eYs3D Windows SDK

1.5.0.0

Generated by Doxygen 1.9.1

Chapter 1

Introduction

This document describes the usage of Application Programming Interfaces of eYs3D Windows SDK

What's inside the SDK

Table 1.1 File List

Folder	Subfolder	Filename	Description
bin	Win32	All files	Sample executables on Win32 platform
	x64	All files	Sample executables on Windows 64-bits platform
eSPDI	include	eSPDI_Common.h	Basic API declaration header
		eSPDI_DM.h	Depth Map specific API declaration header
		eSPDI_ErrCode.h	Error code definitions
	Win32	eSPDI_DM.dll	eSPDI dynamical linked library for Win32 platform
		eSPDI_DM.lib	eSPDI static linked library for Win32 platform
	x64	eSPDI_DM.dll	eSPDI dynamical linked library for Windows 64-bits
		eSPDI_DM.lib	eSPDI static linked library for Windows 64-bits
doc	html	index.html	This documentation
DMPreview			A sample VC++ project demonstrating how to open multiple devices in an application

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

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

Chapter 4

Data Structure Documentation

4.1 DEVINFORMATIONEX

Data Fields

- unsigned short wPID
- unsigned short wVID
- char strDevName [512]
- unsigned short nChipID
- unsigned short nDevType
- unsigned short wUsbNode

4.1.1 Detailed Description

extended device information class

4.1.2 Field Documentation

4.1.2.1 nChipID

```
unsigned short nChipID
```

chip ID, 0x18 for AXES1, 0x1C for KIWI, 0x15 for PUMA

4.1.2.2 nDevType

```
unsigned short nDevType
```

chip enum value, see APC_DEVICE_TYPE

4.1.2.3 strDevName

```
char strDevName[512]
```

device name

4.1.2.4 wPID

```
unsigned short wPID
```

product ID

Table 4.1 PID List

Chip Name	Chip ID	PID
AXES1	0x18	0x0568
		0x0668
		0x0113
		0x0115
		0x0116
KIWI	0x1C	0x0118
PUMA	0x15	0x0112
		0x0120

4.1.2.5 wUsbNode

```
unsigned short wUsbNode
```

USB Node

4.1.2.6 wVID

```
unsigned short wVID
```

vender ID, 0x1E4E for ApcDI device

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

- eSPDI_Common.h

4.2 eSPCtrl_RectLogData

eSPCtrl_RectLogData

4.2.1 Detailed Description

eSPCtrl_RectLogData

Rectified log data structure

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

- eSPDI_Common.h

4.3 ParaLUT

ParaLUT.

Data Fields

- long long file_ID_header
  [00]-[000] File ID header : 2230
- long long file_ID_version
  [01]-[008] File ID version : 4
- double FOV
  [02]-[016] Field of view with degree
- long long semi_FOV_pixels
  [03]-[024] Pixels for semi-FOV
- long long img_src_cols
  [04]-[032] Width for source image (single image)
- long long img_src_rows
  [05]-[040] Height for source image
- double img_L_src_col_center
  [06]-[048] Center of width for L side source image
- double img_L_src_row_center
  [07]-[056] Center of height for L side source image
- double img_R_src_col_center
  [08]-[064] Center of width for R side source image
- double img_R_src_row_center
  [09]-[072] Center of height for R side source image
- double img_L_rotation
  [10]-[080] Rotation for L side image
- double img_R_rotation
  [11]-[088] Rotation for R side image
- double spline_control_v1
  [12]-[096] Spline control value for row = DIV x 0 pixel, DIV = rows/6
- double spline_control_v2
  [13]-[104] Spline control value for row = DIV x 1 pixel, DIV = rows/6
- double spline_control_v3
  [14]-[112] Spline control value for row = DIV x 2 pixel, DIV = rows/6
- double spline_control_v4
  [15]-[120] Spline control value for row = DIV x 3 pixel, DIV = rows/6

- double spline_control_v5

  [16]-[128] Spline control value for row = DIV x 4 pixel, DIV = rows/6
- double spline_control_v6

  [17]-[136] Spline control value for row = DIV x 5 pixel, DIV = rows/6
- double spline_control_v7

  [18]-[144] Spline control value for row = DIV x 6 pixel, DIV = rows/6
- long long img_dst_cols

  [19]-[152] Width for output image (single image), according to "Original" parameters
- long long img_dst_rows

  [20]-[160] Height for output image, according to "Original" parameters
- long long img_L_dst_shift

  [21]-[168] Output L side image shift in row
- long long img_R_dst_shift

  [22]-[176] Output R side image shift in row
- long long img_overlay_LR

  [23]-[184] Overlay between L/R in pixels, far field, (YUV must be even)
- long long img_overlay_RL

  [24]-[192] Overlay between R/L in pixels, far field, (YUV must be even)
- long long img_stream_cols

  [25]-[200] Output image stream of cols
- long long img_stream_rows

  [26]-[208] Output image stream of rows
- long long video_stream_cols

  [27]-[216] Output video stream of cols
- long long video_stream_rows

  [28]-[224] Output video stream of rows
- long long usb_type

  [29]-[232] 2 for usb2, 3 for usb3
- long long img_type

  [30]-[240] 1 for yuv422, 2 for BGR, 3 for RGB
- long long lut_type

  [31]-[248] Output LUT tye eys::LutModes
- long long blending_type

  [32]-[256] 0 for choosed by function, 1 for alpha-blending, 2 for Laplacian pyramid blending
- double overlay_ratio

  [33]-[264] far field overlay value is equal to img_overlay_LR(RL) = overlay_value + overlay_ratio
- long long serial_number_date0

