

eYs3D Linux SDK v5.0.1.6

Generated by Doxygen 1.8.13

# **Contents**

1	Intro	oduction	1		1
2	Clas	ss Index			3
	2.1	Class I	_ist		 3
3	File	Index			5
	3.1	File Lis	st		 5
4	Clas	ss Docu	mentation		7
	4.1	Accele	rationTag \$	ruct Reference	 7
	4.2	APCIm	ageType S	ruct Reference	 7
	4.3	Compa	assTag Stru	t Reference	 8
	4.4	eSPCt	rl_RectLog	ata Struct Reference	 8
		4.4.1	Member	ata Documentation	 9
			4.4.1.1	CamDist1	 9
			4.4.1.2	CamDist2	 9
			4.4.1.3	CamMat1	 9
			4.4.1.4	CamMat2	 9
			4.4.1.5	nImgHeight	 9
			4.4.1.6	nImgWidth	 9
			4.4.1.7	_RotaMat	 9
			4.4.1.8	NewCamMat1	 10
			4.4.1.9	NewCamMat2	 10
			4.4.1.10	nLineBuffers	 10
			4.4.1.11	OutImgHeight	 10

ii CONTENTS

	4.4.1.12	OutlmgWidth	10
	4.4.1.13	RECT_AvgErr	10
	4.4.1.14	RECT_Crop_Col_BG_L	10
	4.4.1.15	RECT_Crop_Col_ED_L	11
	4.4.1.16	RECT_Crop_Row_BG	11
	4.4.1.17	RECT_Crop_Row_ED	11
	4.4.1.18	RECT_CropEnable	11
	4.4.1.19	RECT_Scale_Col_M	11
	4.4.1.20	RECT_Scale_Col_N	11
	4.4.1.21	RECT_Scale_Row_M	11
	4.4.1.22	RECT_Scale_Row_N	11
	4.4.1.23	RECT_ScaleEnable	12
	4.4.1.24	RECT_ScaleHeight	12
	4.4.1.25	RECT_ScaleWidth	12
	4.4.1.26	RotaMat	12
	4.4.1.27	RRotaMat	12
	4.4.1.28	TranMat	12
	4.4.1.29	uByteArray	12
4.5	GyroTag Struct R	Reference	13
4.6	packet_s Struct F	Reference	13
4.7	PointCloudInfo St	truct Reference	13
4.8	tagAPC_STREAM	M_INFO Struct Reference	14
4.9	tagDEVINFORM	ATION Struct Reference	14
4.10	tagDEVSEL Struc	ct Reference	14
4.11	tagKEEP_DATA_	_CTRL Struct Reference	14
4.12	tagZDTableInfo S	Struct Reference	15

CONTENTS

5	File	Docum	entation		17
	5.1	eSPDI	/eSPDI.h F	File Reference	17
		5.1.1	Detailed	Description	26
		5.1.2	Function	Documentation	26
			5.1.2.1	APC_ApplyFilters()	27
			5.1.2.2	APC_CloseCmdFiFo()	27
			5.1.2.3	APC_CloseDevice()	28
			5.1.2.4	APC_CloseDeviceEx()	28
			5.1.2.5	APC_CloseDeviceMBL()	28
			5.1.2.6	APC_ColorFormat_to_RGB24()	29
			5.1.2.7	APC_Convert_Depth_Y_To_Buffer()	29
			5.1.2.8	APC_Convert_Depth_Y_To_Buffer_offset()	30
			5.1.2.9	APC_CreateSwPostProc()	30
			5.1.2.10	APC_DecryptMP4()	31
			5.1.2.11	APC_DecryptString() [1/2]	31
			5.1.2.12	APC_DecryptString() [2/2]	32
			5.1.2.13	APC_DepthMerge()	32
			5.1.2.14	APC_DisableAE()	33
			5.1.2.15	APC_DisableAWB()	33
			5.1.2.16	APC_DoFusion()	34
			5.1.2.17	APC_DoSwPostProc()	34
			5.1.2.18	APC_EdgePreServingFilter()	35
			5.1.2.19	APC_EnableAE()	36
			5.1.2.20	APC_EnableAWB()	36
			5.1.2.21	APC_EnableGPUAcceleration()	36
			5.1.2.22	APC_EnableInterleave()	38
			5.1.2.23	APC_EnableSensorIF()	38
			5.1.2.24	APC_EncryptMP4()	39
			5.1.2.25	APC_EncryptString() [1/2]	39
			5.1.2.26	APC_EncryptString() [2/2]	39

iv CONTENTS

5.1.2.27	APC_FindDevice()	40
5.1.2.28	APC_FlyingDepthCancellation_D11()	40
5.1.2.29	APC_FlyingDepthCancellation_D8()	41
5.1.2.30	APC_GenerateLutFile()	41
5.1.2.31	APC_Get2Image()	42
5.1.2.32	APC_Get_150_mm_depth()	42
5.1.2.33	APC_Get_60_mm_depth()	43
5.1.2.34	APC_Get_Color_30_mm_depth()	43
5.1.2.35	APC_GetAccMeterValue()	44
5.1.2.36	APC_GetAEStatus()	44
5.1.2.37	APC_GetAutoExposureMode()	45
5.1.2.38	APC_GetAWBStatus()	45
5.1.2.39	APC_GetBusInfo()	46
5.1.2.40	APC_GetColorGain()	46
5.1.2.41	APC_GetColorImage()	47
5.1.2.42	APC_GetColorImageWithTimestamp()	47
5.1.2.43	APC_GetControlCounterMode()	48
5.1.2.44	APC_GetCTPropVal()	48
5.1.2.45	APC_GetCTRangeAndStep()	49
5.1.2.46	APC_GetCurrentIRValue()	50
5.1.2.47	APC_GetDepthDataType()	50
5.1.2.48	APC_GetDepthImage()	51
5.1.2.49	APC_GetDepthImageWithTimestamp()	51
5.1.2.50	APC_GetDeviceInfo()	52
5.1.2.51	APC_GetDeviceInfoMBL_15cm()	52
5.1.2.52	APC_GetDeviceNumber()	53
5.1.2.53	APC_GetDeviceResolutionList()	53
5.1.2.54	APC_GetExposureTime()	54
5.1.2.55	APC_GetFlexibleGyroData()	54
5.1.2.56	APC_GetFlexibleGyroLength()	55

CONTENTS

5.1.2.57	APC_GetFWRegister()	55
5.1.2.58	APC_GetFwVersion()	56
5.1.2.59	APC_GetGlobalGain()	56
5.1.2.60	APC_GetHidGyro()	57
5.1.2.61	APC_GetHWRegister()	57
5.1.2.62	APC_GetImage()	58
5.1.2.63	APC_GetImageInterrupt()	58
5.1.2.64	APC_GetInfoHidGyro()	59
5.1.2.65	APC_GetInterleaveMode()	60
5.1.2.66	APC_GetIRMaxValue()	60
5.1.2.67	APC_GetIRMinValue()	61
5.1.2.68	APC_GetIRMode()	61
5.1.2.69	APC_GetLogData()	62
5.1.2.70	APC_GetLutData()	62
5.1.2.71	APC_GetMultiBytesHWRegister()	63
5.1.2.72	APC_GetPidVid()	63
5.1.2.73	APC_GetPointCloud()	64
5.1.2.74	APC_GetPUPropVal()	64
5.1.2.75	APC_GetPURangeAndStep()	65
5.1.2.76	APC_GetRectifyLogData()	66
5.1.2.77	APC_GetRectifyMatLogData()	66
5.1.2.78	APC_GetRectifyTable()	67
5.1.2.79	APC_GetSensorRegister()	67
5.1.2.80	APC_GetSerialNumber()	68
5.1.2.81	APC_GetSRB()	68
5.1.2.82	APC_GetThermalFD()	69
5.1.2.83	APC_GetUACData()	69
5.1.2.84	APC_getUACNAME()	69
5.1.2.85	APC_GetUserData()	70
5.1.2.86	APC_GetYOffset()	70

