of one pixel
int	width depth width
int	height depth height
float	alpha [TODO]
int	history [TODO]

Generated by Doxygen

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.155 APC\_WriteCmdFiFo()

Write Cmd FiFo function.

#### **Parameters**

int	FileDescrption File description	
unsigned	char *pCmd Point to the cmd buffer	
int	lenIndicate the cmd lemgth.	

#### Returns

success: APC\_OK, others: see eSPDI\_def.h

#### 9.1.2.156 APC\_WriteFlashData()

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**

void	*pHandleEYSD CEronDI class
PDEVSELINFO	pDevSelInfo pointer of device select index
FLASH_DATA_TYPE	fdt segment type of flash be wrote
BYTE	*pBuffer buffer of firmware code
unsigned	long int BufferLength Buffer length to be wrote
BOOL	blsDataVerify write data verification flag, if true this function will read data again and do a byte to byte comparison
KEEP_DATA_CTRL	kdc keep function flags

Generated by Doxygen

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.157 APC\_WriteWaveEnd()

```
int APC_WriteWaveEnd ( \label{eq:approx} \mbox{int } fd, \\ \mbox{size\_t } length \mbox{)}
```

Modified Wave Header.

## **Parameters**

```
int fd wave file descript.
```

#### Returns

```
success: APC_OK, others: see eSPDI_def.h
```

## 9.1.2.158 APC\_WriteWaveHeader()

Write Wave Header.