  [34]-[272] 8 bytes, yyyy-mm-dd
- long long serial_number_date1

  [35]-[280] 8 bytes, hh-mm-ss-xxx, xxx for machine number
- double unit_sphere_radius

  [36]-[288] Original : Unit spherical radius for dewarping get x and y
- double min_col

  [37]-[296] Original : Parameters of min position of image width
- double max_col

  [38]-[304] Original : Parameters of max position of image width
- double min_row

  [39]-[312] Original : Parameters of min position of image height
- double max_row

[40]-[320] Original : Parameters of max position of image height

- long long AGD_LR

  [41]-[328] Err : Average gray-level value discrepancy at LR boundary
- long long AGD_RL

  [42]-[336] Err : Average gray-level value discrepancy at RL boundary
- long long out_img_resolution

  [43]-[344] Set output resolution eys::ImgResolutionModes
- long long out_lut_cols

  [44]-[352] Output side-by-side lut width, according to the set of out_img_resolution
- long long out_lut_rows

  [45]-[360] Output lut height, according to the set of out_img_resolution
- long long out_lut_cols_eff

  [46]-[368] Output effective pixels in out_lut_cols, 0 is for all
- long long out_lut_rows_eff

  [47]-[376] Output effecitve pixels in out_lut_rows, 0 is for all
- long long out_img_cols

  [48]-[384] Output side-by-side image width after dewarping and stitching, according to the set of out_img_resolution
- long long out_img_rows

  [49]-[392] Output image height, according to the set of out_img_resolution
- long long out_overlay_LR

  [50]-[340] Output L/R overlay value, according to the set of out_img_resolution
- long long out_overlay_RL

  [51]-[408] Output R/L overlay value, according to the set of out_img_resolution
- long long reserve [44]

  [52]-[416] Reserve 44 parameter to use

4.3.1 Detailed Description

ParaLUT.

Spherical look-up table conversion parameters

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

- eSPDI_Common.h

4.4 tagDEVINFORMATION

DEVINFORMATION.

Data Fields

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

4.4.1 Detailed Description

DEVINFORMATION.

device information

4.4.2 Field Documentation

4.4.2.1 nChipID

```
unsigned short nChipID
```

chip ID, 0x18 for AXES1, 0x1C for KIWI, 0x15 for PUMA

4.4.2.2 nDevType

```
unsigned short nDevType
```

chip enum value,

See also

APC_DEVICE_TYPE

4.4.2.3 strDevName

```
char* strDevName
```

pointer to device name stored inside the SDK

4.4.2.4 wPID

```
unsigned short wPID
```

product ID

Table 4.2 PID List

Chip Name	Chip ID	PID
AXES1	0x18	0x0568
		0x0668
		0x0113
		0x0115
		0x0116
KIWI	0x1C	0x0118
PUMA	0x15	0x0112
		0x0120

4.4.2.5 wUsbNode

```
unsigned short wUsbNode
```

USB Node

4.4.2.6 wVID

```
unsigned short wVID
```

vender ID, 0x1E4E for ApcDI device

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

- eSPDI_Common.h

Chapter 5

File Documentation

5.1 eSPDI_Common.h File Reference

eYs3D SDK API export functions, data structure and variable definition

Data Structures

- struct eSPCtrl_RectLogData
  eSPCtrl_RectLogData
- struct ParaLUT
  ParaLUT.
- struct tagDEVINFORMATION
  DEVINFORMATION.
- class DEVINFORMATIONEX

Typedefs

- typedef struct eSPCtrl_RectLogData eSPCtrl_RectLogData
  eSPCtrl_RectLogData
- typedef struct ParaLUT PARALUT
  ParaLUT.
- typedef struct tagDEVINFORMATION DEVINFORMATION
  DEVINFORMATION.

Enumerations

- enum APC_DEVICE_TYPE { OTHERS = 0 , AXES1 , PUMA , **PLUM** }
- enum USERDATA_SECTION_INDEX {
  USERDATA_SECTION_0 = 0 , USERDATA_SECTION_1 , USERDATA_SECTION_2 , USERDATA_SECTION_3 ,
  USERDATA_SECTION_4 , USERDATA_SECTION_5 , USERDATA_SECTION_6 , USERDATA_SECTION_7 ,
  USERDATA_SECTION_8 , USERDATA_SECTION_9 , USERDATA_SECTION_10 , USERDATA_SECTION_NUM
  }
- enum SENSOR_TYPE_NAME {
  APC_SENSOR_TYPE_H22 = 0 , APC_SENSOR_TYPE_OV7740 , APC_SENSOR_TYPE_AR0134 ,
  APC_SENSOR_TYPE_AR0135 ,
  APC_SENSOR_TYPE_AR0144 , APC_SENSOR_TYPE_OV9714 , APC_SENSOR_TYPE_OV9282 ,
  APC_SENSOR_TYPE_AR0330 ,
  APC_SENSOR_TYPE_AR1335 , **APC_SENSOR_TYPE_H65** , **APC_SENSOR_TYPE_AR0522** }

Functions

- int APC_API APC_Init (void **ppHandleApcDI, bool bIsLogEnabled)

  entry point of eYs3D camera SDK. This API allocates resource and find all the eSPI camera devices connected to the system.
- int APC_API APC_Init2 (void **ppHandleApcDI, bool bIsLogEnabled, bool bAutoRestart)

  entry point of eYs3D camera SDK. This API allocates resource and find all the eSPI camera devices connected to the system.
- int APC_API APC_RegisterDeviceEvents (void *pHandleApcDI, APC_DeviceEventFn cbFunc, void *pData)