vi

5.1.2.87 APC_GetZDTable()	71
5.1.2.88 APC_HoleFill()	71
5.1.2.89 APC_HoleFilled()	72
5.1.2.90 APC_ImgMirro() [1/2]	72
5.1.2.91 APC_ImgMirro() [2/2]	74
5.1.2.92 APC_Init()	74
5.1.2.93 APC_InitialCmdFiFo()	75
5.1.2.94 APC_InitialFlexibleGyro()	75
5.1.2.95 APC_InitialHidGyro()	76
5.1.2.96 APC_InitialUAC()	76
5.1.2.97 APC_InitPostProcess()	76
5.1.2.98 APC_InitSRB()	77
5.1.2.99 APC_InjectExtraDataToMp4()	77
5.1.2.100 APC_IsInterleaveDevice()	78
5.1.2.101 APC_IsMLBaseLine()	78
5.1.2.102 APC_OpenDevice()	78
5.1.2.103 APC_OpenDevice2()	79
5.1.2.104 APC_OpenDeviceMBL()	30
5.1.2.105 APC_PostProcess()	31
5.1.2.106 APC_PutSRB()	32
5.1.2.107 APC_ReadCmdFiFo()	32
5.1.2.108 APC_ReadFlashData()	32
5.1.2.109 APC_RefreshDevice()	33
5.1.2.110 APC_Release()	33
5.1.2.111 APC_ReleaseFlexibleGyro()	34
5.1.2.112 APC_ReleaseHidGyro()	34
5.1.2.113 APC_ReleasePostProcess()	34
5.1.2.114 APC_ReleaseSwPostProc()	35
5.1.2.115 APC_ReleaseUAC()	35
5.1.2.116 APC_ResetFilters()	35

CONTENTS vii

5.1.2.117 APC_ResizeImgToHalf()
5.1.2.118 APC_RetrieveExtraDataFromMp4()
5.1.2.119 APC_RGB2BMP()
5.1.2.120 APC_RotateImg180() [1/2]
5.1.2.121 APC_RotateImg180() [2/2]
5.1.2.122 APC_RotateImg90() [1/2]
5.1.2.123 APC_RotateImg90() [2/2]
5.1.2.124 APC_SaveLutData()
5.1.2.125 APC_SelectDevice()
5.1.2.126 APC_SetAETarget()
5.1.2.127 APC_SetAutoExposureMode() 9
5.1.2.128 APC_SetColorGain()
5.1.2.129 APC_SetControlCounterMode()
5.1.2.130 APC_SetCTPropVal()
5.1.2.131 APC_SetCurrentIRValue()
5.1.2.132 APC_SetDepthDataType()
5.1.2.133 APC_SetExposureTime()
5.1.2.134 APC_SetFWRegister()
5.1.2.135 APC_SetGlobalGain()
5.1.2.136 APC_SetHWRegister()
5.1.2.137 APC_SetInterleaveMode()
5.1.2.138 APC_SetIRMaxValue()
5.1.2.139 APC_SetIRMode()
5.1.2.140 APC_SetLogData()
5.1.2.141 APC_SetMultiBytesHWRegister()
5.1.2.142 APC_SetPidVid()
5.1.2.143 APC_SetPUPropVal()
5.1.2.144 APC_SetRectifyTable()
5.1.2.145 APC_SetSensorRegister()
5.1.2.146 APC_SetSensorTypeName()

viii CONTENTS

		5.1.2.147 APC_SetSerialNumber()
		5.1.2.148 APC_Setup_v4l2_requestbuffers()
		5.1.2.149 APC_SetupBlock()
		5.1.2.150 APC_SetupHidGyro()
		5.1.2.151 APC_SetUserData()
		5.1.2.152 APC_SetYOffset()
		5.1.2.153 APC_SetZDTable()
		5.1.2.154 APC_SubSample()
		5.1.2.155 APC_SwitchBaseline()
		5.1.2.156 APC_TableToData()
		5.1.2.157 APC_TemporalFilter()
		5.1.2.158 APC_WriteCmdFiFo()
		5.1.2.159 APC_WriteFlashData()
		5.1.2.160 APC_WriteWaveEnd()
		5.1.2.161 APC_WriteWaveHeader()
5.2	eSPDI	l/eSPDI_def.h File Reference
	5.2.1	Detailed Description

Index

117

# **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

# **Class Index**

## 2.1 Class List

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

AccelerationTag	
APCImageType	7
CompassTag	8
eSPCtrl_RectLogData	8
GyroTag	13
packet_s	13
PointCloudInfo	
agAPC_STREAM_INFO	14
agDEVINFORMATION	14
agDEVSEL	14
agKEEP_DATA_CTRL	14
agZDTableInfo	15

4 Class Index

# **Chapter 3**

# File Index

## 3.1 File List

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

eSPDI/eSPDI.h	
Functions definitions	7
eSPDI/eSPDI_def.h	
Error/data type definitions	8
eSPDI/eSPDI version.h	?

6 File Index

## **Chapter 4**

## **Class Documentation**

## 4.1 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

## 4.2 APCImageType Struct Reference

#### **Public Types**

enum Value {
 IMAGE\_UNKNOWN = -1, COLOR\_YUY2 = 0, COLOR\_RGB24, COLOR\_MJPG,
 COLOR\_UYVY, DEPTH\_8BITS = 100, DEPTH\_8BITS\_0x80, DEPTH\_11BITS,
 DEPTH\_14BITS }

### **Static Public Member Functions**

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

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

8 Class Documentation

## 4.3 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

## 4.4 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]
 }
};
```

#### 4.4.1 Member Data Documentation

```
4.4.1.1 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
4.4.1.2 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
4.4.1.3 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
4.4.1.4 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
4.4.1.5 InImgHeight
unsigned \ short \ eSPCtrl\_RectLogData{\bf ::} InImgHeight
Input image height
4.4.1.6 InImgWidth
unsigned short eSPCtrl_RectLogData::InImgWidth
Input image width(SideBySide image)
4.4.1.7 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|

10 Class Documentation

#### 4.4.1.8 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

#### 4.4.1.9 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

#### 4.4.1.10 nLineBuffers

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

Linebuffer for Hardware limitation < 60

#### 4.4.1.11 OutImgHeight

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

Output image height

#### 4.4.1.12 OutlmgWidth

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

Output image width(SideBySide image)

### 4.4.1.13 RECT\_AvgErr

```
float eSPCtrl_RectLogData::RECT_AvgErr
```

Reprojection error

### 4.4.1.14 RECT\_Crop\_Col\_BG\_L

```
unsigned \ short \ eSPCtrl\_RectLogData::RECT\_Crop\_Col\_BG\_L
```

#### Rectidied image crop column begin

```
4.4.1.15 RECT_Crop_Col_ED_L
unsigned short eSPCtrl_RectLogData::RECT_Crop_Col_ED_L
Rectidied image crop column end
4.4.1.16 RECT_Crop_Row_BG
{\tt unsigned \ short \ eSPCtrl\_RectLogData::RECT\_Crop\_Row\_BG}
Rectidied image crop row begin
4.4.1.17 RECT_Crop_Row_ED
unsigned short eSPCtrl_RectLogData::RECT_Crop_Row_ED
Rectidied image crop row end
4.4.1.18 RECT_CropEnable
int eSPCtrl_RectLogData::RECT_CropEnable
Rectified image crop
4.4.1.19 RECT_Scale_Col_M
unsigned char eSPCtrl_RectLogData::RECT_Scale_Col_M
Rectified image scale column factor M
4.4.1.20 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
4.4.1.21 RECT_Scale_Row_M
unsigned char eSPCtrl_RectLogData::RECT_Scale_Row_M
Rectified image scale row factor M
4.4.1.22 RECT_Scale_Row_N
unsigned char eSPCtrl_RectLogData::RECT_Scale_Row_N
```

Generated by Doxygen

Rectified image scale row factor N

12 Class Documentation

#### 4.4.1.23 RECT\_ScaleEnable

int eSPCtrl\_RectLogData::RECT\_ScaleEnable

Rectified image scale

#### 4.4.1.24 RECT\_ScaleHeight

unsigned short eSPCtrl\_RectLogData::RECT\_ScaleHeight

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

#### 4.4.1.25 RECT\_ScaleWidth

unsigned short eSPCtrl\_RectLogData::RECT\_ScaleWidth

Input image width(Single image) \*RECT Scale Col N/RECT Scale Col M

#### 4.4.1.26 RotaMat

float eSPCtrl\_RectLogData::RotaMat[9]

#### 4.4.1.27 RRotaMat

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

#### 4.4.1.28 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|

#### 4.4.1.29 uByteArray

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

union data defined as below struct { }

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

## 4.5 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

## 4.6 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

#### 4.7 PointCloudInfo Struct Reference

#### **Public Attributes**

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

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

14 Class Documentation

## 4.8 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

## 4.9 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

## 4.10 tagDEVSEL Struct Reference

#### **Public Attributes**

· int index

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

· eSPDI/eSPDI\_def.h

## 4.11 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:

## 4.12 tagZDTableInfo Struct Reference

## **Public Attributes**

- int nlndex
- int nDataType

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

16 Class Documentation

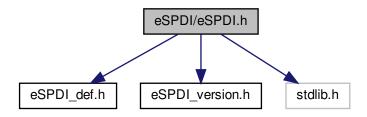
## **Chapter 5**

## **File Documentation**

### 5.1 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 oneming 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.