## **Parameters**

```
int fd wave file descript.
```

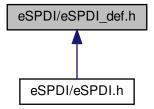
## Returns

```
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)))</li>
- #define BIT\_FLIP(a, b) ((a) ^= (1<<(b)))</li>
- #define BIT\_CHECK(a, b) ((a) & (1<<(b)))</li>
- #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 0 /\* raw (depth off, only 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 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 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\_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\_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 + AP← C 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\_D ← EFAULT + 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\_B
   ITS + 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\_8\_BITS\_RAW (APC\_DEPTH\_DATA\_SCALE\_DOW ← N\_8\_BITS\_RAW + APC\_DEPTH\_DATA\_INTERLEAVE\_MODE\_OFFSET) /\* raw \*/
- #define APC\_DEPTH\_DATA\_SCALE\_DOWN\_ILM\_14\_BITS\_RAW (APC\_DEPTH\_DATA\_SCALE\_DO

  WN\_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\_DO

  WN\_11\_BITS\_RAW + APC\_DEPTH\_DATA\_INTERLEAVE\_MODE\_OFFSET) /\* raw \*/
- #define APC\_DEPTH\_DATA\_SCALE\_DOWN\_ILM\_14\_BITS\_COMBINED\_RECTIFY (APC\_DEPTH\_D ← ATA\_SCALE\_DOWN\_14\_BITS\_COMBINED\_RECTIFY + APC\_DEPTH\_DATA\_INTERLEAVE\_MODE\_O ← FFSET)
- #define APC\_DEPTH\_DATA\_SCALE\_DOWN\_ILM\_11\_BITS\_COMBINED\_RECTIFY (APC\_DEPTH\_D ↔ ATA\_SCALE\_DOWN\_11\_BITS\_COMBINED\_RECTIFY + APC\_DEPTH\_DATA\_INTERLEAVE\_MODE\_O ↔ FFSET)
- #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\_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 17
- #define POSTPAR HR CURVE 28
- #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 DEVICE\_TYPE { OTHERS = 0, AXES1, PUMA, KIWI }
- 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\_SEC $\leftrightarrow$  TION\_3,
  - USERDATA\_SECTION\_4, USERDATA\_SECTION\_5, USERDATA\_SECTION\_6, USERDATA\_SECTIO↔ N 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, A↔
 PC 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 UNKOWN = 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_U←

 NKNOW }
• enum SENSITIVITY_LEVEL_L3G { DPS_245 = 0, DPS_500, DPS_2000 }

    enum SENSITIVITY LEVEL LSM {

 _{2}G = 0, _{4}G, _{6}G, _{8}G,
 _16G }
enum OUTPUT DATA RATE {
 \label{eq:one_Shot} \textbf{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, 63
eSPDI.h, 50	APC_FlyingDepthCancellation_D11
APC_CloseCmdFiFo	eSPDI.h, 63
eSPDI.h, 51	APC_FlyingDepthCancellation_D8
APC_CloseDevice	eSPDI.h, 64
eSPDI.h, 51	APC_GenerateLutFile
APC_CloseDeviceEx	eSPDI.h, 64
eSPDI.h, 52	APC_Get2Image
APC CloseDeviceMBL	eSPDI.h, 65
eSPDI.h, 52	APC_Get_150_mm_depth
APC_ColorFormat_to_RGB24	eSPDI.h, 65
eSPDI.h, 52	APC_Get_60_mm_depth
APC_Convert_Depth_Y_To_Buffer	eSPDI.h, 66
eSPDI.h, 53	APC_Get_Color_30_mm_depth
APC_Convert_Depth_Y_To_Buffer_offset	eSPDI.h, 66
eSPDI.h, 54	APC_GetAEStatus
APC CreateSwPostProc	eSPDI.h, 67
eSPDI.h, 54	APC GetAWBStatus
APC_DecryptMP4	eSPDI.h, 68
eSPDI.h, 55	APC_GetAccMeterValue
APC DecryptString	eSPDI.h, 67
eSPDI.h, 55	APC_GetAutoExposureMode
APC_DepthMerge	eSPDI.h, 68
eSPDI.h, 56	APC GetBusInfo
APC_DisableAWB	eSPDI.h, 69
eSPDI.h, 57	APC_GetCTPropVal
APC_DisableAE	eSPDI.h, 71
eSPDI.h, 57	APC_GetCTRangeAndStep
APC_DoFusion	eSPDI.h, 72
eSPDI.h, 57	APC_GetColorGain
APC_DoSwPostProc	eSPDI.h, 69
eSPDI.h, 58	APC_GetColorImage
APC_EdgePreServingFilter	eSPDI.h, 70
eSPDI.h, 59	APC_GetColorImageWithTimestamp
APC_EnableAWB	eSPDI.h, 70
eSPDI.h, 60	APC_GetControlCounterMode
APC_EnableAE	eSPDI.h, 71
eSPDI.h, 59	APC_GetCurrentIRValue
APC_EnableGPUAcceleration	eSPDI.h, 73
eSPDI.h, 60	APC_GetDepthDataType
APC_EnableInterleave	eSPDI.h, 73
eSPDI.h, 61	APC_GetDepthImage
APC_EnableSensorIF	eSPDI.h, 74
eSPDI.h, 61	APC_GetDepthImageWithTimestamp
APC_EncryptMP4	eSPDI.h, 74
eSPDI.h, 62	APC_GetDeviceInfo
APC_EncryptString	eSPDI.h, 75
eSPDI.h, 62	APC_GetDeviceInfoMBL_15cm
APC FindDevice	eSPDI.h. 75

APC_GetDeviceNumber	APC_GetThermalFD
eSPDI.h, 76	eSPDI.h, 91
APC GetDeviceResolutionList	APC GetUACData
eSPDI.h, 76	eSPDI.h, 92
APC_GetExposureTime	APC GetUserData
eSPDI.h, 77	eSPDI.h, 92
APC_GetFWRegister	APC_GetYOffset
eSPDI.h, 78	eSPDI.h, 93
APC_GetFlexibleGyroData	APC_GetZDTable
eSPDI.h, 77	eSPDI.h, 93
APC_GetFlexibleGyroLength	APC_HoleFill
eSPDI.h, 78	eSPDI.h, <mark>94</mark>
APC_GetFwVersion	APC_HoleFilled
eSPDI.h, 79	eSPDI.h, 95
APC_GetGlobalGain	APC_ImgMirro
eSPDI.h, 79	eSPDI.h, 95, 96
APC_GetHWRegister	APC_Init
eSPDI.h, 80	eSPDI.h, 96
APC_GetHidGyro	APC_InitPostProcess
eSPDI.h, 80	eSPDI.h, 98
APC_GetIRMaxValue	APC_InitSRB
eSPDI.h, 83	eSPDI.h, 99
APC_GetIRMinValue	APC_InitialCmdFiFo
eSPDI.h, 83	eSPDI.h, 97
APC_GetIRMode	APC_InitialFlexibleGyro
eSPDI.h, 84	eSPDI.h, 97
APC_GetImage	APC_InitialHidGyro
eSPDI.h, 81	eSPDI.h, 97
APC_GetImageInterrupt	APC_InitialUAC
eSPDI.h, 81	eSPDI.h, 98
APC_GetInfoHidGyro	APC_InjectExtraDataToMp4
eSPDI.h, 81	eSPDI.h, 99
APC_GetLogData	APC_IsInterleaveDevice
eSPDI.h, 84	eSPDI.h, 99
APC_GetLutData	APC_IsMLBaseLine
eSPDI.h, 85	eSPDI.h, 100
APC_GetMultiBytesHWRegister	APC_OpenDevice
eSPDI.h, 85	eSPDI.h, 100
APC_GetPUPropVal	APC_OpenDevice2
eSPDI.h, 87	eSPDI.h, 101
APC_GetPURangeAndStep	APC_OpenDeviceMBL
eSPDI.h, 87	eSPDI.h, 102
APC GetPidVid	APC PostProcess
eSPDI.h, 86	eSPDI.h, 103
APC GetPointCloud	APC PutSRB
eSPDI.h, 86	eSPDI.h, 103
APC_GetRectifyLogData	
eSPDI.h, 88	APC RGR2RMP
	APC_RGB2BMP
APC CotPootifyMotLogPoto	eSPDI.h, 109
APC_GetRectifyMatLogData	eSPDI.h, 109 APC_ReadCmdFiFo
eSPDI.h, 89	eSPDI.h, 109 APC_ReadCmdFiFo eSPDI.h, 104
eSPDI.h, 89 APC_GetRectifyTable	eSPDI.h, 109 APC_ReadCmdFiFo eSPDI.h, 104 APC_ReadFlashData
eSPDI.h, 89 APC_GetRectifyTable eSPDI.h, 89	eSPDI.h, 109 APC_ReadCmdFiFo eSPDI.h, 104 APC_ReadFlashData eSPDI.h, 104
eSPDI.h, 89 APC_GetRectifyTable	eSPDI.h, 109 APC_ReadCmdFiFo eSPDI.h, 104 APC_ReadFlashData
eSPDI.h, 89 APC_GetRectifyTable eSPDI.h, 89	eSPDI.h, 109  APC_ReadCmdFiFo     eSPDI.h, 104  APC_ReadFlashData     eSPDI.h, 104  APC_RefreshDevice     eSPDI.h, 106
eSPDI.h, 89 APC_GetRectifyTable eSPDI.h, 89 APC_GetSRB	eSPDI.h, 109 APC_ReadCmdFiFo eSPDI.h, 104 APC_ReadFlashData eSPDI.h, 104 APC_RefreshDevice
eSPDI.h, 89  APC_GetRectifyTable     eSPDI.h, 89  APC_GetSRB     eSPDI.h, 91	eSPDI.h, 109  APC_ReadCmdFiFo     eSPDI.h, 104  APC_ReadFlashData     eSPDI.h, 104  APC_RefreshDevice     eSPDI.h, 106
eSPDI.h, 89  APC_GetRectifyTable     eSPDI.h, 89  APC_GetSRB     eSPDI.h, 91  APC_GetSensorRegister	eSPDI.h, 109 APC_ReadCmdFiFo eSPDI.h, 104 APC_ReadFlashData eSPDI.h, 104 APC_RefreshDevice eSPDI.h, 106 APC_Release
eSPDI.h, 89  APC_GetRectifyTable     eSPDI.h, 89  APC_GetSRB     eSPDI.h, 91  APC_GetSensorRegister     eSPDI.h, 90	eSPDI.h, 109  APC_ReadCmdFiFo eSPDI.h, 104  APC_ReadFlashData eSPDI.h, 104  APC_RefreshDevice eSPDI.h, 106  APC_Release eSPDI.h, 106

APC_ReleaseHidGyro	APC_SetSensorTypeName
eSPDI.h, 107	eSPDI.h, 123
APC_ReleasePostProcess	APC_SetSerialNumber
eSPDI.h, 107	eSPDI.h, 123
APC_ReleaseSwPostProc	APC_SetUserData
eSPDI.h, 107	eSPDI.h, 125
APC_ReleaseUAC	APC_SetYOffset
eSPDI.h, 108	eSPDI.h, 125
APC_ResetFilters	APC_SetZDTable
eSPDI.h, 108	eSPDI.h, 126
APC_ResizeImgToHalf	APC_Setup_v4l2_requestbuffers
eSPDI.h, 108	eSPDI.h, 123
APC_RetrieveExtraDataFromMp4	APC_SetupBlock
eSPDI.h, 109	eSPDI.h, 124
APC_RotateImg180	APC_SetupHidGyro
eSPDI.h, 110	eSPDI.h, 124
APC_RotateImg90	APC_SubSample
eSPDI.h, 111, 112	eSPDI.h, 126
APC_SaveLutData	APC_SwitchBaseline
eSPDI.h, 112	eSPDI.h, 127
