Preface

Objective

This document mainly introduces the DPSDK Defense PC client API interface reference, including programming guide, header file notes, data type definition, callback function definition, event definition, interface function definition and error codes.

Product Version

The product version matching with this document is listed as below:

Product Name	**Version**
DPSDK	V1.0.001

Reader

This document (guide) is mainly suitable for the following objects:

- Defense software development engineers.

Document Notes

User Guide

This document is composed of sections of programming guide, header file notes, data type definition, callback function definition, event definition, interface function definition and error codes.

- Programming guide: gives notes of DPSDK, Defense and API to call main procedures and main modules.
- Header file notes: gives DPSDK, Defense and API header file names and their corresponding notes.
- Data type definition: gives the definitions for frequently-used data types appeared in this document.
- Callback function definition: includes definition for all the callback functions of DPSDK, Defense and API.
- Event definition: gives macro definition, macro definition value and returned value.
- Interface function definition: includes detailed notes for all DPSDK, Defense and API interface functions, including description, function, parameter, returned value, etc.
- Error code: Lists the error codes which may appear on interface call.

Parent Subject: Document Notes

Programming Guide

Main Process Flow of Port Call

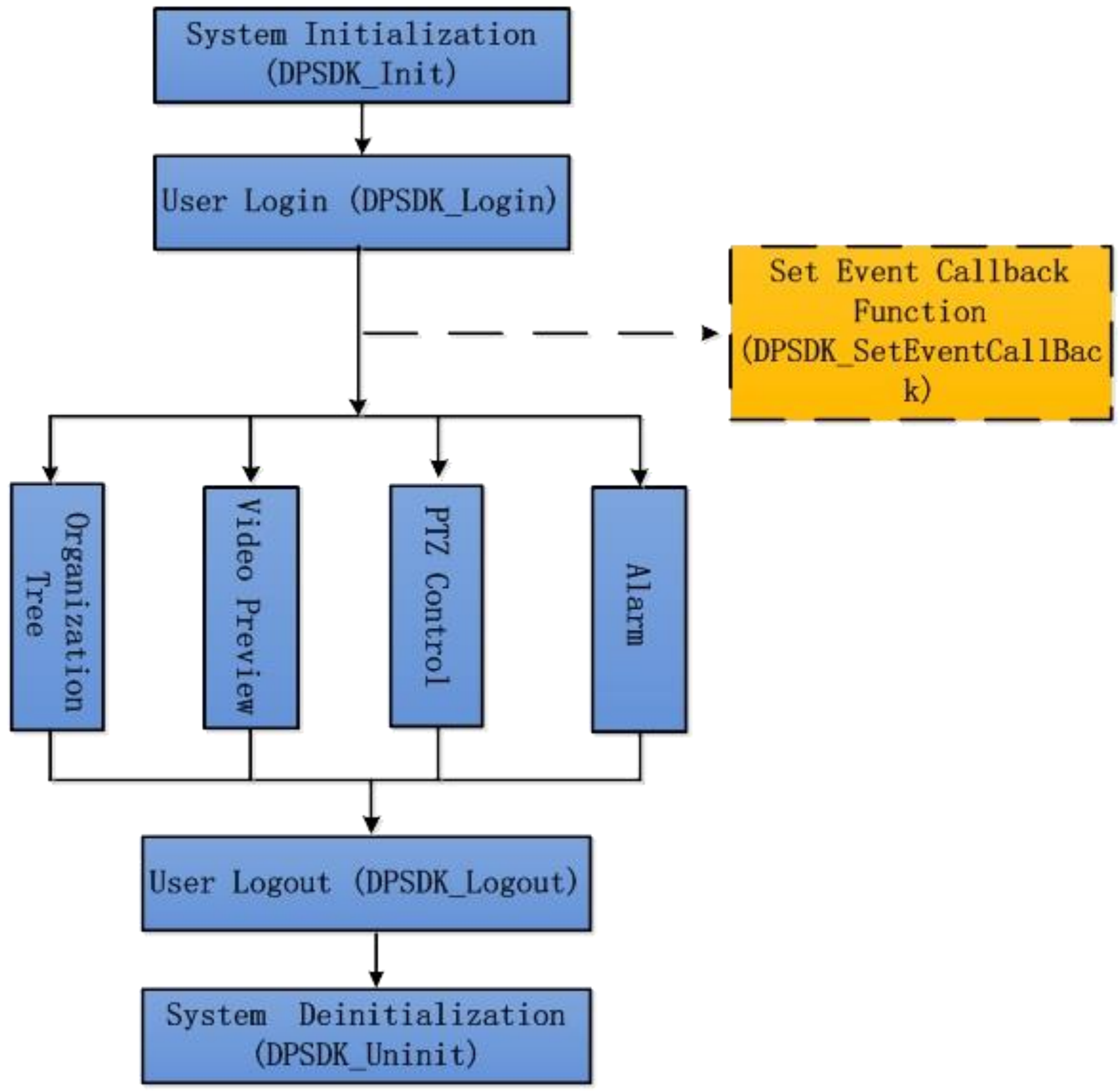

System Initialization (DPSDK_Init)
User Login (DPSDK_Login)
Set Event Callback Function (DPSDK_SetEventCallBack)
Organization Tree
Video Preview
PTZ Control
Alarm
User Logout (DPSDK_Logout)
System Deinitialization (DPSDK_Uninit)