  Register the USB device plug or unplug events. Any USB capture device attachment or detachment events will call the callback function cbFunc.
- void APC_API APC_Release (void **ppHandleApcDI)

  release all resource that APC_Init had allocated
- int APC_API APC_FindDevice (void *pHandleApcDI)

  find out all eYs3D USB devices by PID, VID and ChipID, also remember device types
- int APC_API APC_RefreshDevice (void *pHandleApcDI)

  refresh all eYs3D UVC devices
- int APC_API APC_GetDeviceNumber (void *pHandleApcDI)

  get eYs3D USB device numbers
- int APC_API APC_GetSlaveSensorRegister (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, int nId, unsigned short address, unsigned short *pValue, int flag, int nSensorMode)

  get value from sensor register
- int APC_API APC_GetSensorRegister (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, int nId, unsigned short address, unsigned short *pValue, int flag, int nSensorMode)

  get value from sensor register
- int APC_API APC_GetFWRegister (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, unsigned short address, unsigned short *pValue, int flag)

  get firmware register value
- int APC_API APC_SetFWRegister (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, unsigned short address, unsigned short nValue, int flag)

  set firmware register value
- int APC_API APC_GetSlaveHWRegister (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, unsigned short address, unsigned short *pValue, int flag)

  get hardware register value
- int APC_API APC_GetHWRegister (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, unsigned short address, unsigned short *pValue, int flag)

  get hardware register value
- int APC_API APC_SetSlaveHWRegister (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, unsigned short address, unsigned short nValue, int flag)

  set hardware register
- int APC_API APC_SetHWRegister (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, unsigned short address, unsigned short nValue, int flag)

  set hardware register
- int APC_API APC_GetFwVersion (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, char *pszFwVersion, int nBufferSize, int *pActualLength)

  get the firmware version of device, the version is a string
- int APC_API APC_GetPidVid (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, unsigned short *pPidBuf, unsigned short *pVidBuf)

  get PID(product ID) and VID(vendor ID) of device
- int APC_API APC_SetPidVid (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, unsigned short *pPidBuf, unsigned short *pVidBuf)

  set PID and VID to device

- int APC_API APC_GetSlaveLogData (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, int *pActualLength, int index)

  get log data from flash
- int APC_API APC_GetLogData (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, int *pActualLength, int index)

  get log data from flash
- int APC_API APC_SetSlaveLogData (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, int *pActualLength, int index)

  set log data to flash
- int APC_API APC_SetLogData (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, int *pActualLength, int index)

  set log data to flash
- int APC_API APC_SetUserData (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, USERDATA_SECTION_INDEX usi)

  set user data to flash
- int APC_API APC_ReadFlashData (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, FLASH_DATA_TYPE fdt, BYTE *pBuffer, unsigned long int nLengthOfBuffer, unsigned long int *pActualBufferLen)

  read firmware code(.bin) form flash The firmware code is the combination of boot loader, firmware body and plug-in data. This input buffer length has to match with the flash data type
- int APC_API APC_OpenDevice (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, int colorStreamIndex, int depthStreamIndex, int depthStreamSwitch, int iFps, APC_ImgCallbackFn callbackFn, void *pCallbackParam, int pid=-1)

  open camera device with image callback support
- int APC_API APC_GetColorImage (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *pBuf, unsigned long int *pImageSize, int *pSerial=NULL)

  get color image
- int APC_API APC_CloseDevice (void *pHandleApcDI, PDEVSELINFO pDevSelInfo)

  close device and stop video render
- int APC_API APC_GetDeviceResolutionList (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, int nMaxCount0, APC_STREAM_INFO *pStreamInfo0, int nMaxCount1, APC_STREAM_INFO *pStreamInfo1)

  get the device resolution list
- bool APC_API APC_Is360Device (void *pHandleApcDI, PDEVSELINFO pDevSelInfo)

  check module is spherical device or not
- int APC_API APC_GetSerialNumberFromLog (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, char *pSerialNum, int nBufferSize, int *pActualLength)

  get the module serial number
- int APC_API APC_SetCurrentIRValue (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, WORD wType)

  set current infrared radiation(IR) value
- int APC_API APC_GetCurrentIRValue (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, WORD *pwType)

  get current infrared radiation(IR) value
- int APC_API APC_GetIRMinValue (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, WORD *pwType)

  get minimum IR value the module support
- int APC_API APC_SetIRMaxValue (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, WORD wType)

  set maximum IR value the module support
- int APC_API APC_GetIRMaxValue (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, WORD *pwType)

  get maximum IR value the module support
- int APC_API APC_SetIRMode (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, WORD wType)

  set IR mode, left, right or both
- int APC_API APC_GetIRMode (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, WORD *pwType)

  set IR mode, left, right or both
- int APC_API APC_EnableSensorIF (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, bool bIsEnable)

  turn on/off sensor IF function

- int APC_API APC_SetSensorTypeName (void *pHandleApcDI, SENSOR_TYPE_NAME stn)

  select which sensor to operate
- int APC_API APC_EnableAE (void *pHandleApcDI, PDEVSELINFO pDevSelInfo)

  enable auto exposure function of ISP
- int APC_API APC_DisableAE (void *pHandleApcDI, PDEVSELINFO pDevSelInfo)

  disable auto exposure function of ISP
- int APC_API APC_EnableAWB (void *pHandleApcDI, PDEVSELINFO pDevSelInfo)

  enable auto white balance function of ISP
- int APC_API APC_DisableAWB (void *pHandleApcDI, PDEVSELINFO pDevSelInfo)