APC_SelectDevice	APC_TableToData
eSPDI.h, 113	eSPDI.h, 127
APC_SetAutoExposureMode	APC_TemporalFilter
eSPDI.h, 113	eSPDI.h, 128
APC_SetCTPropVal	APC_WriteCmdFiFo
eSPDI.h, 115	eSPDI.h, 129
APC SetColorGain	APC_WriteFlashData
eSPDI.h, 113	eSPDI.h, 129
APC_SetControlCounterMode	APC_WriteWaveEnd
eSPDI.h, 115	eSPDI.h, 130
APC_SetCurrentIRValue	APC_WriteWaveHeader
eSPDI.h, 116	eSPDI.h, 130
APC_SetDepthDataType	APC_getUACNAME
eSPDI.h, 116	eSPDI.h, 92
APC_SetExposureTime	APCImageType, 31
	AccelerationTag, 31
eSPDI.h, 116	•
APC_SetFWRegister	CamDist1
eSPDI.h, 117	eSPCtrl_RectLogData, 33
APC_SetGlobalGain	CamDist2
eSPDI.h, 117	eSPCtrl_RectLogData, 33
APC_SetHWRegister	CamMat1
eSPDI.h, 118	eSPCtrl_RectLogData, 33
APC_SetIRMaxValue	CamMat2
eSPDI.h, 118	eSPCtrl_RectLogData, 33
APC_SetIRMode	CompassTag, 32
eSPDI.h, 119	
APC_SetLogData	eSPCtrl_RectLogData, 32
eSPDI.h, 119	CamDist1, 33
APC_SetMultiBytesHWRegister	CamDist2, 33
eSPDI.h, 120	CamMat1, 33
APC_SetPUPropVal	CamMat2, 33
eSPDI.h, 121	InImgHeight, 33
APC_SetPidVid	InImgWidth, 33
eSPDI.h, 120	LRotaMat, 33
APC_SetRectifyTable	nLineBuffers, 34
eSPDI.h, 121	NewCamMat1, 33
APC_SetSensorRegister	NewCamMat2, 34
eSPDI.h, 122	OutImgHeight, 34

OutlingWidth, 34         APC_GetColorGain, 69           RECT_Crop_Col_BG_L, 34         APC_GetColorMage, 70           RECT_Crop_Row_BG, 35         APC_GetColorImageWithTimestamp, 70           RECT_Crop_Row_ED, 35         APC_GetControlCounterMode, 71           RECT_Crop_Enable, 35         APC_GetDevicentIvalue, 73           RECT_Scale_Col_M, 35         APC_GetDevicently 76           RECT_Scale_Col_M, 35         APC_GetDevicelnito, 76           RECT_Scale Row_M, 35         APC_GetDevicelnito, 76           RECT_Scale Row_M, 35         APC_GetDevicelnito, 76           RECT_Scale Flowle, 35         APC_GetDevicelnito, 76           RECT_Scale Flowle, 35         APC_GetDevicelnito, 76           RECT_Scale Width, 36         APC_GetDevicelnito, 76           RECT_Scale Width, 36         APC_GetScalewidth, 36           RECT_Scale Width, 36         APC_GetEriscibleGyroLength, 78           REOT_Scale Width, 36         APC_GetScalewidth, 36           RECT_Scalewidth, 36         APC_GetScalewidth, 36 <th></th> <th></th>		
RECT_Crop_Col_EG_L_, 34         APC_GetControlCounterMode, 71           RECT_Crop_Col_ED_L, 34         APC_GetControlCounterMode, 71           RECT_Crop_Row_EG, 35         APC_GetControlCounterMode, 71           RECT_Crop_Row_ED, 35         APC_GetDepthData_Type, 73           RECT_Crop_Enable, 35         APC_GetDepthData_Type, 73           RECT_Scale_Col_M, 35         APC_GetDeviceInto, 75           RECT_Scale Row_M, 35         APC_GetDeviceInto, 75           RECT_Scale Row N, 35         APC_GetDeviceInto, 75           RECT_Scale Helpht, 36         APC_GetDeviceInto, 75           RECT_Scale Helpht, 36         APC_GetDeviceInto, 75           RECT_Scale Helpht, 36         APC_GetEvricePolitofulls, 76           RECT_Scale Helpht, 36         APC_GetEvriceReputionList, 76           RECT_Scale Helpht, 36         APC_GetFWRegister, 78           RECT_Scale Helpht, 36         APC_GetBetWrestron, 79           RECT_Scale Helpht, 36         APC_GetBetWrestron, 79           APC_CoseDeviceResolution_BL         APC_GetBetWrestron, 79           APC_GetBetWrestron, 51	OutImgWidth, 34	APC_GetColorGain, 69
RECT. Crop. Row. BG, 35 RECT. Crop. Row. ED, 35 RECT. Crop. Row. ED, 35 RECT. Crop. Row. ED, 35 RECT. Scale. Col. M, 35 RECT. Scale. Col. M, 35 RECT. Scale. Col. N, 35 RECT. Scale. Row. M, 35 RECT. Scale Row. N, 36 RECT. Scal	RECT_AvgErr, 34	APC_GetColorImage, 70
RECT. Crop. Row. BG, 35 RECT. Crop. Row. ED, 35 RECT. Crop. Row. ED, 35 RECT. Crop. Row. ED, 35 RECT. Scale. Col. M, 35 RECT. Scale. Col. M, 35 RECT. Scale. Col. N, 35 RECT. Scale. Row. M, 35 RECT. Scale Row. N, 36 RECT. Scal	RECT Crop Col BG L, 34	APC GetColorImageWithTimestamp, 70
RECT_Crop_Row_ED, 35 RECT_Crop_Row_ED, 35 RECT_Crop_Row_ED, 35 RECT_Crop_Row_ED, 35 RECT_Scale_Col_M, 35 RECT_Scale_Col_M, 35 RECT_Scale_Col_M, 35 RECT_Scale_Col_M, 35 RECT_Scale_Row_M, 35 RECT_Scale_Row_N, 35 RECT_Scale_Row_N, 35 RECT_Scale_Row_N, 35 RECT_Scale_Row_N, 35 RECT_Scale_Row_N, 36 RECT_ScaleWidth, 36 ROTAMAL, 36 ROCAMAL, 36 ROTAMAL, 36 ROTAMAL, 36 ROTAMAL, 36 ROTAMAL, 36 ROTAMAL, 36 ROTAMAL, 36 ROTA		
RECT_Crop_Row_ED, 35 RECT_CropEnable, 35 RECT_Scale, Col_M, 35 RECT_Scale Col_M, 35 RECT Scale Col_M, 35 RECT Scale Col_M, 35 RECT_Scale Row_M, 35 RECT_Scale Row_M, 35 RECT_Scale Row_N, 35 RPC_Gcale Row_N, 36 RPC_Gcale Row_N, 35 RPC_Gcale Row_N, 36 RPC_Gcale Row_N,		_ ,
RECT_CropEnable, 35 RECT_Scale Col M, 35 RECT_Scale Col M, 35 RECT_Scale, Row M, 35 RECT_Scale, Row M, 35 RECT_Scale, Row M, 35 RECT_Scale, Row M, 35 RECT_Scale Row M, 35 RPC_GelFwelpiden(Rom F, 76 RPC_GelFwelpiden(Rom F, 76 RPC_GelFwelpide(Row Row Row Row Row Row Row Row Row Row		
RECT_Scale_CoL_M, 35         APC_GetDeviceInfo, 75           RECT_Scale_Row_M, 35         APC_GetDeviceInfo, 75           RECT_Scale_Row_M, 35         APC_GetDeviceInfoMBL_15cm, 75           RECT_Scale_Row_N, 35         APC_GetDeviceReNoWBL_15cm, 75           RECT_ScaleHeight, 36         APC_GetDeviceResolutionList, 76           RECT_ScaleHeight, 36         APC_GetTevolumber, 76           RECT_ScaleWidth, 36         APC_GetTevolumber, 38           ReCT_ScaleWidth, 36         APC_GetTevolumber, 39           APC_CloseOvice, 31         APC_GetTevolumber, 30           APC_GetTevolumber, 50         APC_GetTevolumber, 30		
RECT_Scale_Co_N, 35         APC_GetDeviceInfo, 75           RECT_Scale_Row_N, 35         APC_GetDeviceInfoMBL_15cm, 75           RECT_ScaleFable, 35         APC_GetDeviceNumber, 76           RECT_ScaleHeight, 36         APC_GetDeviceResolutionList, 76           RECT_ScaleWidth, 36         APC_GetExposureTime, 77           RECT_ScaleWidth, 36         APC_GetFWRegister, 78           RotaMat, 36         APC_GetFlexibleGyroData, 77           RotaMat, 36         APC_GetFlexibleGyroLength, 78           TranMat, 36         APC_GetFlexibleGyroLength, 78           BrotaMat, 36         APC_GetFlexibleGyroLength, 78           TranMat, 36         APC_GetFlexibleGyroLength, 78           BrotaMat, 36         APC_GetFlexibleGyroLength, 78           APC_GetPowcell County, 79         APC_GetFlexibleGyroLength, 78           APC_GetPlexibleGyroLength, 78         APC_GetFlexibleGyroLength, 78           APC_GetPlexibleGyroLength, 78         APC_GetHlmWRegister, 80           APC_CloseDeviceEx, 52         APC_GetHlmWRegister, 80           APC_CloseDeviceEx, 52         APC_GetIlmMode, 84           APC_CloseDeviceEx, 52         APC_GetIlmMode, 84           APC_CoreateSwPostProc, 54         APC_GetIlmage, 81           APC_DeptilmErg, 55         APC_GetIlmage, 81           APC_DeptilmErg, 55         APC_GetIlmage, 82 <td></td> <td>_ · · · · ·</td>		_ · · · · ·
RECT_Scale_Row_N, 35         APC_GetDeviceInfoMBL_15cm, 75           RECT_ScaleEnable, 35         APC_GetDeviceNumber, 76           RECT_ScaleHeight, 36         APC_GetDeviceNumber, 76           RECT_ScaleHeight, 36         APC_GetExposureTime, 77           RECT_ScaleWidth, 36         APC_GetFWRegister, 78           RotaMat, 36         APC_GetFiexibleGyroData, 77           RotaMat, 36         APC_GetFiexibleGyroLength, 78           uByteArray, 36         APC_GetFiexibleGyroLangth, 78           esPDLh         APC_GetFiexibleGyroLangth, 78           APC_ApplyFilters, 50         APC_GetFiexibleGyroLangth, 78           APC_GetPwersion, 79         APC_GetFiexibleGyroLangth, 78           APC_ColoseCovice.5         APC_GetFiexibleGyroLangth, 78           APC_GetPwersion, 79         APC_GetFiexibleGyroLangth, 78           APC_GetPwersion, 79         APC_GetFiexibleGyroLangth, 78           APC_GetPollosalGain, 79         APC_GetPiexibleGyroLangth, 78           APC_GetPollosalGain, 79         APC_GetPiexibleGyroLangth, 78           APC_GetPollosalGain, 79         APC_GetPiexibleGyroLangth, 78           APC_GetDeviceEx.52         APC_GetHidGyro, 80           APC_GetBetDeviceInfollosal         APC_GetIllosalGain, 79           APC_GetBetDeviceInfollosal         APC_GetIllosalCall           APC_Covert_Depth_Y To_Buf	:	