Fig. 1 Main Process Flow of Port Call

Note:
Flow in the dashed box is optional, and doesn’t affect functional use of other flows. Before using every function, it is necessary to carry out **system initialization, user login, user logout and system deinitialization**. Other port call flows are not described herein.

- **System initialization (DPSDK_Init):** Call this function port to initialize the whole SDK system and pre-allocate the memory and so on.
- **User login (DPSDK_Login):** The user logs in server. After successful login, the returned session ID serves as the unique identification of other functional operations.
- **Set event callback function (DPSDK_SetEventCallBack):** Event callback function is used to receive server event notices and SDK events, such as alarm event, increasing organization and modifying device. After initialization and logging in SDK, the user can set this callback function, receive and deal with messages at application layer. This port is not called if it doesn’t need to receive and deal with events.

- **Organization tree:** Other operations are available after loading organization tree info. Organization device info can be loaded all at once, or organization, device and channel info can be loaded and obtained according to different levels. Please refer to Organization Tree Flow for details.
- **Video preview:** Obtain real-time stream of the channel, decode and display it. Please refer to Video Preview Flow for details.
- **PTZ control:** Control PTZ camera direction and lens. Please refer to PTZ Control Flow for details.
- **Alarm:** Through alarm event callback (please refer to EventCallBack for details), obtain real-time alarm data; through alarm query port (please refer to DPSDK_QueryAlarm), obtain historical alarm data. Please refer to Alarm Flowfor details.