18 File Documentation

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 SetPixelFormat (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, PIXEL FMT fmt)
- int APC\_SetControlCounterMode (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned char nValue) enable or disable interleave function
- int APC\_GetControlCounterMode (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned char \*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\_SetAETarget (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, int index, float \*EV)
   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

20 File Documentation

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\_
 usec)

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

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

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

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

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

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

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

• int APC\_Get\_150\_mm\_depth (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, BYTE \*pBuf, unsigned long int \*pImageSize, int \*pSerial=0, int nDepthDataType=0)

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

• int APC\_Get2Image (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, BYTE \*pColorImgBuf, BYTE \*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)
- int APC\_GetExposureTime (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, int nSensorMode, float \*pf
   ExpTimeMS)

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)

22 File Documentation

- 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 $\leftarrow$  TY V4L2\_CID\_EXPOSURE\_ABSOLUTE V4L2\_CID\_EXPOSURE V4L2\_CID\_FOCUS\_ABSOLUTE V4L2\_CID\_F OCUS\_RELATIVE V4L2\_CID\_FOCUS\_AUTO V4L2\_CID\_IRIS\_ABSOLUTE V4L2\_CID\_IRIS\_RELATIVE V4L2\_CID\_ZOOM\_ABSOLUTE V4L2\_CID\_ZOOM\_RELATIVE V4L2\_CID\_PAN\_ABSOLUTE V4L2\_CID\_PAN\_RELATIVE V4L2\_CID\_TILT\_ABSOLUTE V4L2\_CID\_TILT\_RELATIVE V4L2\_CID\_PRIVACY
- int APC\_GetPURangeAndStep (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, int 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\_SetInterleaveMode (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, bool enable) set depth data type, 11 bit for disparity data, 14 bit for Z data notice: only PUMA type IC can support this setting
- int APC\_GetInterleaveMode (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, bool \*pValue)
   get current depth data type setting
- int APC\_SetCurrentIRValue (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned short nValue) set infrared radiation(IR) value of PUMA type IC
- int APC\_GetCurrentIRValue (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned short \*pValue) get infrared radiation(IR) value of PUMA type IC
- int APC\_GetIRMinValue (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned short \*pValue) get minimum IR value of camera module
- int APC\_SetIRMaxValue (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned short nValue)
   get maximum IR value of camera module
- int APC\_GetIRMaxValue (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned short \*pValue)
   get maximum IR value of camera module
- int APC\_SetIRMode (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned short nValue)
   enable or disable IRs
- int APC\_GetIRMode (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned short \*pValue)
   to check IR is turn on or off

int APC\_GetRectifyLogData (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, eSPCtrl\_RectLogData \*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.

24 File Documentation

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 BMP.

- int APC\_HoleFilled (unsigned short \*pDImgIn, unsigned short \*pDImgOut, int width, int height, int holeFilldiff)
   Hole Filled.
- int APC\_InitialCmdFiFo (const char \*pfifoName, int \*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\_ColorFormat\_to\_BGR24 (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, unsigned char \*ImgDst, unsigned char \*ImgSrc, int SrcSize, int width, int height, APCImageType::Value type)
- int APC\_RotateImg90 (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, APCImageType::Value imgType, int width, int height, unsigned char \*src, unsigned char \*dstBuf, int len, bool clockwise)

Make the image to rotate.

• int APC\_RotateImg180 (void \*pHandleEYSD, PDEVSELINFO pDevSelInfo, APCImageType::Value imgType, int width, int height, unsigned char \*src, unsigned char \*dst, int len)

Rotate the image to 180 degree.

26 File Documentation

• 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.

#### 5.1.1 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.

#### 5.1.2 Function Documentation

## 5.1.2.1 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

## 5.1.2.2 APC\_CloseCmdFiFo()

Cmd FiFo Close function.

## **Parameters**

int FileDescrption File Description

# Returns

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

## 5.1.2.3 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
```

## 5.1.2.4 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
```

## 5.1.2.5 APC\_CloseDeviceMBL()

close Multiple Base Linedevice and free resource

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index

### Returns

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

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

## 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
```

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

## 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

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

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

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

## **Parameters**

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

# 5.1.2.9 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
```

## 5.1.2.10 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
```

## **5.1.2.11** APC\_DecryptString() [1/2]

# APC\_DecryptString.

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

### Returns

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

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

## 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
```

# 5.1.2.13 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

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

## **Parameters**

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
```

# 5.1.2.14 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
```

# 5.1.2.15 APC\_DisableAWB()

disable auto white balance of ISP

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index

### Returns

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

## 5.1.2.16 APC\_DoFusion()

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

## Do Fusion Merge.

### **Parameters**

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
```

## 5.1.2.17 APC\_DoSwPostProc()

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

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

do software post process on a depth buffer

### **Parameters**

void*	handle handle of this software post process class
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

## 5.1.2.18 APC\_EdgePreServingFilter()

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

# ${\sf APC\_EdgePreServingFilter}.$

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
```

## 5.1.2.19 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
```

# 5.1.2.20 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
```

# 5.1.2.21 APC\_EnableGPUAcceleration()

```
int APC_EnableGPUAcceleration ( \mbox{void} \ * \ pHandleEYSD, \label{eq:phandleEYSD}
```

PDEVSELINFO pDevSelInfo,
bool enable )

 $APC\_Enable GPU Acceleration.$ 

## **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
```

# 5.1.2.22 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
```

## 5.1.2.23 APC\_EnableSensorIF()

### enable or disable sensor IF

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
```

## 5.1.2.24 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

# **5.1.2.25 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

## **5.1.2.26 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
```

## 5.1.2.27 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
```

# 5.1.2.28 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	width depth width
int	height depth height

### Returns

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

## 5.1.2.29 APC\_FlyingDepthCancellation\_D8()

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

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
```

## 5.1.2.30 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
```

## 5.1.2.31 APC\_Get2Image()

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
```

## 5.1.2.32 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

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

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

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

## **Parameters**

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

## 5.1.2.34 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

## 5.1.2.35 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

# 5.1.2.36 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
```

## 5.1.2.37 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
```

## 5.1.2.38 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
```

## 5.1.2.39 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
```

## 5.1.2.40 APC\_GetColorGain()

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

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

### **Parameters**

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
```

## 5.1.2.41 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
```

## 5.1.2.42 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

## 5.1.2.43 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

## 5.1.2.44 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
```

### 5.1.2.45 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.complianuxtv.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

## 5.1.2.46 APC\_GetCurrentIRValue()

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

# 5.1.2.47 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
```

## 5.1.2.48 APC\_GetDepthImage()

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
```

## 5.1.2.49 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

## 5.1.2.50 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

# 5.1.2.51 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
```

## 5.1.2.52 APC\_GetDeviceNumber()

# get EYSD USB device numbers

#### **Parameters**

void	*pHandleEYSD handle
------	---------------------

# Returns

number of EYSD device

## 5.1.2.53 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
```

## 5.1.2.54 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

## 5.1.2.55 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
```

## 5.1.2.56 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
```

# 5.1.2.57 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
```

## 5.1.2.58 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
```

## 5.1.2.59 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
```

## 5.1.2.60 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
```

## 5.1.2.61 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
```

## 5.1.2.62 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
```

# 5.1.2.63 APC\_GetImageInterrupt()

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

## Returns

success: 0, others: not got interrupt

# 5.1.2.64 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
```