  disable auto white balance of ISP
- int APC_API APC_GetGPIOValue (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, int nGPIOIndex, BYTE *pValue)

  get general purpose IO value
- int APC_API APC_SetGPIOValue (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, int nGPIOIndex, BYTE nValue)

  set GPIO value
- int APC_API APC_SetGPIOCtrl (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, int nGPIOIndex, BYTE nValue)

  set GPIO control address
- int APC_API APC_GetPUPropVal (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, int nId, int *pValue)

  get processing unit property value `https://msdn.microsoft.com/en-us/library/windows/hardware/ff568122(85).aspx` *The PROPSETID_VIDCAP_VIDEOPROCAMP property set controls devices that can adjust image color attributes of analog or digital signals.*
- int APC_API APC_SetPUPropVal (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, int nId, int nValue)

  get processing unit property value `https://msdn.microsoft.com/en-us/library/windows/hardware/ff568122(85).aspx` `https://msdn.microsoft.com/en-us/library/windows/hardware/ff566089(v=vs.↩85).aspx`
- int APC_API APC_GetCTPropVal (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, int nId, int *pValue)

  set control terminal property value `https://msdn.microsoft.com/en-us/library/windows/hardware/ff567802(85).aspx` *The PROPSETID_VIDCAP_CAMERACONTROL property set controls camera device settings. The controls it provides are a subset of the ITU T.RDC standard.*
- int APC_API APC_SetCTPropVal (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, int nId, int nValue)

  get control terminal property value `https://msdn.microsoft.com/en-us/library/windows/hardware/ff567802(85).aspx` `https://msdn.microsoft.com/en-us/library/windows/hardware/ff566089(v=vs.↩85).aspx`
- int APC_API APC_GetAutoExposureMode (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, unsigned short *mode)

  misc function : get auto exposure mode
- int APC_API APC_SetAutoExposureMode (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, unsigned short mode)

  misc function : set auto exposure mode
- int APC_API APC_GetFlexibleGyroData (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, int length, BYTE *pGyroData)

  get IMU(Gyro) data
- int APC_API APC_GetFlexibleGyroLength (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, unsigned short *GyroLen)

  get the IMU(Gyro) data length
- int APC_API APC_SetHuffmanTableData (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, const char *filename, bool bLogFile)

  set huffman table data for jpeg encode
- int APC_API APC_SetQuantizationTableData (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, const char *filename)

  set quantication table data for jpeg encode

- int APC_API APC_SetPlumAR0330 (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, bool bEnable)
  Set Plum Sensor AR0330.
- int APC_API APC_EnableGPUAcceleration (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, bool enable)
  enable depth filter with GPU acceleration or not
- APC_API char * APC_GetDepthFilterVersion (void *pHandleApcDI, PDEVSELINFO pDevSelInfo)
  get depth filter version

5.1.1 Detailed Description

eYs3D SDK API export functions, data structure and variable definition

Copyright

This file copyright (C) 2017 by eYs3D company

An unpublished work. All rights reserved. This file is proprietary information, and may not be disclosed or copied without the prior permission of eYs3D company.

5.1.2 Typedef Documentation

5.1.2.1 DEVINFORMATION

`typedef struct tagDEVINFORMATION DEVINFORMATION`

DEVINFORMATION.

device information

5.1.2.2 eSPCtrl_RectLogData

`typedef struct eSPCtrl_RectLogData eSPCtrl_RectLogData`

eSPCtrl_RectLogData

Rectified log data structure

5.1.2.3 PARALUT

`typedef struct ParaLUT PARALUT`

ParaLUT.

Spherical look-up table conversion parameters

5.1.3 Enumeration Type Documentation

5.1.3.1 APC_DEVICE_TYPE

`enum APC_DEVICE_TYPE`

chip enum value

Enumerator

OTHERS	Other
AXES1	AXIS1
PUMA	PUMA

5.1.3.2 SENSOR_TYPE_NAME

`enum SENSOR_TYPE_NAME`

Enumerator

APC_SENSOR_TYPE_H22	H22
APC_SENSOR_TYPE_OV7740	OV7740
APC_SENSOR_TYPE_AR0134	AR0134
APC_SENSOR_TYPE_AR0135	AR0135
APC_SENSOR_TYPE_AR0144	AR0144
APC_SENSOR_TYPE_OV9714	OV9714
APC_SENSOR_TYPE_OV9282	OV9282
APC_SENSOR_TYPE_AR0330	AR0330
APC_SENSOR_TYPE_AR1335	AR1335

5.1.3.3 USERDATA_SECTION_INDEX

`enum USERDATA_SECTION_INDEX`

Enumerator

USERDATA_SECTION_0	Section 0
USERDATA_SECTION_1	Section 1
USERDATA_SECTION_2	Section 2
USERDATA_SECTION_3	Section 3
USERDATA_SECTION_4	Section 4
USERDATA_SECTION_5	Section 5
USERDATA_SECTION_6	Section 6
USERDATA_SECTION_7	Section 7
USERDATA_SECTION_8	Section 8
USERDATA_SECTION_9	Section 9
USERDATA_SECTION_10	Section 10
USERDATA_SECTION_NUM	Total Section Number

5.1.4 Function Documentation

5.1.4.1 APC_CloseDevice()

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

close device and stop video render

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index

Returns

success:APC_OK, others:see eSPDI_ErrCode.h

5.1.4.2 APC_DisableAE()

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

disable auto exposure function of ISP

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.3 APC_DisableAWB()

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

disable auto white balance of ISP

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.4 APC_EnableAE()

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

enable auto exposure function of ISP

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.5 APC_EnableAWB()

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

enable auto white balance function of ISP

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.6 APC_EnableGPUAcceleration()

```
int APC_API APC_EnableGPUAcceleration (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            bool enable )
```

enable depth filter with GPU acceleration or not

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
enable	true:enable, fales:diable

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.7 APC_FindDevice()

```
int APC_FindDevice (
            void * pHandleApcDI )
```

find out all eYs3D USB devices by PID, VID and ChipID, also remember device types

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.8 APC_GetCTPropVal()

```
int APC_GetCTPropVal (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            int nId,
            int * pValue )
```

set control terminal property value https://msdn.microsoft.com/en-us/library/windows/hardware/ff567885).aspx The PROPSETID_VIDCAP_CAMERACONTROL property set controls camera device settings. The controls it provides are a subset of the ITU T.RDC standard.