Parent subject: Programming Guide

-

Organization Tree Flow

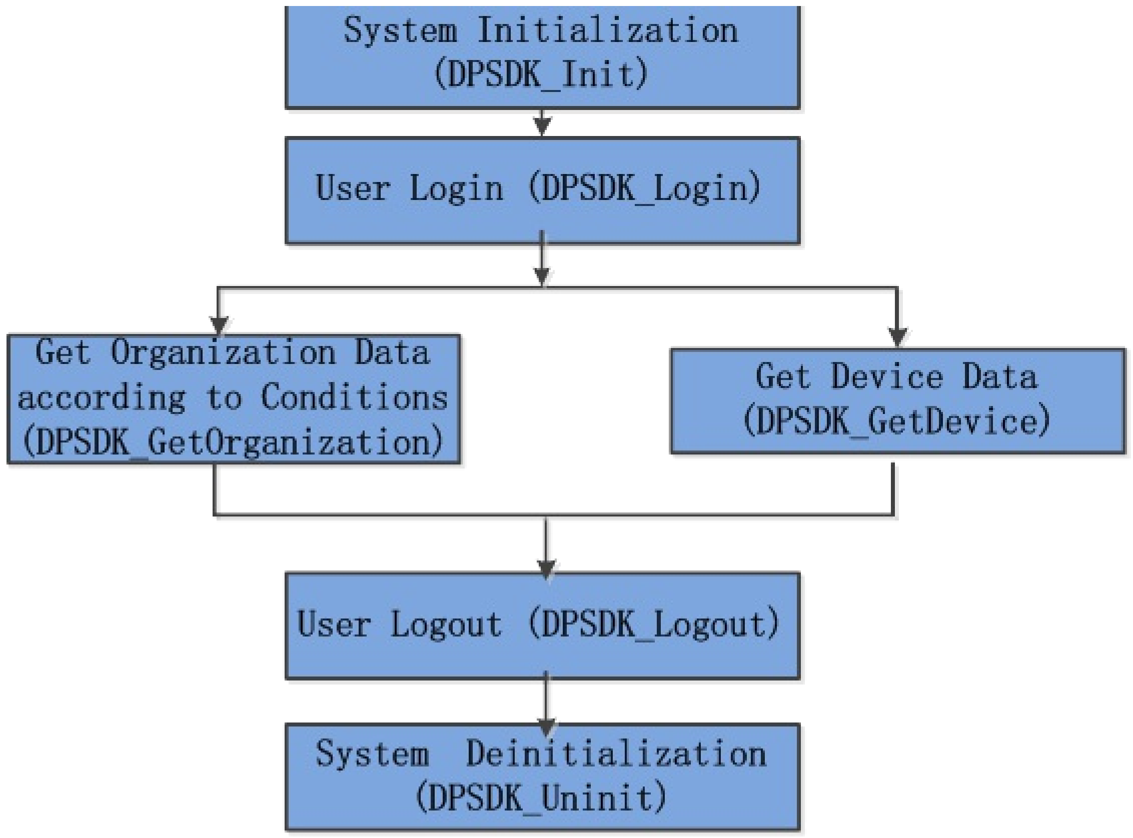

System Initialization (DPSDK_Init)
User Login (DPSDK_Login)
Get Organization Data according to Conditions (DPSDK_GetOrganization)
Get Device Data (DPSDK_GetDevice)
User Logout (DPSDK_Logout)
System Deinitialization (DPSDK_Uninit)

Fig. 1 Organization Tree Flow