# 5.1.2.65 APC\_GetInterleaveMode()

## get current depth data type setting

## **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
bool	*pValue pointer of enable/disable status in device

## Returns

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

## 5.1.2.66 APC\_GetIRMaxValue()

# get maximum IR value of camera module

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
```

## 5.1.2.67 APC\_GetIRMinValue()

## get minimum IR value of camera module

## **Parameters**

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

### Returns

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

# 5.1.2.68 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

## 5.1.2.69 APC\_GetLogData()

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

## get log data from flash

### **Parameters**

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
```

## 5.1.2.70 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

# 5.1.2.71 APC\_GetMultiBytesHWRegister()

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

# set hardware register

### **Parameters**

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

### 5.1.2.72 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

### 5.1.2.73 APC\_GetPointCloud()

# get point cloud

#### **Parameters**

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

# 5.1.2.74 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  $\leftarrow$  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  $\leftarrow$  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

# 5.1.2.75 APC\_GetPURangeAndStep()

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

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

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
```

# 5.1.2.76 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
```

# 5.1.2.77 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

# 5.1.2.78 APC\_GetRectifyTable()

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

get rectify values (file ID 40+) from flash

### **Parameters**

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
```

# 5.1.2.79 APC\_GetSensorRegister()

# get value from sensor register

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
```

# 5.1.2.80 APC\_GetSerialNumber()

# 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
```

# 5.1.2.81 APC\_GetSRB()

### Get Packet from SRB.

# **Parameters**

void	*pSrbHandle pointer to SRB class
packet←	*pPacket Input Packet
_s	

### Returns

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

# 5.1.2.82 APC\_GetThermalFD()

get file description of thermal device

# **Parameters**

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

### Returns

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

# 5.1.2.83 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
```

# 5.1.2.84 APC\_getUACNAME()

Get EYSD UAC Name.

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

### Returns

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

# 5.1.2.85 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
```

# 5.1.2.86 APC\_GetYOffset()

# get Y offset (file ID 30+) value

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
```

# 5.1.2.87 APC\_GetZDTable()

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

# get disparity and Z values from flash

### **Parameters**

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 depth

# Returns

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

# 5.1.2.88 APC\_HoleFill()

# APC\_HoleFill.

# **Parameters**

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

# 5.1.2.89 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

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

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

Make the image to Mirro.

### **Parameters**

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
```

# **5.1.2.91** APC\_ImgMirro() [2/2]

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
```

# 5.1.2.92 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

# **Parameters**

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

### Returns

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

# 5.1.2.93 APC\_InitialCmdFiFo()

Cmd FiFo Initial function.

# **Parameters**

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

# Returns

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

# 5.1.2.94 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
```

# 5.1.2.95 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
```

# 5.1.2.96 APC\_InitialUAC()

UAC inital function.

# **Parameters**

char	*deviceName Point to device Name.
Ciiai	* *uevicemaine Fulli lu device maine.

# Returns

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

# 5.1.2.97 APC\_InitPostProcess()

APC\_InitPostProcess.

# **Parameters**

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

# Returns

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

# 5.1.2.98 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
```

# 5.1.2.99 APC\_InjectExtraDataToMp4()

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

# $APC\_InjectExtraDataToMp4.$

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

### Returns

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

# 5.1.2.100 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

# 5.1.2.101 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.

# 5.1.2.102 APC\_OpenDevice()

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

the implement layer to open EYSD camera device by V4L2(https://en.wikipedia.org/wiki/ $\leftarrow$  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

# 5.1.2.103 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

# 5.1.2.104 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

# 5.1.2.105 APC\_PostProcess()

# APC\_PostProcess.

# **Parameters**

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

### Generated by Doxygen

### Returns

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

# 5.1.2.106 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
```

# 5.1.2.107 APC\_ReadCmdFiFo()

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

# 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
```

# 5.1.2.108 APC\_ReadFlashData()

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

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

#### **Parameters**

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

# 5.1.2.109 APC\_RefreshDevice()

#### refresh all EYSD UVC devices

### **Parameters**

void	*pHandleEYSD handle

### Returns

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

# 5.1.2.110 APC\_Release()

release resource that APC\_Init had allocated

# **Parameters**

Returns

none

# 5.1.2.111 APC\_ReleaseFlexibleGyro()

gyro sensor release function

Returns

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

# 5.1.2.112 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
```

# 5.1.2.113 APC\_ReleasePostProcess()

APC\_ReleasePostProcess.

### **Parameters**

```
void *ppPostProcessHandle [TODO]
```

# Returns

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

### 5.1.2.114 APC\_ReleaseSwPostProc()

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
```

# 5.1.2.115 APC\_ReleaseUAC()

```
int APC_ReleaseUAC (
     void )
```

UAC inital function.

#### Returns

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

# 5.1.2.116 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
```

# 5.1.2.117 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
```

# 5.1.2.118 APC\_RetrieveExtraDataFromMp4()

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

# APC\_RetrieveExtraDataFromMp4.

### **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
```

# 5.1.2.119 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
```

# **5.1.2.120** 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

# 5.1.2.121 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

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

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

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
bClockwise,false	not supported.
bOpencv	useage, not supported.

### Returns

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

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

Make the image to rotate.

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
bool	clockwise clockwise rotate or not

### Returns

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

# 5.1.2.124 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
```

# 5.1.2.125 APC\_SelectDevice()

do not support currently

# Returns

APC\_NotSupport

# 5.1.2.126 APC\_SetAETarget()

set hardware register

# **Parameters**

void	*pHandleEYSD handle
PDEVSELINFO	pDevSelInfo pointer of device select index
int	index range from -6 to 9, 0 is default AE
float	*EV -2.0EV - +3.0EV in 1/3EV step intervals, ie [index, EV] => [-6, -2.00EV] [-5, -1.67EV] [-4, -1.33EV] [-3, -1.00EV] [-1, -0.33EV] [0, 0.00EV] [1, 0.33EV] [2, 0.67EV] [3, 1.00EV] [4, 1.33EV] [5, 1.67EV] [6, 2.00EV] [7, 2.33EV] [8, 2.67EV] [9, 3.00EV]

# Returns

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

# 5.1.2.127 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

# 5.1.2.128 APC\_SetColorGain()

# 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
```

# 5.1.2.129 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

### 5.1.2.130 APC\_SetCTPropVal()

set camera terminal property values By v4l2\_control to set

#### **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
long	int nValue CT property value to set

### Returns

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

# 5.1.2.131 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
```

# 5.1.2.132 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
```

# 5.1.2.133 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
```

# 5.1.2.134 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

# 5.1.2.135 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

# 5.1.2.136 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

# 5.1.2.137 APC\_SetInterleaveMode()

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
bool	enable enable/disable interleave mode see APC_DEPTH_DATA_xxx in eSPDI_def.h

### Returns

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

# 5.1.2.138 APC\_SetIRMaxValue()

get maximum IR value of camera module

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
```

# 5.1.2.139 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
```

# 5.1.2.140 APC\_SetLogData()

# set log data to flash

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
```

# 5.1.2.141 APC\_SetMultiBytesHWRegister()

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

# 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
```

# 5.1.2.142 APC\_SetPidVid()

# set PID and VID to device

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
```

# 5.1.2.143 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
```

# 5.1.2.144 APC\_SetRectifyTable()

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

set rectify values to flash

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
```

# 5.1.2.145 APC\_SetSensorRegister()

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

# set sensor register value

## **Parameters**

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
```

# 5.1.2.146 APC\_SetSensorTypeName()

# set the sensor type you want to work on

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
```

# 5.1.2.147 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
```

# 5.1.2.148 APC\_Setup\_v4I2\_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

## 5.1.2.149 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
```

#### 5.1.2.150 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
```

## 5.1.2.151 APC\_SetUserData()

```
PDEVSELINFO pDevSelInfo,
BYTE * buffer,
int BufferLength,
USERDATA_SECTION_INDEX usi )
```

## 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
```

# 5.1.2.152 APC\_SetYOffset()

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

## 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
```

# 5.1.2.153 APC\_SetZDTable()

```
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
```

# 5.1.2.154 APC\_SubSample()

```
int APC_SubSample (
    void * pHandleEYSD,
    PDEVSELINFO pDevSelInfo,
    unsigned char ** SubSample,
    unsigned char * depthBuf,
    int bytesPerPixel,
    int width,
    int height,
    int & new_width,
    int & new_height,
    int mode = 0,
    int factor = 3 )
```

# APC\_SubSample.

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
```

# 5.1.2.155 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
```

# 5.1.2.156 APC\_TableToData()

## transfer Src to Dst by Table

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
```

# 5.1.2.157 APC\_TemporalFilter()

# APC\_TemporalFilter.

#### **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]

# Returns

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

## 5.1.2.158 APC\_WriteCmdFiFo()

# Write Cmd FiFo function.

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
```

## 5.1.2.159 APC\_WriteFlashData()

```
int APC_WriteFlashData (
    void * pHandleEYSD,
    PDEVSELINFO pDevSelInfo,
    FLASH_DATA_TYPE fdt,
    BYTE * pBuffer,
    unsigned long int BufferLength,
    bool bIsDataVerify,
    KEEP_DATA_CTRL kdc )
```

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

#### **Parameters**

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

# Returns

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

## 5.1.2.160 APC\_WriteWaveEnd()

#### Modified Wave Header.

#### **Parameters**

int fd wave file descript.