The KSPROPERTY_VIDCAP_CAMERACONTROL enumeration in Ksmedia.h specifies the properties of this set.

Support for this property set is optional and should be implemented only by minidrivers of devices that offer camera control settings. For more information, see the ITU website.

Prior to USB video class, this enumeration contained the following properties: KSPROPERTY_CAMERACONTROL↩ _EXPOSURE KSPROPERTY_CAMERACONTROL_FOCUS KSPROPERTY_CAMERACONTROL_IRIS KSPROPERTY↩ _CAMERACONTROL_ZOOM KSPROPERTY_CAMERACONTROL_PAN KSPROPERTY_CAMERACONTROL↩ _ROLL KSPROPERTY_CAMERACONTROL_TILT

https://msdn.microsoft.com/en-us/library/windows/hardware/ff566089(v=vs.↩ 85).aspx

Parameters

∗pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
nId	specifies the member of the property set
pValue	pointer of store CT property value

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.9 APC_GetCurrentIRValue()

```
int APC_GetCurrentIRValue (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            WORD * pwType )
```

get current infrared radiation(IR) value

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
pwType	value of current IR

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.10 APC_GetDepthFilterVersion()

```
char *APC_API APC_GetDepthFilterVersion (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo )
```

get depth filter version

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index

Returns

success: get version string, others: get N/A string

5.1.4.11 APC_GetDeviceNumber()

```
int APC_GetDeviceNumber (
            void * pHandleApcDI )
```

get eYs3D USB device numbers

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance

Returns

number of eYs3D device

5.1.4.12 APC_GetDeviceResolutionList()

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

get the device resolution list

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
nMaxCount0	max count of endpoint1 resolutions
pStreamInfo0	resolution infos of endpoint1
nMaxCount1	max count of endpoint2 resolutions
pStreamInfo1	resolutions infos of endpoint2

Returns

success: nCount0∗256+nCount1, others: see eSPDI_ErrCode.h

5.1.4.13 APC_GetFlexibleGyroData()

```
int APC_GetFlexibleGyroData (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            int length,
            BYTE * pGyroData )
```

get IMU(Gyro) data

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
length	length of IMU data to read, should be get from APC_GetFlexibleGyroLength
pGyroData	data buffer to store IMU data

5.1.4.14 APC_GetFlexibleGyroLength()

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

get the IMU(Gyro) data length

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
GyroLen	pointer to store IMU data length

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.15 APC_GetFWRegister()

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

get firmware register value

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
address	register address
pValue	pointer of value got from register address
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_ErrCode.h

5.1.4.16 APC_GetFwVersion()

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

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

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
pszFwVersion	firmware version string
nBufferSize	input buffer length to receive FW version
pActualLength	the actual length of FW version in byte

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.17 APC_GetGPIOValue()

```
int APC_GetGPIOValue (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            int nGPIOIndex,
            BYTE * pValue )
```

get general purpose IO value

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
nGPIOIndex	GPIO index, 1 or 2 is valid
pValue	pointer of GPIO value

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.18 APC_GetHWRegister()

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

get hardware register value

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
address	register address
pValue	pointer of value got from register address
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_ErrCode.h

5.1.4.19 APC_GetIRMaxValue()

```
int APC_GetIRMaxValue (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            WORD * pwType )
```

get maximum IR value the module support

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
pwType	pointer strors maximum IR value

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.20 APC_GetIRMinValue()

```
int APC_GetIRMinValue (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            WORD * pwType )
```

get minimum IR value the module support

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
pwType	pointer strors minimum IR value

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.21 APC_GetLogData()

```
int APC_GetLogData (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            BYTE * buffer,
            int BufferLength,
            int * pActualLength,
            int index )
```

get log data from flash

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
buffer	buffer to store log data
BufferLength	input buffer length
pActualLength	actual length has written to buffer
index	index to identify log data for corresponding depth

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.22 APC_GetPidVid()

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

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

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
pPidBuf	4 byte buffer to store PID value
pVidBuf	4 byte buffer to store VID value

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.23 APC_GetPUPropVal()

```
int APC_GetPUPropVal (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            int nId,
            int * pValue )
```

get processing unit property value https://msdn.microsoft.com/en-us/library/windows/hardware/ff56812
85).aspx The PROPSETID_VIDCAP_VIDEOPROCAMP property set controls devices that can adjust image color attributes of analog or digital signals.

The KSPROPERTY_VIDCAP_VIDEOPROCAMP enumeration in ksmedia.h specifies the properties of this set.

Support for this property set is optional and should be implemented only by devices that allow adjustment of brightness, contrast, hue, and other image quality settings.