RECT_Scale Row, N, 35         APC_GetDeviceNumber, 76           RECT_ScaleBriable, 35         APC_GetDeviceResolutionList, 76           RECT_ScaleWight, 36         APC_GetExposureTime, 77           RECT_ScaleWighth, 36         APC_GetFlexibleGyroLength, 78           RRotaMat, 36         APC_GetFlexibleGyroData, 77           RotaMat, 36         APC_GetFlexibleGyroLength, 78           TranMat, 36         APC_GetFlexibleGyroLength, 78           UByteArray, 36         APC_GetFlexibleGyroLength, 78           eSPDI.h         APC_GetIGloalGalin, 79           APC_CloseComdFiFo, 51         APC_GetHWRegister, 80           APC_CloseComdFiFo, 51         APC_GetHWRegister, 80           APC_CloseDevice, 51         APC_GetIRMavaluue, 83           APC_CloseDeviceEx, 52         APC_GetIRMode, 84           APC_CloseDeviceEx, 52         APC_GetIRMode, 84           APC_ColorFormat_to_RGB24, 52         APC_GetIRMode, 84           APC_Corent_Depth_Y_To_Buffer, 53         APC_GetIRMode, 84           APC_Corent_Depth_Y_To_Buffer, 53         APC_GetIndFidGyro, 81           APC_Corent_Depth_Y_To_Buffer, 53         APC_GetILogData, 84           APC_DeryptString, 55         APC_GetILogData, 84           APC_DeryptString, 55         APC_GetIlogData, 84           APC_DosableAWB, 57         APC_GetPuffer, 61		
RECT_ScaleHable, 35         APC_GetEvoiceResolutionList, 76           RECT_ScaleHight, 36         APC_GetExposureTime, 77           RECT_ScaleWidth, 36         APC_GetExposureTime, 77           RRotaMat, 36         APC_GetFlexibleGyroLata, 77           RotaMat, 36         APC_GetFlexibleGyroLength, 78           BranMat, 36         APC_GetFlexibleGyroLength, 78           ByteArray, 36         APC_GetFloxibleGyroLength, 78           ByteArray, 36         APC_GetFloxibleGyroLength, 78           BYDLh         APC_GetIMate, 30           APC_ApplyFilters, 50         APC_GetHWRegister, 80           APC_CoseDevice, 51         APC_GetHidGyro, 80           APC_CoseDevice, 51         APC_GetIMate, 33           APC_CloseDevice, 51         APC_GetIMate, 81           APC_GetIMED, 52         APC_GetIMate, 83           APC_CloseDevice, 52         APC_GetIMate, 84           APC_GetIMED, 72         APC_GetIMED, 81           APC_GetIMED, 72         APC_GetIMED, 81           APC_Dest		
RECT_ScaleHeight, 36         APC_GetExposureTime, 77           RECT_ScaleWidth, 36         APC_GetIWRegister, 78           RRotaMat, 36         APC_GetIexibleGyroData, 77           RotaMat, 36         APC_GetIexibleGyroData, 77           RotaMat, 36         APC_GetIexibleGyroLength, 78           IranMat, 36         APC_GetIexibleGyroLength, 78           UByteArray, 36         APC_GetIexibleGyroLength, 78           eSPDI.h         APC_GetIexibleGyro. 30           APC_CloseDevice, 51         APC_GetHidGyro. 30           APC_CloseDevice, 51         APC_GetIRMode, 84           APC_CloseDeviceEx, 52         APC_GetIRMode, 84           APC_CloseDeviceMBL, 52         APC_GetImBidode, 84           APC_Convert_Depth_Y To_Buffer, 53         APC_GetImage, 81           APC_Convert_Depth_Y To_Buffer, 53         APC_GetImage, 81           APC_CreateSWPostProc, 54         APC_GetLogData, 84           APC_DecryptIMP4, 55         APC_GetIvData, 85           APC_DeptIMerge, 56         APC_GetIvData, 85           APC_DisableAWB, 57         APC_GetPUPropVal, 87           APC_DoswPostProc, 58         APC_GetPidVid, 86           APC_GetPreServingFilter, 59         APC_GetPectifyAugua, 89           APC_EnableAWB, 60         APC_GetPectifyAugua, 89           APC_EnableAWB, 60 <t< td=""><td></td><td></td></t<>		
RECT_ScaleWidth, 36		
RRotaMat, 36	_ <del>-</del>	_ ·
RotaMat, 36		
TranMat, 36         APC_GetFwVersion, 79           eSPDI.h         APC_GetGlobalGain, 79           eSPDI.h         APC_GetHWRegister, 80           APC_LoseDevice, 51         APC_GetHWRegister, 80           APC_CloseDevice, 51         APC_GetIRMaxValue, 83           APC_CloseDevice, 51         APC_GetIRModulae, 83           APC_CloseDevice, 52         APC_GetIRModulae, 83           APC_CloseDevice, 52         APC_GetIRMode, 84           APC_ColorFormat_Depth_Y_To_Buffer, 53         APC_GetImdole, 84           APC_Convert_Depth_Y_To_Buffer, 53         APC_GetImloHidGyro, 81           APC_Convert_Depth_Y_To_Buffer_offset, 54         APC_GetLuOpdata, 84           APC_CoreateSwPostProc, 54         APC_GetLutData, 85           APC_DecryptMP4, 55         APC_GetLutData, 85           APC_DeryptMP4, 55         APC_GetPUPropVal, 87           APC_DesybtMerge, 56         APC_GetPUPropVal, 87           APC_DisableAWB, 57         APC_GetPUPropVal, 87           APC_DisableAWB, 57         APC_GetPloHidd, 86           APC_EnableAWB, 60         APC_GetRectifyAgoData, 88           APC_EnableAWB, 60         APC_GetRectifyAgoData, 89           APC_EnableSensorif, 61         APC_GetSerialNumber, 90           APC_EnableSensorif, 61         APC_GetSerialNumber, 90           APC_EntryDytStrin		_ ·
uByteArray, 36 eSPDI.h  APC_ApplyFilters, 50  APC_CloseCmdFiFo, 51  APC_CloseCmdFiFo, 51  APC_CloseDevice, 51  APC_CloseDevice, 51  APC_CloseDeviceMBL, 52  APC_CloseDeviceMBL, 52  APC_ColoreTormat_to_RGB24, 52  APC_Convert_Depth_Y_To_Buffer, 53  APC_Cornert_Depth_Y_To_Buffer, 53  APC_Cornert_Depth_Y_To_Buffer, 53  APC_DeryptMP4, 55  APC_DeryptMP4, 55  APC_DeryptMP4, 55  APC_DeryptMP4, 55  APC_DepthMerge, 56  APC_DisableAWB, 57  APC_DisableAWB, 57  APC_DisableAWB, 57  APC_DoFusion, 57  APC_DoFusion, 57  APC_DoFusion, 57  APC_EnableAWB, 60  APC_EnableAWB, 60  APC_EnableAE, 59  APC_EnableAE, 59  APC_EnableAE, 59  APC_EnableBeriac in, 60  APC_EnableBeriac in, 60  APC_EnableBeriac in, 60  APC_EnableBeriac in, 61  APC_EnableBeriac in, 62  APC_EnableBeriac in, 62  APC_EncryptString, 62  APC_EncryptString, 62  APC_GertDeriac in, 63  APC_GertDeriac in, 64  APC_GertDeriac in, 65  APC_GertDerial in, 64  APC_GertDeriac in, 65  APC_GertDeriac in, 66  APC_GertDeriac in, 66  APC_GertSerial in, 66  APC_GertSerial in, 66  APC_GertSerial in, 69  APC_GertAwBStatus, 68  APC_GertBeutiffle, 97  APC_GertAwBStatus, 68  APC_GertAwBStatus, 68  APC_GertBustinfo, 69  APC_GertSerial in, 60  APC_GertAwBStatus, 68  APC_GertBustinfo, 69  APC_GertCorpval, 71  APC_GertBaseLine, 100		