#### Returns

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

# 5.1.2.161 APC\_WriteWaveHeader()

```
int APC_WriteWaveHeader ( \label{eq:approx} \mbox{int } fd \; )
```

Write Wave Header.

# **Parameters**

int fd wave file descript.

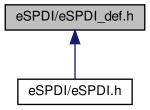
#### Returns

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

# 5.2 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) &=  $\sim$ (1<<(b)))
- #define **BIT\_FLIP**(a, b) ((a)  $^{=}$  (1<<(b)))
- #define **BIT\_CHECK**(a, b) ((a) & (1<<(b)))
- #define FG\_Address\_1Byte 0x01
- #define FG Address 2Byte 0x02
- #define FG\_Value\_1Byte 0x10
- #define FG\_Value\_2Byte 0x20
- #define EVENT\_BUFFER\_SHM\_COLOR "/shm\_ring\_buffer\_color"
- #define EVENT\_BUFFER\_SHM\_DEPTH "/shm\_ring\_buffer\_depth"
- #define EVENT\_BUFFER\_SHM "/shm\_ring\_buffer"
- #define CMD\_FIFO\_PATH "/tmp/cmdfifo"
- #define ZD PATH "/tmp/zd addr"
- #define RECTIFY LOG PATH "/tmp/rectifylog addr"
- #define SRB\_LENGTH 10
- #define CHIPID ADDR 0xf014
- #define SERIAL 2BIT ADDR 0xf0fe
- #define APC\_DEPTH\_DATA\_OFF\_RAW 0 /\* raw (depth off, only raw color) \*/
- #define APC\_DEPTH\_DATA\_DEFAULT APC\_DEPTH\_DATA\_OFF\_RAW /\* raw (depth off, only gray raw color) \*/
- #define APC DEPTH DATA 8 BITS 1 /\* rectify, 1 byte per pixel \*/
- #define APC DEPTH DATA 14 BITS 2 /\* rectify, 2 byte per pixel \*/
- #define APC\_DEPTH\_DATA\_8\_BITS\_x80 3 /\* rectify, 2 byte per pixel but using 1 byte only \*/
- #define APC DEPTH DATA 11 BITS 4 /\* rectify, 2 byte per pixel but using 11 bit only \*/
- #define APC\_DEPTH\_DATA\_OFF\_RECTIFY 5 /\* rectify (depth off, only rectify raw color) \*/
- #define APC\_DEPTH\_DATA\_8\_BITS\_RAW 6 /\* raw \*/
- #define APC\_DEPTH\_DATA\_14\_BITS\_RAW 7 /\* raw \*/
- #define APC\_DEPTH\_DATA\_8\_BITS\_x80\_RAW 8 /\* raw \*/
- #define APC DEPTH DATA 11 BITS RAW 9 /\* raw \*/
- #define APC\_DEPTH\_DATA\_14\_BITS\_COMBINED\_RECTIFY 11
- #define APC\_DEPTH\_DATA\_11\_BITS\_COMBINED\_RECTIFY 13
- #define APC\_DEPTH\_DATA\_OFF\_BAYER\_RAW 14
- #define APC\_DEPTH\_DATA\_INTERLEAVE\_MODE\_OFFSET 16
- #define APC\_DEPTH\_DATA\_ILM\_OFF\_RAW APC\_DEPTH\_DATA\_OFF\_RAW + APC\_DEPTH\_DATA\_← INTERLEAVE\_MODE\_OFFSET /\* raw (depth off, only raw color) \*/

- #define APC\_DEPTH\_DATA\_ILM\_8\_BITS APC\_DEPTH\_DATA\_8\_BITS + APC\_DEPTH\_DATA\_INTER ← LEAVE\_MODE\_OFFSET /\* rectify, 1 byte per pixel \*/
- #define APC\_DEPTH\_DATA\_ILM\_14\_BITS APC\_DEPTH\_DATA\_14\_BITS + APC\_DEPTH\_DATA\_INT

  ERLEAVE MODE OFFSET /\* rectify, 2 byte per pixel \*/
- #define APC\_DEPTH\_DATA\_ILM\_8\_BITS\_x80 APC\_DEPTH\_DATA\_8\_BITS\_x80 + APC\_DEPTH\_DAT
   A\_INTERLEAVE\_MODE\_OFFSET /\* rectify, 2 byte per pixel but using 1 byte only \*/
- #define APC\_DEPTH\_DATA\_ILM\_11\_BITS APC\_DEPTH\_DATA\_11\_BITS + APC\_DEPTH\_DATA\_INT

   ERLEAVE MODE OFFSET /\* rectify, 2 byte per pixel but using 11 bit only \*/
- #define APC\_DEPTH\_DATA\_ILM\_OFF\_RECTIFY APC\_DEPTH\_DATA\_OFF\_RECTIFY + APC\_DEPTH

   DATA INTERLEAVE MODE OFFSET /\* rectify (depth off, only rectify color) \*/
- #define APC\_DEPTH\_DATA\_ILM\_8\_BITS\_RAW APC\_DEPTH\_DATA\_8\_BITS\_RAW + APC\_DEPTH\_D

   ATA INTERLEAVE MODE OFFSET /\* raw \*/
- #define APC\_DEPTH\_DATA\_ILM\_14\_BITS\_RAW APC\_DEPTH\_DATA\_14\_BITS\_RAW + APC\_DEPTH
   —DATA\_INTERLEAVE\_MODE\_OFFSET /\* raw \*/
- #define APC\_DEPTH\_DATA\_ILM\_8\_BITS\_x80\_RAW APC\_DEPTH\_DATA\_8\_BITS\_x80\_RAW + APC\_←
  DEPTH\_DATA\_INTERLEAVE\_MODE\_OFFSET /\* raw \*/
- #define APC\_DEPTH\_DATA\_ILM\_11\_BITS\_RAW APC\_DEPTH\_DATA\_11\_BITS\_RAW + APC\_DEPTH

   DATA\_INTERLEAVE\_MODE\_OFFSET /\* raw \*/
- #define APC\_DEPTH\_DATA\_ILM\_11\_BITS\_COMBINED\_RECTIFY APC\_DEPTH\_DATA\_11\_BITS\_CO
   MBINED\_RECTIFY + APC\_DEPTH\_DATA\_INTERLEAVE\_MODE\_OFFSET
- #define APC DEPTH DATA SCALE DOWN MODE OFFSET 32
- #define APC\_DEPTH\_DATA\_SCALE\_DOWN\_OFF\_RAW (APC\_DEPTH\_DATA\_OFF\_RAW + APC\_DE

  PTH\_DATA\_SCALE\_DOWN\_MODE\_OFFSET)/\* raw (depth off, only raw color) \*/
- #define APC\_DEPTH\_DATA\_SCALE\_DOWN\_DEFAULT (APC\_DEPTH\_DATA\_DEFAULT + APC\_DEP← TH\_DATA\_SCALE\_DOWN\_MODE\_OFFSET) /\* raw (depth off, only raw color) \*/
- #define APC\_DEPTH\_DATA\_SCALE\_DOWN\_8\_BITS (APC\_DEPTH\_DATA\_8\_BITS + APC\_DEPTH\_D