Prior to USB video class, this enumeration contained the following property items: KSPROPERTY_↩
VIDEOPROCAMP_BACKLIGHT_COMPENSATION KSPROPERTY_VIDEOPROCAMP_BRIGHTNESS KSPROPERTY↩
_VIDEOPROCAMP_COLORENABLE KSPROPERTY_VIDEOPROCAMP_CONTRAST KSPROPERTY_↩
VIDEOPROCAMP_GAMMA KSPROPERTY_VIDEOPROCAMP_HUE KSPROPERTY_VIDEOPROCAMP_↩
SATURATION KSPROPERTY_VIDEOPROCAMP_SHARPNESS KSPROPERTY_VIDEOPROCAMP_WHITEBALANCE KSPROPERTY_VIDEOPROCAMP_GAIN

https://msdn.microsoft.com/en-us/library/windows/hardware/ff566089(v=vs.↩
85).aspx The KSPROPERTY_VIDEOPROCAMP_S structure describes filter-based property settings in the PROPSETID_VIDCAP_VIDEOPROCAMP property set.

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
nId	specifies the member of the property set
pValue	pointer of store PU property value

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.24 APC_GetSensorRegister()

```
int APC_GetSensorRegister (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            int nId,
            unsigned short address,
            unsigned short * pValue,
            int flag,
            int nSensorMode )
```

get value from sensor register

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
nId	sensor slave address. see SENSOR_TYPE_NAME enum definition
address	register address
pValue	pointer of value got from register address
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
nSensorMode	sensor mode(sensor A, B or Both) A is 0, B is 1, Both is 2

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.25 APC_GetSlaveHWRegister()

```
int APC_GetSlaveHWRegister (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            unsigned short address,
            unsigned short * pValue,
            int flag )
```

get hardware register value

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
address	register address
pValue	pointer of value got from register address
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_ErrCode.h

5.1.4.26 APC_GetSlaveLogData()

```
int APC_GetSlaveLogData (
            void * pHandleApcDI,
```

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

get log data from flash

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
buffer	buffer to store log data
BufferLength	input buffer length
pActualLength	actual length has written to buffer
index	index to identify log data for corresponding depth

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.27 APC_GetSlaveSensorRegister()

```
int APC_GetSlaveSensorRegister (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            int nId,
            unsigned short address,
            unsigned short * pValue,
            int flag,
            int nSensorMode )
```

get value from sensor register

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
nId	sensor slave address. see SENSOR_TYPE_NAME enum definition
address	register address
pValue	pointer of value got from register address
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
nSensorMode	sensor mode(sensor A, B or Both) A is 0, B is 1, Both is 2

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.28 APC_Init()

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

entry point of eYs3D camera SDK. This API allocates resource and find all the eSPI camera devices connected to the system.

Parameters

ppHandleApcDI	a pointer of pointer to receive ApcDI SDK instance
bIsLogEnabled	set to true to generate log file, named log.txt in current folder

Returns

success: none negative integer to indicate numbers of devices found in the system.

5.1.4.29 APC_Init2()

```
int APC_Init2 (
            void ** ppHandleApcDI,
            bool bIsLogEnabled,
            bool bEnableAutoRestart )
```

entry point of eYs3D camera SDK. This API allocates resource and find all the eSPI camera devices connected to the system.

Parameters

ppHandleApcDI	a pointer of pointer to receive ApcDI SDK instance
bIsLogEnabled	set to true to generate log file, named log.txt in current folder
bEnableAutoRestart	set true to auto-restart the device if the device was detached and attached again.

Returns

success: none negative integer to indicate numbers of devices found in the system.

Note

Calls APC_Init or APC_Init2 to initilize the ApcDI SDK. APC_Init2 adds the auto-restart function to the initilization options. If you call APC_Init, the bEnableAutoRestart is set as disabled.

5.1.4.30 APC_Is360Device()

```
bool APC_Is360Device (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo )
```

check module is spherical device or not

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index

Returns

true: module support 360, false: not support

5.1.4.31 APC_OpenDevice()

```
int APC_OpenDevice (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            int colorStreamIndex,
            int depthStreamIndex,
            int depthStreamSwitch,
            int iFps,
            APC_ImgCallbackFn callbackFn,
            void * pCallbackParam,
            int pid = -1 )
```

open camera device with image callback support

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
colorStreamIndex	index of the desired color stream
depthStreamIndex	index of the desired sdepth tream
depthStreamSwitch	depth switch for S0, S1 or S2
iFps	pointer to the desired frame rate, returns the actual frame rate.
callbackFn	set image callback function
pCallbackParam	the data to associate with the callback function
pid	Specify device pid.

Table 5.34 Image Control Mode

Mode	Description
0x01	color and depth frame output synchrously, for depth map module only
0x02	enable post-process, for Depth Map module only
0x04	stitch images if this bit is set, for fisheye spherical module only
0x08	use OpenCL in stitching. This bit effective only when bit-2 is set.

Returns

success:APC_OK, others:see eSPDI_ErrCode.h

5.1.4.32 APC_ReadFlashData()

```
int APC_ReadFlashData (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            FLASH_DATA_TYPE fdt,
            BYTE * pBuffer,
            unsigned long int nLengthOfBuffer,
            unsigned long int * pActualBufferLen )
```

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

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
fdt	segment type of flash be read
pBuffer	buffer to store firmware code
nLengthOfBuffer	input buffer length
pActualBufferLen	actual length has written to pBuffer

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.33 APC_RefreshDevice()

```
int APC_RefreshDevice (
            void * pHandleApcDI )
```

refresh all eYs3D UVC devices

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.34 APC_RegisterDeviceEvents()

```
int APC_RegisterDeviceEvents (
            void * pHandleApcDI,
            APC_DeviceEventFn cbFunc,
            void * pData )
```

Register the USB device plug or unplug events. Any USB capture device attachment or detachment events will call the callback function cbFunc.