eSPDI.n  APC_ApplyFilters, 50  APC_CloseCmdFiFo, 51  APC_CloseDevice, 51  APC_CloseDevice, 51  APC_CloseDevice, 51  APC_CloseDevice, 52  APC_CloseDeviceMBL, 52  APC_ColorEdviceMBL, 52  APC_Convert_Depth_Y_To_Buffer, 53  APC_Cornvert_Depth_Y_To_Buffer, 53  APC_Cornvert_Depth_Y_To_Buffer, 54  APC_DecryptIMP4, 55  APC_DecryptString, 55  APC_DecryptString, 55  APC_DecryptString, 55  APC_DisableAWB, 57  APC_DisableAWB, 57  APC_DoSwPostProc, 58  APC_DoSwPostProc, 58  APC_EdgePreServingFilter, 59  APC_EdgePreServingFilter, 59  APC_EnableAE, 59  APC_EnableAE, 59  APC_EnableGPUAcceleration, 60  APC_EnableGPSensoriF, 61  APC_EnableAE, 59  APC_EnableAE, 59  APC_EnableAE, 59  APC_EnableAE, 59  APC_EnableAC, 61  APC_EnableBernam_depth, 65  APC_Gett_Sing_02  APC_Gett_Sing_91  APC_GettData, 92  APC_GettSerialNumber, 90  APC_GetelSerialNumber, 90  APC_GetelSerialNumber, 90  APC_GetelDepthCancellation_D11, 63  APC_Gett_Sing_02  APC_Gett_Sing_02  APC_Gett_Sing_02  APC_Gett_Sing_03  APC_Gett_Sing_04  APC_Gett_Sing_04  APC_Gett_Sing_05  APC_Gett_Sing_05  APC_Gett_Sing_05  APC_Gett_Sing_05  APC_Gett_Sing_05  APC_Gett_Sing_05  APC_Gett_Sing_06  APC_InitialHidGyro, 97  APC_GetAcMeterValue, 67  APC_GetBuslanfo, 69  APC_GetCiTPropVal, 71  APC_GetIMRavValue, 83  APC_GetIMMavValue, 83  APC_GetIMMavValue, 83  APC_GetIMMavValue, 83  APC_GetIMMode, 84  APC_GetIMMode, 84  APC_GetIMIndivalue, 83  APC_GetIMMode, 84  APC_GetIMMode, 81  APC_GetIMmode, 81  APC_GetIMindivalue, 83  APC_GetIMedepth, 85  APC_GetIVefixet, 83  APC_GetIVefixet, 85  APC_GetIVefixet, 85  APC_InitialFlackyro, 97  APC_GetAcMeterValue, 67  APC_GetAcMeterValue, 67  APC_GetBuslanfo, 69  APC_GetCiTPropVal, 71  APC_GetIMedepthere 83  APC_GetIMedepthere 83  APC_GetIVerset 83  APC_G		
APC_ApplyFilters, 50 APC_CloseCmdFiFo, 51 APC_CloseDevice, 51 APC_CloseDevice, 51 APC_CloseDevice, 51 APC_CloseDevice, 52 APC_CloseDeviceMBL, 52 APC_CloseDeviceMBL, 52 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer_offset, 54 APC_Corvert_Depth_Y_To_Buffer_offset, 54 APC_Corvert_Depth_Y_To_Buffer_offset, 54 APC_Corvert_Depth_Y_To_Buffer_offset, 54 APC_DecryptMP4, 55 APC_DecryptMP4, 55 APC_DecryptMP4, 55 APC_DepthMerge, 56 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DoFusion, 57 APC_DoFusion, 57 APC_DoFusion, 57 APC_DepthMerge, 56 APC_EdgePreServingFilter, 59 APC_EdgePreServingFilter, 59 APC_EnableAWB, 60 APC_EnableAWB, 60 APC_EnableAE, 59 APC_EnableAE, 59 APC_EnableBensorIF, 61 APC_EnableSensorIF, 61 APC_EnableSensorIF, 61 APC_EncryptMP4, 62 APC_EncryptMP4, 62 APC_EncryptString, 62 APC_EncryptString, 62 APC_EncryptString, 62 APC_GetLondelation_D11, 63 APC_FlyingDepthCancellation_D8, 64 APC_GetLondent, 65 APC_GetLondent, 65 APC_GetLondent, 66 APC_GetLondent, 66 APC_GetLondent, 66 APC_GetLondent, 66 APC_GetLondent, 67 APC_GetLondent, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetCorrPropVal, 71 APC_GetIBaseLine, 100	uByteArray, 36	<del>-</del>
APC_CloseDevice, 51 APC_CloseDevice, 51 APC_CloseDevice, 51 APC_CloseDeviceMBL, 52 APC_CloseDeviceMBL, 52 APC_CloseDeviceMBL, 52 APC_CloseDeviceMBL, 52 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer, 54 APC_CreateSwPostProc, 54 APC_DecryptMP4, 55 APC_DecryptMP4, 55 APC_DecryptString, 55 APC_DecryptString, 55 APC_DespthMerge, 56 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DoSwPostProc, 58 APC_EdgePreServingFilter, 59 APC_EnableAWB, 60 APC_EntWBStatus, 66 APC_GetZUTable, 93 APC_InitialHidGyro, 97 APC_InitialHidGy	eSPDI.h	APC_GetHWRegister, 80
APC_CloseDeviceEx, 52 APC_CloseDeviceEx, 52 APC_CloseDeviceMBL, 52 APC_CloseDeviceMBL, 52 APC_ColorFormat_to_RGB24, 52 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer, 54 APC_CreateSwPostProc, 54 APC_DecryptMP4, 55 APC_DecryptMP4, 55 APC_DecryptString, 55 APC_DecryptString, 55 APC_DespthMerge, 56 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DoSwPostProc, 58 APC_EdgePreServingFilter, 59 APC_EnableAWB, 60 APC_EnableAWB, 60 APC_EnableAE, 59 APC_EnableAE, 59 APC_EnableGPUAcceleration, 60 APC_EnableGPUAcceleration, 60 APC_EnableGPUAcceleration, 60 APC_EnableGPUAcceleration, 60 APC_EncryptString, 62 APC_FlyingDepthCancellation_D11, 63 APC_GetPindDevice, 63 APC_GetPindDevice, 63 APC_GetPindDevice, 63 APC_GetPindDevice, 63 APC_GetPindDevice, 65 APC_Get_150_mm_depth, 65 APC_Get_Color_30_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_GetAEStatus, 67 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetCTPropVal, 71 APC_InitialFlackDelevice, 99 APC_IsMLBaseLine, 100	APC_ApplyFilters, 50	APC_GetHidGyro, 80
APC_CloseDeviceMBL, 52 APC_CloseDeviceMBL, 52 APC_CloseDeviceMBL, 52 APC_CloseDeviceMBL, 52 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer, 54 APC_CreateSwPostProc, 54 APC_DecryptMP4, 55 APC_DecryptMr9, 55 APC_DepthMerge, 56 APC_DisableABL, 57 APC_DisableABL, 57 APC_DisableABL, 57 APC_DoFusion, 57 APC_DoSwPostProc, 58 APC_EnableABL, 69 APC_EnableABL, 69 APC_EnableABL, 69 APC_EnableBePUAcceleration, 60 APC_EnableBePUAcceleration, 60 APC_EnableBePUAcceleration, 60 APC_EnableBePUAcceleration, 60 APC_EncryptString, 62 APC_FlyingDepthCancellation_D11, 63 APC_GetPUBC, 63 APC_GetPUBC, 63 APC_GetPUBC, 66 APC_Get_150_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_Get_ACMEDLE, 67 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetCTPropVal, 71 APC_GetSUseData, 91 APC_InitialFlexibleGyro, 97 APC_InitialFlexibleGyro, 97 APC_InitialFlexibleGyro, 99 APC_IsMLBaseLine, 100	APC_CloseCmdFiFo, 51	APC_GetIRMaxValue, 83
APC_CloseDeviceMBL, 52 APC_ColorFormat_to_RGB24, 52 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer_offset, 54 APC_CreateSwPostProc, 54 APC_CreateSwPostProc, 54 APC_DecryptMP4, 55 APC_DecryptMP4, 55 APC_DecryptMP4, 55 APC_DepthMerge, 56 APC_DepthMerge, 56 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DoFusion, 57 APC_DoFusion, 57 APC_DoSwPostProc, 58 APC_EdgePreServingFilter, 59 APC_EnableAWB, 60 APC_EnableAWB, 60 APC_EnableAWB, 60 APC_EnableBerouse, 61 APC_EnableBerouse, 61 APC_EnableBerouse, 61 APC_EnableBerouse, 61 APC_EnableBerouse, 61 APC_EncryptString, 62 APC_EncryptString, 62 APC_EncryptString, 62 APC_EncryptString, 62 APC_GetPlWanda, 92 APC_FlyingDepthCancellation_D11, 63 APC_GetPlose, 63 APC_InitialCeptplose, 93 APC_GetAEStatus, 67 APC_GetAEStatus, 68 APC_InitialFlexibleGyro, 97 APC_GetAUCDExposureMode, 68 APC_InitialFlexibleGyro, 97 APC_GetAUCDExposureMode, 68 APC_GetCTPropVal, 71 APC_InMBaseLine, 100	APC_CloseDevice, 51	APC_GetIRMinValue, 83
APC_ColorFormat_to_RGB24, 52 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer, 53 APC_CreateSwPostProc, 54 APC_CreateSwPostProc, 54 APC_DecryptMP4, 55 APC_DecryptMP4, 55 APC_DecryptString, 55 APC_DepthMerge, 56 APC_DepthMerge, 56 APC_DisableAKB, 57 APC_DisableAKB, 57 APC_DisableAKB, 57 APC_DoFusion, 57 APC_DoFusion, 57 APC_DoFusion, 57 APC_DespthMerge, 56 APC_GetPUPropVal, 87 APC_DepthMerge, 56 APC_DespthMerge, 56 APC_DespthMerge, 56 APC_DespthMerge, 56 APC_DisableAKB, 57 APC_DisableAKB, 57 APC_DisableAKB, 57 APC_DoFusion, 57 APC_DespthPort, 58 APC_GetPointCloud, 86 APC_GetRectifyLogData, 88 APC_GetRectifyLogData, 89 APC_GetRectifyTable, 89 APC_GetRectifyTable, 89 APC_GetRectifyTable, 89 APC_GetRectifyTable, 89 APC_GetSensorRegister, 90 APC_EnableAWB, 60 APC_EnableAKB, 59 APC_GetSensorRegister, 90 APC_EnableAKB, 59 APC_EnableAKB, 59 APC_GetSensorRegister, 90 APC_GetSensorIF, 61 APC_GetSensorIF, 61 APC_GetSensorIF, 61 APC_GetUACData, 92 APC_GetUACData, 92 APC_GetUACData, 92 APC_GetUSerData, 92 APC_GetUserData, 92 APC_GetUserData, 92 APC_GetVoffset, 93 APC_GetSensorIF, 64 APC_GetPointCancellation_D11, 63 APC_FlyingDepthCancellation_D11, 63 APC_FlyingDepthCancellation_D11, 63 APC_HoleFille, 95 APC_GetZolor_30_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_InitSRB, 99 APC_InitIalICmdFiFo, 97 APC_InitialIIdGyro, 97 APC_GetAUSEXpoureMode, 68 APC_GetAUSEXpoureMode, 68 APC_GetAUSEXpoureMode, 68 APC_GetBusInfo, 69 APC_GetCOrropVal, 71 APC_IntItalPlaseLine, 100	APC_CloseDeviceEx, 52	APC_GetIRMode, 84