Get organization data according to conditions (DPSDK_GetOrganization**):** Call this function port to get organization data according to query conditions (organization node, child node type and channel type set etc.).

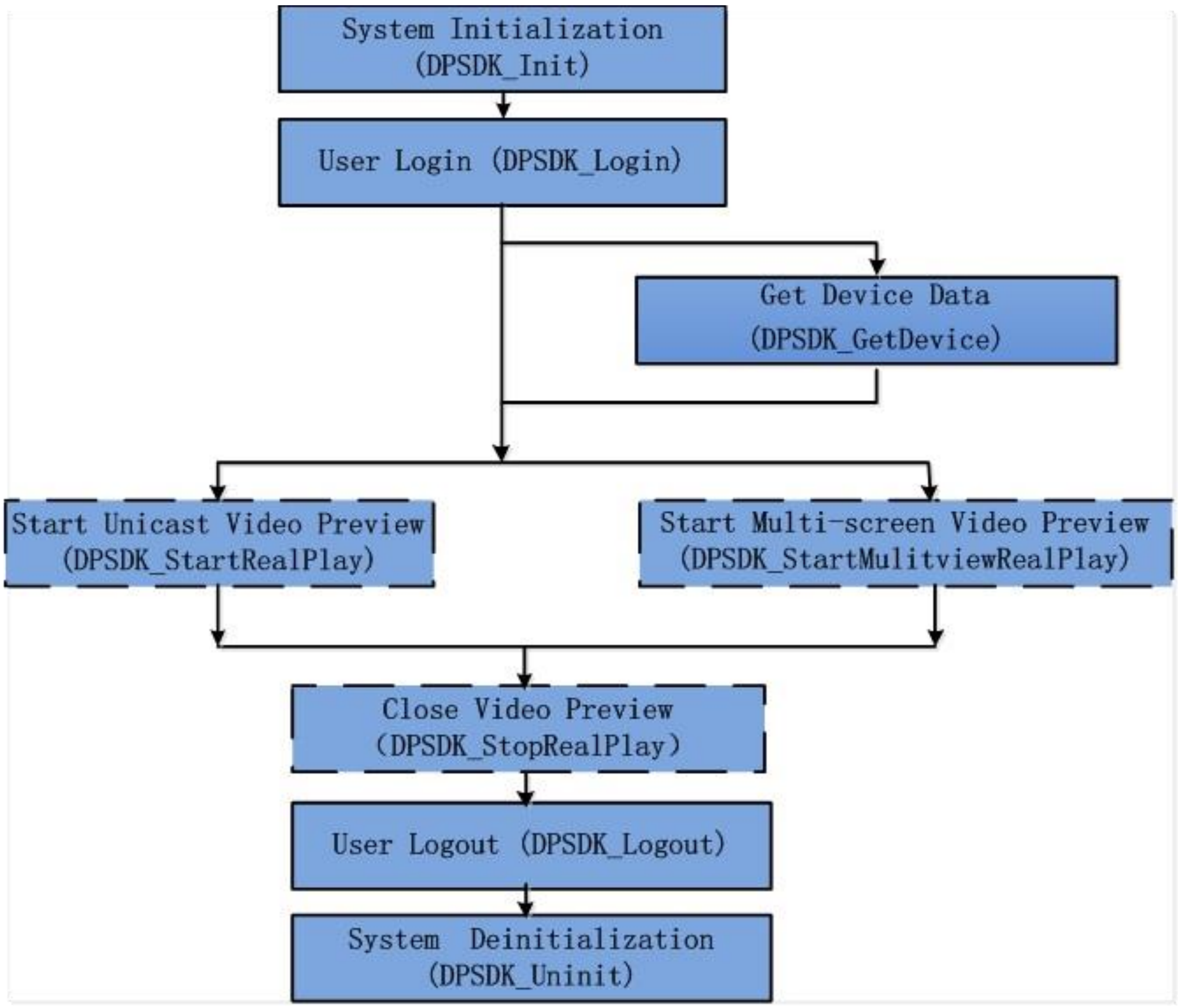
Get device data (DPSDK_GetDevice**):** Call this function port to get device data according to device ID list.