   ATA SCALE DOWN MODE OFFSET)/\* rectify, 1 byte per pixel \*/
- #define APC\_DEPTH\_DATA\_SCALE\_DOWN\_14\_BITS (APC\_DEPTH\_DATA\_14\_BITS + APC\_DEPTH
   —DATA\_SCALE\_DOWN\_MODE\_OFFSET) /\* rectify, 2 byte per pixel \*/
- #define APC\_DEPTH\_DATA\_SCALE\_DOWN\_8\_BITS\_x80 (APC\_DEPTH\_DATA\_8\_BITS\_x80 + APC\_← DEPTH\_DATA\_SCALE\_DOWN\_MODE\_OFFSET) /\* rectify, 2 byte per pixel but using 1 byte only \*/
- #define APC\_DEPTH\_DATA\_SCALE\_DOWN\_11\_BITS (APC\_DEPTH\_DATA\_11\_BITS + APC\_DEPTH
   —DATA\_SCALE\_DOWN\_MODE\_OFFSET)/\* rectify, 2 byte per pixel but using 11 bit only \*/
- #define APC\_DEPTH\_DATA\_SCALE\_DOWN\_OFF\_RECTIFY (APC\_DEPTH\_DATA\_OFF\_RECTIFY + APC\_DEPTH\_DATA\_SCALE\_DOWN\_MODE\_OFFSET) /\* Rule 0.4b Reserved unused in any firmware\*/
- #define APC\_DEPTH\_DATA\_SCALE\_DOWN\_8\_BITS\_RAW (APC\_DEPTH\_DATA\_8\_BITS\_RAW + 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 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 HYPATIA2 0x0173
- #define APC PID 8062 0x0162
- #define APC\_PID\_GRAP 0x0179
- #define APC\_PID\_GRAP\_K 0x0183
- #define APC PID GRAP SLAVE 0x0279
- #define APC PID GRAP SLAVE K 0x0283
- #define APC PID SANDRA 0x0167
- #define APC PID NORA 0x0168
- #define APC\_PID\_HELEN 0x0171
- #define APC\_PID\_GRAP\_THERMAL 0xf9f9
- #define APC PID GRAP THERMAL2 0xf8f8
- #define APC VID GRAP THERMAL 0x04b4
- #define APC VID 2170 0x0110
- #define CT\_PROPERTY\_ID\_AUTO\_EXPOSURE\_MODE\_CTRL 0
- #define CT\_PROPERTY\_ID\_AUTO\_EXPOSURE\_PRIORITY\_CTRL 1
- #define CT\_PROPERTY\_ID\_EXPOSURE\_TIME\_ABSOLUTE\_CTRL 2
- #define CT\_PROPERTY\_ID\_EXPOSURE\_TIME\_RELATIVE\_CTRL 3
- #define CT\_PROPERTY\_ID\_FOCUS\_ABSOLUTE\_CTRL 4
- #define CT PROPERTY ID FOCUS RELATIVE CTRL 5
- #define CT PROPERTY ID FOCUS AUTO CTRL 6
- #define CT\_PROPERTY\_ID\_IRIS\_ABSOLUTE\_CTRL 7
- #define CT\_PROPERTY\_ID\_IRIS\_RELATIVE\_CTRL 8
- #define CT PROPERTY ID ZOOM ABSOLUTE CTRL 9
- #define CT PROPERTY ID ZOOM RELATIVE CTRL 10
- #define CT\_PROPERTY\_ID\_PAN\_ABSOLUTE\_CTRL 11
- #define CT\_PROPERTY\_ID\_PAN\_RELATIVE\_CTRL 12
- #define CT\_PROPERTY\_ID\_TILT\_ABSOLUTE\_CTRL 13
- #define CT\_PROPERTY\_ID\_TILT\_RELATIVE\_CTRL 14
- #define CT PROPERTY ID PRIVACY CTRL 15
- #define PU PROPERTY ID BACKLIGHT COMPENSATION CTRL 0
- #define PU PROPERTY ID BRIGHTNESS CTRL 1
- #define PU\_PROPERTY\_ID\_CONTRAST\_CTRL 2
- #define PU\_PROPERTY\_ID\_GAIN\_CTRL 3
- #define PU PROPERTY ID POWER LINE FREQUENCY CTRL 4
- #define PU PROPERTY ID HUE CTRL 5
- #define PU PROPERTY ID HUE AUTO CTRL 6
- #define PU PROPERTY ID SATURATION CTRL 7
- #define PU\_PROPERTY\_ID\_SHARPNESS\_CTRL 8

- #define PU\_PROPERTY\_ID\_GAMMA\_CTRL 9
- #define PU\_PROPERTY\_ID\_WHITE\_BALANCE\_CTRL 10
- #define PU\_PROPERTY\_ID\_WHITE\_BALANCE\_AUTO\_CTRL 11
- #define POSTPAR HR MODE 5
- #define POSTPAR HR CURVE 06
- #define POSTPAR\_HR\_CURVE\_1 7
- #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 PIXEL FMT {
 YUV22_YUYV_PIXEL_FMT = 0, YUV22_UYVY_PIXEL_FMT, RAW10_GBRG_PIXEL_FMT, RAW10_B↔
 GGR PIXEL FMT,
 RAW10 RGGB PIXEL FMT, RAW10 GRBG PIXEL FMT, MJPEG PIXEL FMT, UNKOWN PIXEL F↔
 MT = 0xffff 
enum DEVICE_TYPE {
 OTHERS = 0, AXES1, PUMA, KIWI,
 UNKNOWN_DEVICE_TYPE = 0xffff }

    enum FLASH_DATA_TYPE {

 Total = 0, FW_PLUGIN, BOOTLOADER_ONLY, FW_ONLY,
 PLUGIN_ONLY }

    enum USERDATA SECTION INDEX {

 USERDATA SECTION 0 = 0, USERDATA SECTION 1, USERDATA SECTION 2, USERDATA SEC
 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,
 {\tt SENSOR\_OFFSET}, {\tt AUTO\_ADJUST\_LOG}, {\tt RECTIFY\_LOG}, {\tt 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,
  \textbf{APC\_SENSOR\_TYPE\_AR1335} = 8, \ \textbf{APC\_SENSOR\_TYPE\_OV9714} = 9, \ \textbf{APC\_SENSOR\_TYPE\_OV9282} 
 = 10, APC_SENSOR_TYPE_H68 = 11,
 APC_SENSOR_TYPE_OV2740 = 12, APC_SENSOR_TYPE_OC0SA10 = 13, APC_SENSOR_TYPE_U ←
 NKOWN = 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 {
  2G = 0, 4G, 6G, 8G,
 _16G }
enum OUTPUT_DATA_RATE {
 One Shot = 0, 1 HZ 1 HZ, 7 HZ 1 HZ, 12 5 HZ 1HZ,
 25 HZ 1 HZ, 7 HZ 7 HZ, 12 5 HZ 12 5 HZ, 25 HZ 25 HZ }
enum POWER_STATE { POWER_ON = 0, POWER_OFF }
```