APC_ColorFormat_to_RGB24, 52 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer, 53 APC_CreateSwPostProc, 54 APC_CreateSwPostProc, 54 APC_DecryptMP4, 55 APC_DecryptMP4, 55 APC_DecryptString, 55 APC_DepthMerge, 56 APC_DepthMerge, 56 APC_DisableAKB, 57 APC_DisableAKB, 57 APC_DisableAKB, 57 APC_DoFusion, 57 APC_DoFusion, 57 APC_DoFusion, 57 APC_DespthMerge, 56 APC_GetPUPropVal, 87 APC_DepthMerge, 56 APC_DespthMerge, 56 APC_DespthMerge, 56 APC_DespthMerge, 56 APC_DisableAKB, 57 APC_DisableAKB, 57 APC_DisableAKB, 57 APC_DoFusion, 57 APC_DespthPort, 58 APC_GetPointCloud, 86 APC_GetRectifyLogData, 88 APC_GetRectifyLogData, 89 APC_GetRectifyTable, 89 APC_GetRectifyTable, 89 APC_GetRectifyTable, 89 APC_GetRectifyTable, 89 APC_GetSensorRegister, 90 APC_EnableAWB, 60 APC_EnableAKB, 59 APC_GetSensorRegister, 90 APC_EnableAKB, 59 APC_EnableAKB, 59 APC_GetSensorRegister, 90 APC_GetSensorIF, 61 APC_GetSensorIF, 61 APC_GetSensorIF, 61 APC_GetUACData, 92 APC_GetUACData, 92 APC_GetUACData, 92 APC_GetUSerData, 92 APC_GetUserData, 92 APC_GetUserData, 92 APC_GetVoffset, 93 APC_GetSensorIF, 64 APC_GetPointCancellation_D11, 63 APC_FlyingDepthCancellation_D11, 63 APC_FlyingDepthCancellation_D11, 63 APC_HoleFille, 95 APC_GetZolor_30_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_InitSRB, 99 APC_InitIalICmdFiFo, 97 APC_InitialIIdGyro, 97 APC_GetAUSEXpoureMode, 68 APC_GetAUSEXpoureMode, 68 APC_GetAUSEXpoureMode, 68 APC_GetBusInfo, 69 APC_GetCOrropVal, 71 APC_IntItalPlaseLine, 100	APC CloseDeviceMBL, 52	APC GetImage, 81
APC_Convert_Depth_Y_To_Buffer, 53 APC_Convert_Depth_Y_To_Buffer_offset, 54 APC_Convert_Depth_Y_To_Buffer_offset, 54 APC_CreateSwPostProc, 54 APC_GetLudData, 85 APC_DecryptMP4, 55 APC_DecryptMP4, 55 APC_DepthMerge, 56 APC_DepthMerge, 56 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DisableAE, 57 APC_DoFusion, 57 APC_DoSwPostProc, 58 APC_GetPUloud, 86 APC_GetPointCloud, 86 APC_GetSensorRegister, 90 APC_GetSensorRegister, 90 APC_GetSensorRegister, 90 APC_GetSensorRegister, 90 APC_GetSensorRegister, 90 APC_GetSensorRegister, 90 APC_GetSerialNumber, 90 APC_GetDerpointialNumber, 90 APC_GetDerpointialNumber, 90 APC_GetDerpointialNumber, 90 APC_GetDerpointialNumber, 90 APC_GetDerpointialNumber, 90 APC_GetAutSetatus, 67 APC_GetAutSetatus, 67 APC_GetAutSetatus, 68 APC_InitialPidGyro, 97 APC_GetAutSetatus, 68 APC_InitialPidGyro, 97 APC_InitialPidGyro, 97 APC_InitialPidGyro, 97 APC	— · · · · · · · · · · · · · · · · · · ·	
APC_Convert_Depth_Y_To_Buffer_offset, 54 APC_CreateSwPostProc, 54 APC_DecryptMP4, 55 APC_DecryptString, 55 APC_DepthMerge, 56 APC_DepthMerge, 56 APC_DisableAWB, 57 APC_DisableAE, 57 APC_DisableAE, 57 APC_DoswPostProc, 58 APC_GetPURangeAndStep, 87 APC_DoswPostProc, 58 APC_GetPeroval, 88 APC_GetPerintCloud, 86 APC_GetPerintCloud, 86 APC_GetPerintCloud, 86 APC_GetPerintCloud, 86 APC_GetPerintCloud, 86 APC_GetRectifyLogData, 89 APC_GetRectifyLogData, 89 APC_GetRectifyTable, 89 APC_GetRectifyTable, 89 APC_GetSensorRegister, 90 APC_EnableAE, 59 APC_EnableAE, 59 APC_EnableGPUAcceleration, 60 APC_EnableBensorIF, 61 APC_GetSensorIF, 61 APC_GetThermalFD, 91 APC_GetThermalFD, 91 APC_GetUACData, 92 APC_EncryptMP4, 62 APC_EncryptString, 62 APC_FindDevice, 63 APC_FlyingDepthCancellation_D11, 63 APC_FlyingDepthCancellation_D8, 64 APC_Get2DTable, 93 APC_HoleFill, 94 APC_Get2DTable, 93 APC_HoleFill, 94 APC_Get2DTable, 95 APC_Get_Color_30_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_GetAEStatus, 67 APC_GetAUCEACMeterValue, 67 APC_GetBusInfo, 69 APC_InitialIHidGyro, 97 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_InitialIPaxibataToMp4, 99 APC_GetBusInfo, 69 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100		_ • •
APC_CerateSwPostProc, 54 APC_DecryptMP4, 55 APC_DecryptString, 55 APC_DecryptString, 55 APC_DepthMerge, 56 APC_DepthMerge, 56 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DoswPostProc, 58 APC_GetPidVid, 86 APC_GetPectifyLogData, 88 APC_GetBectifyLogData, 89 APC_EdgePreServingFilter, 59 APC_GetBectifyTable, 89 APC_GetBectifyTable, 89 APC_EnableAWB, 60 APC_EnableAWB, 60 APC_EnableGPUAcceleration, 60 APC_EnableGPUAcceleration, 60 APC_EnableBensorIF, 61 APC_GetBectifyTable, 91 APC_EncryptMP4, 62 APC_EncryptMP4, 62 APC_EncryptMP4, 62 APC_EncryptString, 62 APC_FindDevice, 63 APC_FlyingDepthCancellation_D11, 63 APC_FlyingDepthCancellation_D8, 64 APC_GetBectifyLogData, 89 APC_GetBectifyLogData, 89 APC_HoleFill, 94 APC_GetUAcoData, 92 APC_EncryptString, 62 APC_FlyingDepthCancellation_D11, 63 APC_FlyingDepthCancellation_D8, 64 APC_GetBectBectBectBectBectBectBectBectBectBe	·	<del>-</del>
APC_DecryptMP4, 55 APC_DecryptString, 55 APC_DecryptString, 55 APC_DepthMerge, 56 APC_DepthMerge, 56 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DisableAE, 57 APC_DisableAE, 57 APC_DoFusion, 57 APC_DoFusion, 57 APC_DoFusion, 57 APC_DoSwPostProc, 58 APC_GetRectifyLogData, 88 APC_EdgePreServingFilter, 59 APC_EnableAWB, 60 APC_EnableAE, 59 APC_EnableAE, 59 APC_EnableGPUAcceleration, 60 APC_EnableGPUAcceleration, 60 APC_EnableSensorIF, 61 APC_EncryptMP4, 62 APC_EncryptMP4, 62 APC_EncryptString, 62 APC_FindDevice, 63 APC_FlyingDepthCancellation_D11, 63 APC_FlyingDepthCancellation_D8, 64 APC_Get_150_mm_depth, 65 APC_Get_150_mm_depth, 65 APC_Get_150_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_Get_ACCMeterValue, 67 APC_GetAuoexposureMode, 68 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetCIPropVal, 71 APC_IsMLBaseLine, 100	·	
APC_DecryptString, 55 APC_DepthMerge, 56 APC_DisableAWB, 57 APC_DisableAWB, 57 APC_DoFusion, 57 APC_DoFusion, 57 APC_DoSwPostProc, 58 APC_EdgePreServingFilter, 59 APC_EnableAWB, 60 APC_EnableAWB, 60 APC_EnableGPUAcceleration, 60 APC_EnableGPUAcceleration, 60 APC_EnableSensorIF, 61 APC_EncryptMP4, 62 APC_EncryptString, 62 APC_EncryptString, 62 APC_EncryptString, 62 APC_ElindDevice, 63 APC_ElindDevice, 63 APC_ElindDevice, 63 APC_GetPindDevice, 65 APC_GetPindDevice, 65 APC_GetPindDevice, 65 APC_GetPindDevice, 65 APC_GetIllage, 65 APC_GetIllage, 65 APC_GetIllage, 65 APC_GetIllage, 65 APC_GetIllage, 65 APC_Get_Color_30_mm_depth, 66 APC_GetAWBStatus, 68 APC_GetBusInfo, 69 APC_GetCIllage, 64 APC_InstitubleQvio, 97 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_InstitubleQvio, 99 APC_GetCIPropVal, 71 APC_GetCIPNDVal, 71 APC_InstItableAction, 87 APC_InstItableAction, 87 APC_InitableAction, 98 APC_InititalPlackDevice, 99 APC_GetBusInfo, 69 APC_InstItableAction, 99 APC_I		<del>-</del>
APC_DepthMerge, 56 APC_DisableAWB, 57 APC_DisableAE, 57 APC_DoFusion, 57 APC_DoFusion, 57 APC_DoSwPostProc, 58 APC_EdgePreServingFilter, 59 APC_EnableAE, 59 APC_EnableGPUAcceleration, 60 APC_EnableBesorIF, 61 APC_EncryptMP4, 62 APC_EncryptString, 62 APC_EncryptString, 62 APC_EnindDevice, 63 APC_EnindDevice, 63 APC_GetlentFile, 64 APC_Getlange, 65 APC_InitipOstProcess, 98 APC_InitialCmdFiFo, 97 APC_GetAWBStatus, 68 APC_InitialIdGyro, 97 APC_GetAWBStatus, 68 APC_InitialIdGyro, 97 APC_GetAUGEXposureMode, 68 APC_InitialUAC, 98 APC_InitialUAC, 98 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_InitialEaveDevice, 99 APC_GetBusInfo, 69 APC_InterleaveDevice, 99	,	
APC_DisableAWB, 57 APC_DisableAE, 57 APC_DisableAE, 57 APC_DoFusion, 57 APC_DoFusion, 57 APC_DoSwPostProc, 58 APC_EdgePreServingFilter, 59 APC_EdgePreServingFilter, 59 APC_EnableAWB, 60 APC_EnableAWB, 60 APC_EnableAE, 59 APC_EnableGPUAcceleration, 60 APC_EnableInterleave, 61 APC_EnableSensorIF, 61 APC_EnableSensorIF, 61 APC_EncryptMP4, 62 APC_EncryptMP4, 62 APC_EncryptString, 62 APC_EncryptString, 62 APC_FlyingDepthCancellation_D11, 63 APC_FlyingDepthCancellation_D8, 64 APC_Get2Image, 65 APC_Get2Image, 65 APC_Get_Color_30_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_Get_ACD_ACD, 98 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBlaseLine, 100		-