Parameters

pHandleApcDI	a pointer to ApcDI SDK instance
cbFunc	a callback function of type #APC_DeviceEventFn that will receive USB cappure device events when the device is attached or detached.
pData	user defined data which will send to the callback function

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.35 APC_Release()

```
void APC_Release (
            void ** ppHandleApcDI )
```

release all resource that APC_Init had allocated

Parameters

ppHandleApcDI	pointer of the pointer to the initilized ApcDI SDK instance.

Returns

none

Note

the pointer to ppHandleApcDI will be set to NULL when this call returns successfully.

5.1.4.36 APC_SetCTPropVal()

```
int APC_SetCTPropVal (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
```

```
            int nId,
            int nValue )
```

get control terminal property value https://msdn.microsoft.com/en-us/library/windows/hardware/ff567885).aspx https://msdn.microsoft.com/en-us/library/windows/hardware/ff566089(v=vs.↩85).aspx

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
nId	specifies the member of the property set
nValue	CT property value to set

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.37 APC_SetCurrentIRValue()

```
int APC_SetCurrentIRValue (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            WORD wType )
```

set current infrared radiation(IR) value

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
wType	value to set

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.38 APC_SetFWRegister()

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

set firmware register value

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
address	register address
nValue	register value to set
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_ErrCode.h

5.1.4.39 APC_SetGPIOCtrl()

```
int APC_SetGPIOCtrl (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            int nGPIOIndex,
            BYTE nValue )
```

set GPIO control address

Parameters

nGPIOIndex	index of GPIO (1 ~ 4)
nValue	register value to set

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.40 APC_SetGPIOValue()

```
int APC_SetGPIOValue (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            int nGPIOIndex,
            BYTE nValue )
```

set GPIO value

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
nGPIOIndex	GPIO index, 1 or 2 is valid
nValue	GPIO value to set

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.41 APC_SetHuffmanTableData()

```
int APC_SetHuffmanTableData (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            const char * filename,
            bool bLogFile )
```

set huffman table data for jpeg encode

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
filename	huffman table file, see jh_vga_422.dat sample file
bLogFile	if true then puma_htable.dat file is generated

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.42 APC_SetHWRegister()

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

set hardware register

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
address	register address
nValue	register value to set
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_ErrCode.h

5.1.4.43 APC_SetIRMaxValue()

```
int APC_SetIRMaxValue (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            WORD wType )
```

set maximum IR value the module support

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
wType	pointer strors maximum IR value

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.44 APC_SetLogData()

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

set log data to flash

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
buffer	buffer to store log data
BufferLength	input buffer length
pActualLength	actual length has written to buffer
index	index to identify log data for corresponding depth

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.45 APC_SetPidVid()

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

set PID and VID to device

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
pPidBuf	4 byte PID value buffer to set
pVidBuf	4 byte VID value buffer to set

Returns

success: EtronDI_OK, others: see eSPDI_ErrCode.h

5.1.4.46 APC_SetPUPropVal()

```
int APC_SetPUPropVal (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            int nId,
            int nValue )
```

get processing unit property value https://msdn.microsoft.com/en-us/library/windows/hardware/ff5681
85).aspx https://msdn.microsoft.com/en-us/library/windows/hardware/ff566089(v=vs.↩
85).aspx

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
nId	specifies the member of the property set
nValue	value to set

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.47 APC_SetQuantizationTableData()

```
int APC_SetQuantizationTableData (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            const char * filename )
```

set quantication table data for jpeg encode

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
filename	quantization table file, see FS_DEF_010.txt sample file

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.48 APC_SetSensorTypeName()

```
int APC_SetSensorTypeName (
            void * pHandleApcDI,
            SENSOR_TYPE_NAME stn )
```

select which sensor to operate

Parameters

pHandleApcDI	CApcDI handler
stn	sensor type

Returns

APC_OK

5.1.4.49 APC_SetSlaveHWRegister()

```
int APC_SetSlaveHWRegister (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            unsigned short address,
            unsigned short nValue,
            int flag )
```

set hardware register

Parameters

pHandleApcDI	CApcDI handler
pDevSelInfo	pointer of device select index
address	register address
nValue	register value to set
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_ErrCode.h

5.1.4.50 APC_SetSlaveLogData()

```
int APC_SetSlaveLogData (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            BYTE * buffer,
            int BufferLength,
            int * pActualLength,
            int index )
```

set log data to flash

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
buffer	buffer to store log data
BufferLength	input buffer length
pActualLength	actual length has written to buffer
index	index to identify log data for corresponding depth

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.1.4.51 APC_SetUserData()

```
int APC_SetUserData (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            BYTE * buffer,
            int BufferLength,
            USERDATA_SECTION_INDEX usi )
```

set user data to flash

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
buffer	buffer to store user data
BufferLength	input buffer length
usi	which user index data to select

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.2 eSPDI_DM.h File Reference

eYs3D SDK API export functions, data structure and variable definition for depth map module