```
    enum BRIGHTNESS_LEVEL {
    LEVEL_0 = 0, LEVEL_1, LEVEL_2, LEVEL_3,
    LEVEL_4, LEVEL_5, LEVEL_6, LEVEL_7,
    LEVEL_8, LEVEL_9, LEVEL_10, LEVEL_11,
    LEVEL_12, LEVEL_13, LEVEL_14, LEVEL_15 }
```

# 5.2.1 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, 40
eSPDI.h, 26	APC_FlyingDepthCancellation_D11
APC_CloseCmdFiFo	eSPDI.h, 40
eSPDI.h, 27	APC_FlyingDepthCancellation_D8
APC_CloseDevice	eSPDI.h, 41
eSPDI.h, 27	APC_GenerateLutFile
APC_CloseDeviceEx	eSPDI.h, 41
eSPDI.h, 28	APC_Get2Image
APC_CloseDeviceMBL	eSPDI.h, 42
eSPDI.h, 28	APC_Get_150_mm_depth
APC_ColorFormat_to_RGB24	eSPDI.h, 42
eSPDI.h, 29	APC_Get_60_mm_depth
APC_Convert_Depth_Y_To_Buffer	eSPDI.h, 43
eSPDI.h, 29	APC_Get_Color_30_mm_depth
APC_Convert_Depth_Y_To_Buffer_offset	eSPDI.h, 43
eSPDI.h, 30	APC_GetAEStatus
APC_CreateSwPostProc	eSPDI.h, 44
eSPDI.h, 30	APC_GetAWBStatus
APC_DecryptMP4	eSPDI.h, 45
eSPDI.h, 31	APC_GetAccMeterValue
APC_DecryptString	eSPDI.h, 44
eSPDI.h, 31, 32	APC_GetAutoExposureMode
APC_DepthMerge	eSPDI.h, 45
eSPDI.h, 32	APC_GetBusInfo
APC_DisableAWB	eSPDI.h, 46
eSPDI.h, 33	APC_GetCTPropVal
APC_DisableAE	eSPDI.h, 48
eSPDI.h, 33	APC_GetCTRangeAndStep
APC_DoFusion	eSPDI.h, 49
eSPDI.h, 34	APC_GetColorGain
APC_DoSwPostProc	eSPDI.h, 46
eSPDI.h, 34	APC_GetColorImage
APC_EdgePreServingFilter	eSPDI.h, 47
eSPDI.h, 35	APC_GetColorImageWithTimestamp
APC_EnableAWB	eSPDI.h, 47
eSPDI.h, 36	APC_GetControlCounterMode
APC_EnableAE	eSPDI.h, 48
eSPDI.h, 36	APC_GetCurrentIRValue
APC_EnableGPUAcceleration	eSPDI.h, 50
eSPDI.h, 36	APC_GetDepthDataType
APC_EnableInterleave	eSPDI.h, 50
eSPDI.h, 38	APC_GetDepthImage
APC_EnableSensorIF	eSPDI.h, 51
eSPDI.h, 38	APC_GetDepthImageWithTimestamp
APC_EncryptMP4	eSPDI.h, 51
eSPDI.h, 39	APC_GetDeviceInfo
APC_EncryptString	eSPDI.h, 52
eSPDI.h, 39	APC_GetDeviceInfoMBL_15cm
APC FindDevice	eSPDI.h. 52

APC_GetDeviceNumber	APC_GetSerialNumber
eSPDI.h, 53	eSPDI.h, 68
APC GetDeviceResolutionList	APC GetThermalFD
eSPDI.h, 53	eSPDI.h, 68
APC GetExposureTime	APC GetUACData
eSPDI.h, 54	eSPDI.h, 69
APC_GetFWRegister	APC_GetUserData
eSPDI.h, 55	eSPDI.h, 70
APC_GetFlexibleGyroData	APC_GetYOffset
eSPDI.h, 54	eSPDI.h, 70
APC_GetFlexibleGyroLength	APC_GetZDTable
eSPDI.h, 55	eSPDI.h, 71
APC_GetFwVersion	APC_HoleFill
eSPDI.h, 56	eSPDI.h, 71
APC_GetGlobalGain	APC HoleFilled
eSPDI.h, 56	eSPDI.h, 72
APC GetHWRegister	APC ImgMirro
eSPDI.h, 57	eSPDI.h, 72, 74
	• •
APC_GetHidGyro	APC_Init
eSPDI.h, 57	eSPDI.h, 74
APC_GetIRMaxValue	APC_InitPostProcess
eSPDI.h, 60	eSPDI.h, 76
APC_GetIRMinValue	APC_InitSRB
eSPDI.h, 61	eSPDI.h, 77
APC_GetIRMode	APC_InitialCmdFiFo
eSPDI.h, 61	eSPDI.h, 75
APC_GetImage	APC_InitialFlexibleGyro
eSPDI.h, 58	eSPDI.h, 75
APC_GetImageInterrupt	APC_InitialHidGyro
eSPDI.h, 58	eSPDI.h, 76
APC_GetInfoHidGyro	APC InitialUAC
eSPDI.h, 58	eSPDI.h, 76
APC_GetInterleaveMode	APC_InjectExtraDataToMp4
eSPDI.h, 60	eSPDI.h, 77
APC_GetLogData	APC_IsInterleaveDevice
eSPDI.h, 61	eSPDI.h, 78
APC_GetLutData	APC_IsMLBaseLine
eSPDI.h, 62	eSPDI.h, 78
APC_GetMultiBytesHWRegister	APC_OpenDevice
eSPDI.h, 62	eSPDI.h, 78
APC GetPUPropVal	APC OpenDevice2
eSPDI.h, 64	eSPDI.h, 79
APC GetPURangeAndStep	APC_OpenDeviceMBL
eSPDI.h, 65	eSPDI.h, 80
APC GetPidVid	APC PostProcess
eSPDI.h, 63	eSPDI.h, 81
APC_GetPointCloud	APC_PutSRB
eSPDI.h, 63	eSPDI.h, 82
APC_GetRectifyLogData	APC_RGB2BMP
eSPDI.h, 66	eSPDI.h, 87
APC_GetRectifyMatLogData	APC_ReadCmdFiFo
eSPDI.h, 66	eSPDI.h, 82
APC_GetRectifyTable	APC_ReadFlashData
eSPDI.h, 66	eSPDI.h, 82
APC GetSRB	*
<del>-</del>	APC RefreshDevice
eSPDI.h, 68	APC_RefreshDevice eSPDI.h, 83
eSPDI.h, 68 APC GetSensorBegister	eSPDI.h, 83
eSPDI.h, 68 APC_GetSensorRegister eSPDI.h, 67	<del>-</del>

APC_ReleaseFlexibleGyro	APC_SetPidVid
eSPDI.h, 84	eSPDI.h, 98
APC_ReleaseHidGyro	APC_SetRectifyTable
eSPDI.h, 84	eSPDI.h, 99
APC_ReleasePostProcess	APC_SetSensorRegister
eSPDI.h, 84	eSPDI.h, 100
APC_ReleaseSwPostProc	APC_SetSensorTypeName
eSPDI.h, 85	eSPDI.h, 100
APC_ReleaseUAC	APC_SetSerialNumber
eSPDI.h, 85	eSPDI.h, 101
APC_ResetFilters	APC_SetUserData
eSPDI.h, 85	eSPDI.h, 102
APC_ResizeImgToHalf	APC_SetYOffset
eSPDI.h, 86	eSPDI.h, 103
APC_RetrieveExtraDataFromMp4	APC_SetZDTable
eSPDI.h, 86	eSPDI.h, 103
APC_RotateImg180	APC_Setup_v4l2_requestbuffers
eSPDI.h, 87, 88	eSPDI.h, 101
APC_RotateImg90	APC_SetupBlock
eSPDI.h, 88, 89	eSPDI.h, 101
APC_SaveLutData	APC_SetupHidGyro
eSPDI.h, 90	eSPDI.h, 102
APC_SelectDevice	APC_SubSample
eSPDI.h, 90	eSPDI.h, 104
APC_SetAETarget	APC_SwitchBaseline
eSPDI.h, 90	eSPDI.h, 105 APC_TableToData
APC_SetAutoExposureMode	eSPDI.h, 105
eSPDI.h, 91	APC_TemporalFilter
APC_SetCTPropVal	eSPDI.h, 106
eSPDI.h, 92	APC WriteCmdFiFo
APC_SetColorGain	eSPDI.h, 106
eSPDI.h, 91	APC WriteFlashData
APC_SetControlCounterMode	eSPDI.h, 107
eSPDI.h, 92	APC WriteWaveEnd
APC_SetCurrentIRValue	eSPDI.h, 107
eSPDI.h, 93	APC_WriteWaveHeader
APC_SetDepthDataType	eSPDI.h, 108
eSPDI.h, 93	APC_getUACNAME
APC_SetExposureTime	eSPDI.h, 69
eSPDI.h, 94	APCImageType, 7
APC_SetFWRegister	AccelerationTag, 7
eSPDI.h, 94	<b>3</b> ,
APC_SetGlobalGain	CamDist1
eSPDI.h, 95	eSPCtrl_RectLogData, 9
APC_SetHWRegister	CamDist2
eSPDI.h, 95	eSPCtrl_RectLogData, 9
APC_SetIRMaxValue	CamMat1
eSPDI.h, 96	eSPCtrl_RectLogData, 9
APC_SetIRMode	CamMat2
eSPDI.h, 97	eSPCtrl_RectLogData, 9
APC_SetInterleaveMode	CompassTag, 8