APC_DisableAE, 57 APC_DoFusion, 57 APC_DoFusion, 57 APC_DoSwPostProc, 58 APC_EdgePreServingFilter, 59 APC_EnableAWB, 60 APC_EnableAWB, 60 APC_EnableAWB, 60 APC_EnableGPUAcceleration, 60 APC_EnableInterleave, 61 APC_EnableSensorIF, 61 APC_EnableSensorIF, 61 APC_EncryptMP4, 62 APC_EncryptMP4, 62 APC_EncryptString, 62 APC_FindDevice, 63 APC_FlyingDepthCancellation_D11, 63 APC_GetZDTable, 93 APC_GetZlmage, 65 APC_GetZlmage, 65 APC_GetZlmage, 65 APC_GetZlmage, 65 APC_Get_Get_Get_Get_Get_Get_Get_Get_Get_Get	_ · ·	
APC_DoFusion, 57 APC_DoSwPostProc, 58 APC_DoSwPostProc, 58 APC_EdgePreServingFilter, 59 APC_EnableAWB, 60 APC_EnableAE, 59 APC_EnableGPUAcceleration, 60 APC_EnableSensorIF, 61 APC_EnableSensorIF, 61 APC_EncyptMP4, 62 APC_EncyptString, 62 APC_EflyingDepthCancellation_D11, 63 APC_GetZDTable, 93 APC_GetZDTable, 93 APC_GetSensorBegister, 90 APC_HoleFill, 94 APC_HoleFill, 94 APC_HoleFilled, 95 APC_GetZDTable, 65 APC_GetZDTable, 66 APC_GetZDTable, 97 APC_GetZDTable, 98 APC_GetZDTable, 98 APC_HoleFilled, 95 APC_HoleFilled, 95 APC_InitSPB, 99 APC_Get_Color_30_mm_depth, 66 APC_InitSPB, 99 APC_GetAWBStatus, 68 APC_InitialFlexibleGyro, 97 APC_GetAWBStatus, 68 APC_InitialHidGyro, 97 APC_GetAutoExposureMode, 68 APC_InjectExtraDataToMp4, 99 APC_GetBusInfo, 69 APC_IsInterleaveDevice, 99 APC_InterleaveDevice, 99		
APC_DoswPostProc, 58 APC_EdgePreServingFilter, 59 APC_EnableAWB, 60 APC_EnableAWB, 60 APC_EnableAE, 59 APC_EnableGPUAcceleration, 60 APC_EnableInterleave, 61 APC_EnableSensorIF, 61 APC_EnableSensorIF, 61 APC_EncryptMP4, 62 APC_EncryptString, 62 APC_FindDevice, 63 APC_FlyingDepthCancellation_D11, 63 APC_GetZDTable, 93 APC_GetJImge, 65 APC_Get_150_mm_depth, 65 APC_Get_Color_30_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_GetAWBStatus, 68 APC_GetAuge, 67 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetCTPropVal, 71 APC_InitcleaveDevice, 99 APC_IniteleaveDevice, 99		
APC_EdgePreServingFilter, 59 APC_EnableAWB, 60 APC_EnableAE, 59 APC_EnableGPUAcceleration, 60 APC_GetSensorRegister, 90 APC_EnableInterleave, 61 APC_EnableSensorIF, 61 APC_EnableSensorIF, 61 APC_GetUACData, 92 APC_EncryptMP4, 62 APC_EncryptString, 62 APC_FlyingDepthCancellation_D11, 63 APC_GetZDTable, 93 APC_FlyingDepthCancellation_D8, 64 APC_GetPlyingDepthCancellation_D8, 64 APC_GetIlsorPlying, 65 APC_Get_150_mm_depth, 65 APC_Get_150_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_GetACDTable, 97 APC_GetACDTable, 97 APC_GetACSTable, 98 APC_InitialCmdFiFo, 97 APC_GetACSTatus, 68 APC_InitialFlexibleGyro, 97 APC_GetACSMBStatus, 68 APC_InitialIldCx, 98 APC_GetBusInfo, 69 APC_IsIMBaseLine, 100	_ · · · · · · · · · · · · · · · · · · ·	
APC_EnableAWB, 60 APC_EnableAE, 59 APC_EnableGPUAcceleration, 60 APC_GetSensorRegister, 90 APC_EnableInterleave, 61 APC_EnableSensorIF, 61 APC_GetUACData, 92 APC_EncryptMP4, 62 APC_EncryptString, 62 APC_EncryptString, 62 APC_ElindDevice, 63 APC_ElindDevice, 63 APC_ElingDepthCancellation_D11, 63 APC_GetZDTable, 93 APC_GetZDTable, 95 APC_GetZlmage, 65 APC_Get_150_mm_depth, 65 APC_Get_60_mm_depth, 66 APC_Get_60_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_GetAEStatus, 67 APC_GetAEStatus, 68 APC_GetACMBStatus, 68 APC_GetBusInfo, 69 APC_GetBusInfo, 69 APC_GetCTPropVal, 71	_ '	
APC_EnableAE, 59 APC_EnableGPUAcceleration, 60 APC_EnableInterleave, 61 APC_EnableSensorIF, 61 APC_EnableSensorIF, 61 APC_EncryptMP4, 62 APC_EncryptString, 62 APC_EncryptString, 62 APC_FlyingDepthCancellation_D11, 63 APC_Get2Image, 65 APC_Get2Image, 65 APC_Get2Image, 65 APC_Get_150_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_GetAEStatus, 67 APC_GetAECDTABLE, 69 APC_GetAUBStatus, 68 APC_GetBusInfo, 69 APC_GetCTPropVal, 71 APC_GetCTPropVal, 71 APC_GetCTPropVal, 71 APC_GetSerialNumber, 90 APC_GetSerialNumber, 90 APC_GetUserData, 92 APC_GetUserData, 92 APC_GetUserData, 92 APC_GetUserData, 92 APC_GetYOffset, 93 APC_GetZDTable, 93 APC_GetZDTable, 93 APC_HoleFill, 94 APC_HoleFilled, 95 APC_HoleFilled, 95 APC_Initi, 96 APC_Initi, 96 APC_InitiPostProcess, 98 APC_InitialCmdFiFo, 97 APC_InitialFlexibleGyro, 97 APC_InitialFlexibleGyro, 97 APC_InitialIndAC, 98 APC_InjectExtraDataToMp4, 99 APC_InitialIndAC, 98 APC_InjectExtraDataToMp4, 99 APC_InitialInderedevice, 99 APC_InitialInderedevice, 99 APC_InitialInderedevice, 99 APC_InitialInderedevice, 100		<del>-</del> • • •
APC_EnableGPUAcceleration, 60 APC_EnableInterleave, 61 APC_EnableInterleave, 61 APC_EnableSensorIF, 61 APC_EnableSensorIF, 61 APC_EncryptMP4, 62 APC_EncryptString, 62 APC_EncryptString, 62 APC_FindDevice, 63 APC_FlyingDepthCancellation_D11, 63 APC_FlyingDepthCancellation_D8, 64 APC_Get2DTable, 95 APC_Get2Image, 65 APC_Get2Image, 65 APC_Get2Image, 65 APC_Get2Image, 65 APC_Get_150_mm_depth, 66 APC_Get_60_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_GetAEStatus, 67 APC_GetACCMeterValue, 67 APC_GetBusInfo, 69 APC_IniterleaveDevice, 99 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100		
APC_EnableInterleave, 61  APC_EnableSensorIF, 61  APC_EncryptMP4, 62  APC_EncryptString, 62  APC_EncryptString, 62  APC_FlyingDepthCancellation_D11, 63  APC_GetZDTable, 93  APC_FlyingDepthCancellation_D8, 64  APC_GetZlmage, 65  APC_GetZlmage, 65  APC_GetZlmage, 65  APC_Get_150_mm_depth, 66  APC_Get_Color_30_mm_depth, 66  APC_GetAEStatus, 67  APC_GetAUBStatus, 68  APC_GetAUGTEAU, 91  APC_InitialCmdFiFo, 97  APC_GetAUGTEAU, 67  APC_InitialUAC, 98  APC_InitialCadp4, 99  APC_GetBusInfo, 69  APC_InitialCadp4, 99  APC_InitialCad		
APC_EnableSensorIF, 61  APC_EncryptMP4, 62  APC_EncryptString, 62  APC_EncryptString, 62  APC_FindDevice, 63  APC_FlyingDepthCancellation_D11, 63  APC_HoleFill, 94  APC_GetPoleFilled, 95  APC_GetPoleFilled, 95  APC_GetPoleFilled, 95  APC_GetPoleFilled, 95  APC_GetPoleFilled, 95  APC_Init; 96  APC_Init; 96  APC_InitSRB, 99  APC_Get_Color_30_mm_depth, 66  APC_InitialCmdFiFo, 97  APC_GetAEStatus, 67  APC_InitialFlexibleGyro, 97  APC_GetACCMeterValue, 67  APC_InitialUAC, 98  APC_InjectExtraDataToMp4, 99  APC_GetBusInfo, 69  APC_IsInterleaveDevice, 99  APC_InterleaveDevice, 99  APC_InterleaveDevice, 100	——————————————————————————————————————	<del>_</del>
APC_EncryptMP4, 62 APC_EncryptString, 62 APC_FindDevice, 63 APC_FlyingDepthCancellation_D11, 63 APC_FlyingDepthCancellation_D8, 64 APC_GetzDTable, 95 APC_GetzDTable, 95 APC_GenerateLutFile, 64 APC_HoleFilled, 95 APC_GetzImage, 65 APC_GetzImage, 65 APC_Get_150_mm_depth, 65 APC_Get_60_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_GetAEStatus, 67 APC_GetAEStatus, 67 APC_GetAWBStatus, 68 APC_InitialFlexibleGyro, 97 APC_GetAutoExposureMode, 68 APC_InjectExtraDataToMp4, 99 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100		
APC_EncryptString, 62 APC_FindDevice, 63 APC_FlyingDepthCancellation_D11, 63 APC_FlyingDepthCancellation_D8, 64 APC_HoleFille, 95 APC_GetzDTable, 95 APC_GenerateLutFile, 64 APC_HoleFilled, 95 APC_GetzImage, 65 APC_Init, 96 APC_InitPostProcess, 98 APC_Get_150_mm_depth, 65 APC_InitPostProcess, 98 APC_Get_Color_30_mm_depth, 66 APC_InitIntCmdFiFo, 97 APC_GetAEStatus, 67 APC_InitialFlexibleGyro, 97 APC_GetAWBStatus, 68 APC_InitialHidGyro, 97 APC_GetAccMeterValue, 67 APC_InitialUAC, 98 APC_InitialUAC, 98 APC_InjectExtraDataToMp4, 99 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100		