Functions

- int APC_API APC_GetSlaveYOffset (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, int *pActualLength, int index)

  get Y offset data
- int APC_API APC_GetYOffset (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int Buffer↩ Length, int *pActualLength, int index)

  get Y offset data
- int APC_API APC_GetSlaveRectifyTable (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, int *pActualLength, int index)

  get rectify values from flash
- int APC_API APC_GetRectifyTable (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, int *pActualLength, int index)

  get rectify values from flash
- int APC_API APC_GetZDTable (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, int *pActualLength, PZDTABLEINFO pZDTableInfo)

get disparity and Z values from flash

- int APC_API APC_SetSlaveYOffset (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, int *pActualLength, int index)

  set Y offset data
- int APC_API APC_SetYOffset (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, int *pActualLength, int index)

  set Y offset data
- int APC_API APC_SetSlaveRectifyTable (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, int *pActualLength, int index)

  set rectify data to flash, see APC_SetRectifyTable except set
- int APC_API APC_SetRectifyTable (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, int *pActualLength, int index)

  set rectify data to flash, see APC_SetRectifyTable except set
- int APC_API APC_SetZDTable (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, BYTE *buffer, int BufferLength, int *pActualLength, PZDTABLEINFO pZDTableInfo)

  set disparity and Z values to flash, see APC_GetZDTable except get
- int APC_API APC_GetRectifyMatLogData (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, eSPCtrl_RectLogData *pData, int index)

  get rectify log data from flash for Puma IC
- int APC_API APC_SetDepthDataType (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, WORD wType)

  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_API APC_GetDepthDataType (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, WORD *pwType)

  get current depth data type setting
- int APC_API APC_SetHWPostProcess (void *pHandleApcDI, PDEVSELINFO pDevSelInfo, bool enable)

  enable or disable internal chip post processing function

5.2.1 Detailed Description

eYs3D SDK API export functions, data structure and variable definition for depth map module

Copyright

5.2.2 Function Documentation

5.2.2.1 APC_GetDepthDataType()

```
int APC_GetDepthDataType (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            WORD * pwType )
```

get current depth data type setting

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
pwType	pointer of current depth data type in device

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.2.2.2 APC_GetRectifyMatLogData()

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

get rectify log data from flash for Puma IC

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
pData	rectify log data, its buffer size is 4096 bytes see eSPCtrl_RectLogData for detailed members
index	index to identify rectify log data for corresponding depth

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.2.2.3 APC_GetRectifyTable()

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

get rectify values from flash

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
buffer	buffer to store rectify table data
BufferLength	input buffer length
pActualLength	actual length has written to buffer
index	index to identify rectify table for corresponding depth

Returns

success:APC_OK, others: see eSPDI_ErrCode.h

5.2.2.4 APC_GetSlaveRectifyTable()

```
APC_GetSlaveRectifyTable (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            BYTE * buffer,
            int BufferLength,
            int * pActualLength,
            int index )
```

get rectify values from flash

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
buffer	buffer to store rectify table data
BufferLength	input buffer length
pActualLength	actual length has written to buffer
index	index to identify rectify table for corresponding depth

Returns

success:APC_OK, others: see eSPDI_ErrCode.h

5.2.2.5 APC_GetSlaveYOffset()

```
int APC_GetSlaveYOffset (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            BYTE * buffer,
```

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

get Y offset data

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
buffer	buffer to store
BufferLength	length of buffer
pActualLength	actual byte of reading
index	index of Y offset file ID

Returns

success:APC_OK, others:see eSPDI_ErrCode.h

5.2.2.6 APC_GetYOffset()

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

get Y offset data

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
buffer	buffer to store
BufferLength	length of buffer
pActualLength	actual byte of reading
index	index of Y offset file ID

Returns

success:APC_OK, others:see eSPDI_ErrCode.h

5.2.2.7 APC_GetZDTable()

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

get disparity and Z values from flash

1. if depth data type is APC_DEPTH_DATA_14_BITS then just get Z value from depth buffer
2. if depth data type is APC_ZD_TABLE_FILE_SIZE_11_BITS then using depth buffer value as a index to get Z value inside ZD table
3. see GetZValue() of example.c to get Z value from different depth data type

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
buffer	bufer to store ZD table
BufferLength	input buffer length
pActualLength	actual length has written to buffer
pZDTableInfo	index to identify ZD table and data type for corrresponding depth

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.2.2.8 APC_SetDepthDataType()

```
APC_SetDepthDataType (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            WORD wType )
```

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

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
wType	depth data type you want to set, see APC_DEPTH_DATA_xxx in APC_O.h \output success: APC_OK, others: see eSPDI_ErrCode.h

5.2.2.9 APC_SetHWPostProcess()

```
int APC_SetHWPostProcess (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            bool enable )
```

enable or disable internal chip post processing function

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
enable	set true to enable post-process, or set false to disable post-process

Returns

success: APC_OK, others: see eSPDI_ErrCode.h

5.2.2.10 APC_SetSlaveYOffset()

```
int APC_SetSlaveYOffset (
            void * pHandleApcDI,
            PDEVSELINFO pDevSelInfo,
            BYTE * buffer,
            int BufferLength,
            int * pActualLength,
            int index )
```

set Y offset data

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
buffer	buffer to store
BufferLength	length of buffer
pActualLength	actual byte of reading
index	index of Y offset file ID

Returns

success:APC_OK, others:see eSPDI_ErrCode.h

5.2.2.11 APC_SetYOffset()

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

set Y offset data

Parameters

pHandleApcDI	the pointer to the initilized ApcDI SDK instance
pDevSelInfo	pointer of device select index
buffer	buffer to store
BufferLength	length of buffer
pActualLength	actual byte of reading
index	index of Y offset file ID

Returns

success:APC_OK, others:see eSPDI_ErrCode.h

5.3 eSPDI_ErrCode.h File Reference

definition of eYs3D SDK error code Copyright: This file copyright (C) 2017 by

5.3.1 Detailed Description

definition of eYs3D SDK error code Copyright: This file copyright (C) 2017 by

eYs3D company

Index