eSPDI.h, 96	-000415 11 5 10
APC_SetLogData	eSPCtrl_RectLogData, 8
eSPDI.h, 97	CamDist1, 9
APC_SetMultiBytesHWRegister	CamDist2, 9
eSPDI.h, 98	CamMat1, 9
APC_SetPUPropVal	CamMat2, 9
eSPDI.h, 99	InImgHeight, 9

InImgWidth, 9	APC_GetAWBStatus, 45
LRotaMat, 9	APC_GetAccMeterValue, 44
nLineBuffers, 10	APC_GetAutoExposureMode, 45
NewCamMat1, 9	APC_GetBusInfo, 46
NewCamMat2, 10	APC_GetCTPropVal, 48
OutImgHeight, 10	APC_GetCTRangeAndStep, 49
OutImgWidth, 10	APC_GetColorGain, 46
RECT_AvgErr, 10	APC_GetColorImage, 47
RECT_Crop_Col_BG_L, 10	APC_GetColorImageWithTimestamp, 47
RECT_Crop_Col_ED_L, 10	APC_GetControlCounterMode, 48
RECT_Crop_Row_BG, 11	APC_GetCurrentIRValue, 50
RECT_Crop_Row_ED, 11	APC_GetDepthDataType, 50
RECT_CropEnable, 11	APC_GetDepthImage, 51
RECT_Scale_Col_M, 11	APC_GetDepthImageWithTimestamp, 51
RECT_Scale_Col_N, 11	APC_GetDeviceInfo, 52
RECT_Scale_Row_M, 11	APC_GetDeviceInfoMBL_15cm, 52
RECT_Scale_Row_N, 11	APC_GetDeviceNumber, 53
RECT_ScaleEnable, 11	APC_GetDeviceResolutionList, 53
RECT_ScaleHeight, 12	APC_GetExposureTime, 54
RECT_ScaleWidth, 12	APC_GetFWRegister, 55
RRotaMat, 12	APC_GetFlexibleGyroData, 54
RotaMat, 12	APC_GetFlexibleGyroLength, 55
TranMat, 12	APC_GetFwVersion, 56
uByteArray, 12	APC_GetGlobalGain, 56
eSPDI.h	APC_GetHWRegister, 57
APC_ApplyFilters, 26	APC_GetHidGyro, 57
APC_CloseCmdFiFo, 27	APC_GetIRMaxValue, 60
APC_CloseDevice, 27	APC_GetIRMinValue, 61
APC_CloseDeviceEx, 28	APC_GetIRMode, 61
APC_CloseDeviceMBL, 28	APC_GetImage, 58
APC_ColorFormat_to_RGB24, 29	APC_GetImageInterrupt, 58
APC Convert Depth Y To Buffer, 29	APC_GetInfoHidGyro, 58
APC_Convert_Depth_Y_To_Buffer_offset, 30	APC GetInterleaveMode, 60
APC CreateSwPostProc, 30	APC_GetLogData, 61
APC_DecryptMP4, 31	APC GetLutData, 62
APC_DecryptString, 31, 32	APC_GetMultiBytesHWRegister, 62
APC_DepthMerge, 32	APC_GetPUPropVal, 64
APC_DisableAWB, 33	APC_GetPURangeAndStep, 65
APC DisableAE, 33	APC_GetPidVid, 63
APC_DoFusion, 34	APC_GetPointCloud, 63
APC DoSwPostProc, 34	APC GetRectifyLogData, 66
APC_EdgePreServingFilter, 35	APC_GetRectifyMatLogData, 66
APC EnableAWB, 36	APC_GetRectifyTable, 66
<del>-</del>	
APC_EnableAE, 36	APC_GetSanor Posintar, 67
APC_EnableGPUAcceleration, 36	APC_GetSensorRegister, 67
APC_EnableInterleave, 38	APC_GetSerialNumber, 68
APC_EnableSensorIF, 38	APC_GetThermalFD, 68
APC_EncryptMP4, 39	APC_GetUACData, 69
APC_EncryptString, 39	APC_GetUserData, 70
APC_FindDevice, 40	APC_GetYOffset, 70
APC_FlyingDepthCancellation_D11, 40	APC_GetZDTable, 71
APC_FlyingDepthCancellation_D8, 41	APC_HoleFill, 71
APC_GenerateLutFile, 41	APC_HoleFilled, 72
APC_Get2Image, 42	APC_ImgMirro, 72, 74
APC_Get_150_mm_depth, 42	APC_Init, 74
APC_Get_60_mm_depth, 43	APC_InitPostProcess, 76
APC_Get_Color_30_mm_depth, 43	APC_InitSRB, 77
APC_GetAEStatus, 44	APC_InitialCmdFiFo, 75

APC_InitialFlexibleGyro, 75 APC_InitialHidGyro, 76 APC_InitialUAC, 76 APC_InjectExtraDataToMp4, 77 APC_IsInterleaveDevice, 78 APC_IsMLBaseLine, 78 APC_OpenDevice, 78 APC_OpenDevice2, 79 APC_OpenDevice4, 80	APC_TableToData, 105 APC_TemporalFilter, 106 APC_WriteCmdFiFo, 106 APC_WriteFlashData, 107 APC_WriteWaveEnd, 107 APC_WriteWaveHeader, 108 APC_getUACNAME, 69 eSPDI/eSPDI.h, 17 eSPDI/eSPDI_def.h, 108
APC_PostProcess, 81 APC_PutSRB, 82	GyroTag, 13
APC_RGB2BMP, 87	InImgHeight
APC_ReadCmdFiFo, 82 APC ReadFlashData, 82	eSPCtrl_RectLogData, 9
APC_headriashData, 62 APC RefreshDevice, 83	InImgWidth
APC_Refreshibevice, 63  APC Release, 83	eSPCtrl_RectLogData, 9
APC ReleaseFlexibleGyro, 84	I DeteMet
APC ReleaseHidGyro, 84	LRotaMat
APC ReleasePostProcess, 84	eSPCtrl_RectLogData, 9
APC_ReleaseSwPostProc, 85	nLineBuffers
APC ReleaseUAC, 85	eSPCtrl_RectLogData, 10
APC_ResetFilters, 85	NewCamMat1
APC_ResizeImgToHalf, 86	eSPCtrl_RectLogData, 9
APC_RetrieveExtraDataFromMp4, 86	NewCamMat2
APC_RotateImg180, 87, 88	eSPCtrl_RectLogData, 10
APC_RotateImg90, 88, 89	
APC_SaveLutData, 90	OutImgHeight
APC_SelectDevice, 90	eSPCtrl_RectLogData, 10 OutImgWidth
APC_SetAETarget, 90	eSPCtrl_RectLogData, 10
APC_SetAutoExposureMode, 91	esroiii_neciLogDaia, io
APC_SetCTPropVal, 92	packet_s, 13
APC_SetColorGain, 91	PointCloudInfo, 13
APC_SetControlCounterMode, 92 APC SetCurrentIRValue, 93	
APC_SetDepthDataType, 93	RECT_AvgErr
APC_SetExposureTime, 94	eSPCtrl_RectLogData, 10
APC SetFWRegister, 94	RECT_Crop_Col_BG_L
APC_SetGlobalGain, 95	eSPCtrl_RectLogData, 10
APC_SetHWRegister, 95	RECT_Crop_Col_ED_L eSPCtrl_RectLogData, 10
APC_SetIRMaxValue, 96	RECT_Crop_Row_BG
APC_SetIRMode, 97	eSPCtrl_RectLogData, 11
APC_SetInterleaveMode, 96	RECT_Crop_Row_ED
APC_SetLogData, 97	eSPCtrl_RectLogData, 11
APC_SetMultiBytesHWRegister, 98	RECT_CropEnable
APC_SetPUPropVal, 99	eSPCtrl_RectLogData, 11
APC_SetPidVid, 98	RECT_Scale_Col_M
APC_SetRectifyTable, 99	eSPCtrl_RectLogData, 11
APC_SetSensorRegister, 100	RECT_Scale_Col_N
APC_SetSensorTypeName, 100	eSPCtrl_RectLogData, 11
APC_SetUser(Date 100	RECT_Scale_Row_M
APC_SetVOffeet 102	eSPCtrl_RectLogData, 11
APC_SetYOffset, 103 APC_SetZDTable, 103	RECT_Scale_Row_N
APC_Setup_v4l2_requestbuffers, 101	eSPCtrl_RectLogData, 11 RECT_ScaleEnable
APC_SetupBlock, 101	eSPCtrl_RectLogData, 11
APC_SetupHidGyro, 102	RECT_ScaleHeight
APC_SubSample, 104	eSPCtrl_RectLogData, 12
APC_SwitchBaseline, 105	RECT_ScaleWidth
	— <del>-</del>