APC_FindDevice, 63  APC_FlyingDepthCancellation_D11, 63  APC_FlyingDepthCancellation_D8, 64  APC_GenerateLutFile, 64  APC_Get2Image, 65  APC_Get2Image, 65  APC_Get_150_mm_depth, 65  APC_Get_60_mm_depth, 66  APC_Get_Color_30_mm_depth, 66  APC_InitSRB, 99  APC_GetAEStatus, 67  APC_GetAWBStatus, 68  APC_InitialFlexibleGyro, 97  APC_GetAccMeterValue, 67  APC_GetAutoExposureMode, 68  APC_InitialUAC, 98  APC_InitialUAC, 98  APC_InitialIdata Open Proceedings of the process	APC_EncryptMP4, 62	APC_GetUserData, 92
APC_FlyingDepthCancellation_D11, 63 APC_HoleFill, 94 APC_FlyingDepthCancellation_D8, 64 APC_GenerateLutFile, 64 APC_Get2Image, 65 APC_Get_150_mm_depth, 65 APC_Get_60_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_InitSRB, 99 APC_Get_Color_30_mm_depth, 66 APC_InitalCmdFiFo, 97 APC_GetAEStatus, 67 APC_InitialFlexibleGyro, 97 APC_GetACCMeterValue, 67 APC_InitialUAC, 98 APC_InitialUAC, 98 APC_InitialUAC, 98 APC_GetBusInfo, 69 APC_IniteleaveDevice, 99 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100	APC_EncryptString, 62	APC_GetYOffset, 93
APC_FlyingDepthCancellation_D8, 64 APC_GenerateLutFile, 64 APC_Get2Image, 65 APC_Get_150_mm_depth, 65 APC_Get_60_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_InitSRB, 99 APC_GetAEStatus, 67 APC_InitialCmdFiFo, 97 APC_GetAWBStatus, 68 APC_InitialFlexibleGyro, 97 APC_GetAccMeterValue, 67 APC_GetAutoExposureMode, 68 APC_InitialUAC, 98 APC_InitialUAC, 98 APC_GetBusInfo, 69 APC_InitialFlexibleQvice, 99 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100	APC_FindDevice, 63	APC_GetZDTable, 93
APC_ImgMirro, 95, 96 APC_Get2Image, 65 APC_Get_150_mm_depth, 65 APC_Get_60_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_GetAEStatus, 67 APC_GetAWBStatus, 68 APC_InitialIHidGyro, 97 APC_GetAccMeterValue, 67 APC_GetAutoExposureMode, 68 APC_InitialIUAC, 98 APC_InjectExtraDataToMp4, 99 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100	APC_FlyingDepthCancellation_D11, 63	APC_HoleFill, 94
APC_Get2Image, 65 APC_Get_150_mm_depth, 65 APC_Get_60_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_InitSRB, 99 APC_GetAEStatus, 67 APC_GetAWBStatus, 68 APC_InitialCmdFiFo, 97 APC_GetAWBStatus, 68 APC_InitialFlexibleGyro, 97 APC_GetAccMeterValue, 67 APC_InitialIUAC, 98 APC_InitialUAC, 98 APC_InjectExtraDataToMp4, 99 APC_GetBusInfo, 69 APC_IsInterleaveDevice, 99 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100	APC_FlyingDepthCancellation_D8, 64	APC_HoleFilled, 95
APC_Get2Image, 65 APC_Get_150_mm_depth, 65 APC_Get_60_mm_depth, 66 APC_Get_Color_30_mm_depth, 66 APC_InitSRB, 99 APC_GetAEStatus, 67 APC_GetAWBStatus, 68 APC_InitialCmdFiFo, 97 APC_GetAWBStatus, 68 APC_InitialFlexibleGyro, 97 APC_GetAccMeterValue, 67 APC_InitialIUAC, 98 APC_InitialUAC, 98 APC_InjectExtraDataToMp4, 99 APC_GetBusInfo, 69 APC_IsInterleaveDevice, 99 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100	APC_GenerateLutFile, 64	APC_ImgMirro, 95, 96
APC_Get_150_mm_depth, 65 APC_InitPostProcess, 98 APC_Get_60_mm_depth, 66 APC_InitSRB, 99 APC_Get_Color_30_mm_depth, 66 APC_InitalCmdFiFo, 97 APC_GetAEStatus, 67 APC_InitialFlexibleGyro, 97 APC_GetAWBStatus, 68 APC_InitialHidGyro, 97 APC_GetAccMeterValue, 67 APC_InitialUAC, 98 APC_InitialUAC, 98 APC_InjectExtraDataToMp4, 99 APC_GetBusInfo, 69 APC_IsInterleaveDevice, 99 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100	APC Get2Image, 65	
APC_Get_60_mm_depth, 66 APC_InitSRB, 99 APC_Get_Color_30_mm_depth, 66 APC_InitialCmdFiFo, 97 APC_GetAEStatus, 67 APC_InitialFlexibleGyro, 97 APC_GetAWBStatus, 68 APC_InitialHidGyro, 97 APC_GetAccMeterValue, 67 APC_InitialUAC, 98 APC_GetAutoExposureMode, 68 APC_InjectExtraDataToMp4, 99 APC_GetBusInfo, 69 APC_IsInterleaveDevice, 99 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100	_ •	
APC_Get_Color_30_mm_depth, 66 APC_InitialCmdFiFo, 97 APC_GetAEStatus, 67 APC_GetAWBStatus, 68 APC_InitialFlexibleGyro, 97 APC_GetAccMeterValue, 67 APC_InitialIHidGyro, 97 APC_GetAccMeterValue, 67 APC_InitialUAC, 98 APC_GetAutoExposureMode, 68 APC_InjectExtraDataToMp4, 99 APC_GetBusInfo, 69 APC_IsInterleaveDevice, 99 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100		
APC_GetAEStatus, 67 APC_GetAWBStatus, 68 APC_InitialFlexibleGyro, 97 APC_GetAccMeterValue, 67 APC_GetAutoExposureMode, 68 APC_InjectExtraDataToMp4, 99 APC_GetBusInfo, 69 APC_IsInterleaveDevice, 99 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100		<del>-</del>
APC_GetAWBStatus, 68 APC_InitialHidGyro, 97 APC_GetAccMeterValue, 67 APC_GetAutoExposureMode, 68 APC_InjectExtraDataToMp4, 99 APC_GetBusInfo, 69 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100	·	<del>-</del>
APC_GetAccMeterValue, 67  APC_InitialUAC, 98  APC_GetAutoExposureMode, 68  APC_InjectExtraDataToMp4, 99  APC_GetBusInfo, 69  APC_IsInterleaveDevice, 99  APC_IsMLBaseLine, 100		<del>-</del>
APC_GetAutoExposureMode, 68 APC_InjectExtraDataToMp4, 99 APC_GetBusInfo, 69 APC_IsInterleaveDevice, 99 APC_IsMLBaseLine, 100		
APC_GetBusInfo, 69 APC_IsInterleaveDevice, 99 APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100		<del>-</del>
APC_GetCTPropVal, 71 APC_IsMLBaseLine, 100		
AFO_Geto i natigeatiustep, 72 AFO_OpenDevice, 100		
	AI O_GETOTTIAIIGEATIUSTEP, 72	AI O_OpenDevice, 100

APC_OpenDevice2, 101	GyroTag, 37
APC_OpenDeviceMBL, 102	
APC_PostProcess, 103	InImgHeight
APC_PutSRB, 103	eSPCtrl_RectLogData, 33
APC_RGB2BMP, 109	InImgWidth
APC_ReadCmdFiFo, 104	eSPCtrl_RectLogData, 33
APC_ReadFlashData, 104	
APC_RefreshDevice, 106	LRotaMat
APC_Release, 106	eSPCtrl_RectLogData, 33
APC ReleaseFlexibleGyro, 106	alia-Duffers
APC_ReleaseHidGyro, 107	nLineBuffers
APC ReleasePostProcess, 107	eSPCtrl_RectLogData, 34
APC_ReleaseSwPostProc, 107	NewCamMat1
APC_ReleaseUAC, 108	eSPCtrl_RectLogData, 33
APC_ResetFilters, 108	NewCamMat2
APC_ResizeImgToHalf, 108	eSPCtrl_RectLogData, 34
	Outles of Island
APC_RetrieveExtraDataFromMp4, 109	OutImgHeight
APC_RotateImg180, 110	eSPCtrl_RectLogData, 34
APC_RotateImg90, 111, 112	OutlmgWidth
APC_SaveLutData, 112	eSPCtrl_RectLogData, 34
APC_SelectDevice, 113	packet a 27
APC_SetAutoExposureMode, 113	packet_s, 37
APC_SetCTPropVal, 115	PointCloudInfo, 37
APC_SetColorGain, 113	RECT_AvgErr
APC_SetControlCounterMode, 115	eSPCtrl_RectLogData, 34
APC_SetCurrentlRValue, 116	RECT_Crop_Col_BG_L
APC_SetDepthDataType, 116	eSPCtrl_RectLogData, 34
APC_SetExposureTime, 116	RECT_Crop_Col_ED_L
APC_SetFWRegister, 117	eSPCtrl_RectLogData, 34
APC_SetGlobalGain, 117	RECT_Crop_Row_BG
APC_SetHWRegister, 118	eSPCtrl_RectLogData, 35
APC_SetIRMaxValue, 118	
APC_SetIRMode, 119	RECT_Crop_Row_ED
APC_SetLogData, 119	eSPCtrl_RectLogData, 35 RECT_CropEnable
APC_SetMultiBytesHWRegister, 120	_ ·
APC_SetPUPropVal, 121	eSPCtrl_RectLogData, 35
APC_SetPidVid, 120	RECT_Scale_Col_M
APC_SetRectifyTable, 121	eSPCtrl_RectLogData, 35
APC SetSensorRegister, 122	RECT_Scale_Col_N
APC_SetSensorTypeName, 123	eSPCtrl_RectLogData, 35
APC SetSerialNumber, 123	RECT_Scale_Row_M
APC_SetUserData, 125	eSPCtrl_RectLogData, 35
APC SetYOffset, 125	RECT_Scale_Row_N
APC SetZDTable, 126	eSPCtrl_RectLogData, 35
APC_Setup_v4l2_requestbuffers, 123	RECT_ScaleEnable
APC_SetupBlock, 124	eSPCtrl_RectLogData, 35
APC SetupHidGyro, 124	RECT_ScaleHeight
APC_SubSample, 126	eSPCtrl_RectLogData, 36
	RECT_ScaleWidth
APC_SwitchBaseline, 127	eSPCtrl_RectLogData, 36
APC_TableToData, 127	RRotaMat
APC_TemporalFilter, 128	eSPCtrl_RectLogData, 36
APC_WriteCmdFiFo, 129	RotaMat
APC_WriteFlashData, 129	eSPCtrl_RectLogData, 36
APC_WriteWaveEnd, 130	
APC_WriteWaveHeader, 130	tagAPC_STREAM_INFO, 38
APC_getUACNAME, 92	tagDEVINFORMATION, 38
PDI/eSPDI.h, 41	tagDEVSEL, 38
PDI/eSPDI_def.h, 130	tagKEEP_DATA_CTRL, 38