Parent Subject: Programming Guide

-

Video Preview Flow

Fig. 1 Video Preview Flow


System Initialization (DPSDK_Init)
User Login (DPSDK_Login)
Get Device Data (DPSDK_GetDevice)
Start Unicast Video Preview (DPSDK_StartRealPlay)
Start Multi-screen Video Preview (DPSDK_StartMulitviewRealPlay)
Close Video Preview (DPSDK_StopRealPlay)
User Logout (DPSDK_Logout)
System Deinitialization (DPSDK_Uninit)

Get real-time stream of the channel, decode and display it.

Get device data (DPSDK_GetDevice**):** Before real-time preview, call this port to get the latest channel list from the server. Business port is independent of this port. If the user has gotten channel info in other ways, call real-time preview port directly, without need to call device query port.

Start unicast video preview (DPSDK_StartRealPlay**):** Call this function port to realize real-time unicast video preview function. The user can select a channel from channel list, and realize real-time preview. Dashed box is relevant with real-time preview; Snapshot is available only after real-time preview or record playback has been enabled. Generally, real-time preview shall be started during PTZ control (PTZ control port itself is independent of real-time preview). Local recording can be called after real-time preview is enabled; sound control can control the opening, closing and volume of channel-associated sound when real-time preview is enabled.

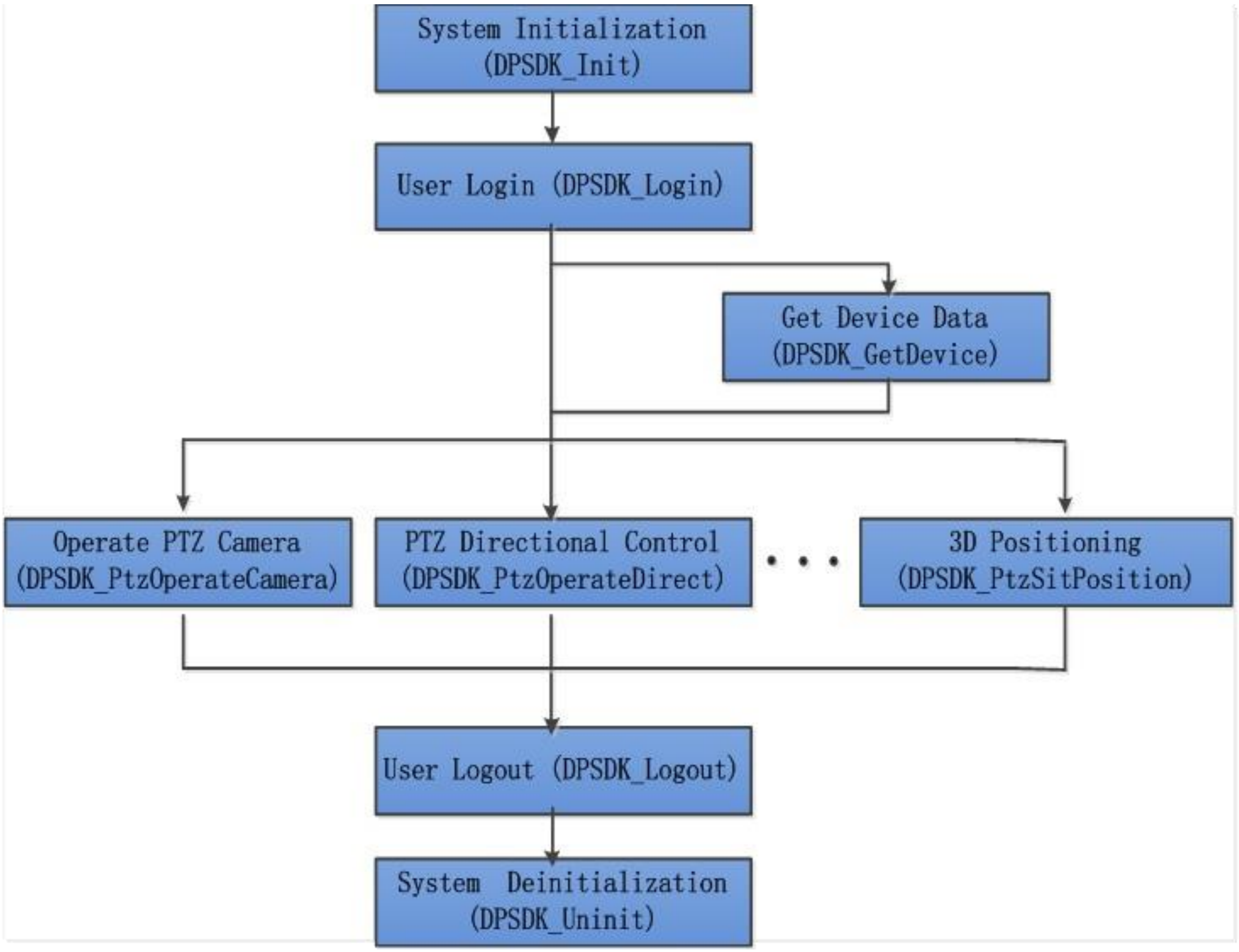
Start multi-screen video preview (DPSDK_StartMulitviewRealPlay**):** Call this function port to realize real-time preview function of multi-screen video. The user can select a channel from channel list, and realize real-time preview.

Close video preview (DPSDK_StopRealPlay): Call this function port to close video preview.

-

PTZ Control Flow

Fig. 1 PTZ Control Flow


System Initialization (DPSDK_Init)
User Login (DPSDK_Login)
Get Device Data (DPSDK_GetDevice)
Operate PTZ Camera (DPSDK_PtzOperateCamera)
PTZ Directional Control (DPSDK_PtzOperateDirect)
• • •
3D Positioning (DPSDK_PtzSitPosition)
User Logout (DPSDK_Logout)
System Deinitialization (DPSDK_Uninit)

Open real-time video preview (PTZ control port itself is independent of video preview) before PTZ control. Carry out PTZ control on basis of real-time video preview.

PTZ functional operation (DPSDK_PtzOperateFunction**):** Call this function port to realize PTZ function operation. For example, display PTZ menu, control PTZ menu direction, enable and disable line scanning, light and wiper.

PTZ directional control (DPSDK_PtzOperateDirect**):** Call this function port to control PTZ rotation upwards, downwards, left and right.

Operate PTZ camera (DPSDK_PtzOperateCamera**):** Call this function port to operate PTZ camera (zoom, focus and iris).

Electrical focus control (DPSDK_PtzOperateFocus**):** Call this function port to realize PTZ continuous focus or auto focus function.

PTZ 3D positioning (DPSDK_PtzSitPosition**):** Call this function port to realize 3D positioning, including ordinary 3D positioning, precise 3D positioning and radar PTZ 3D positioning.

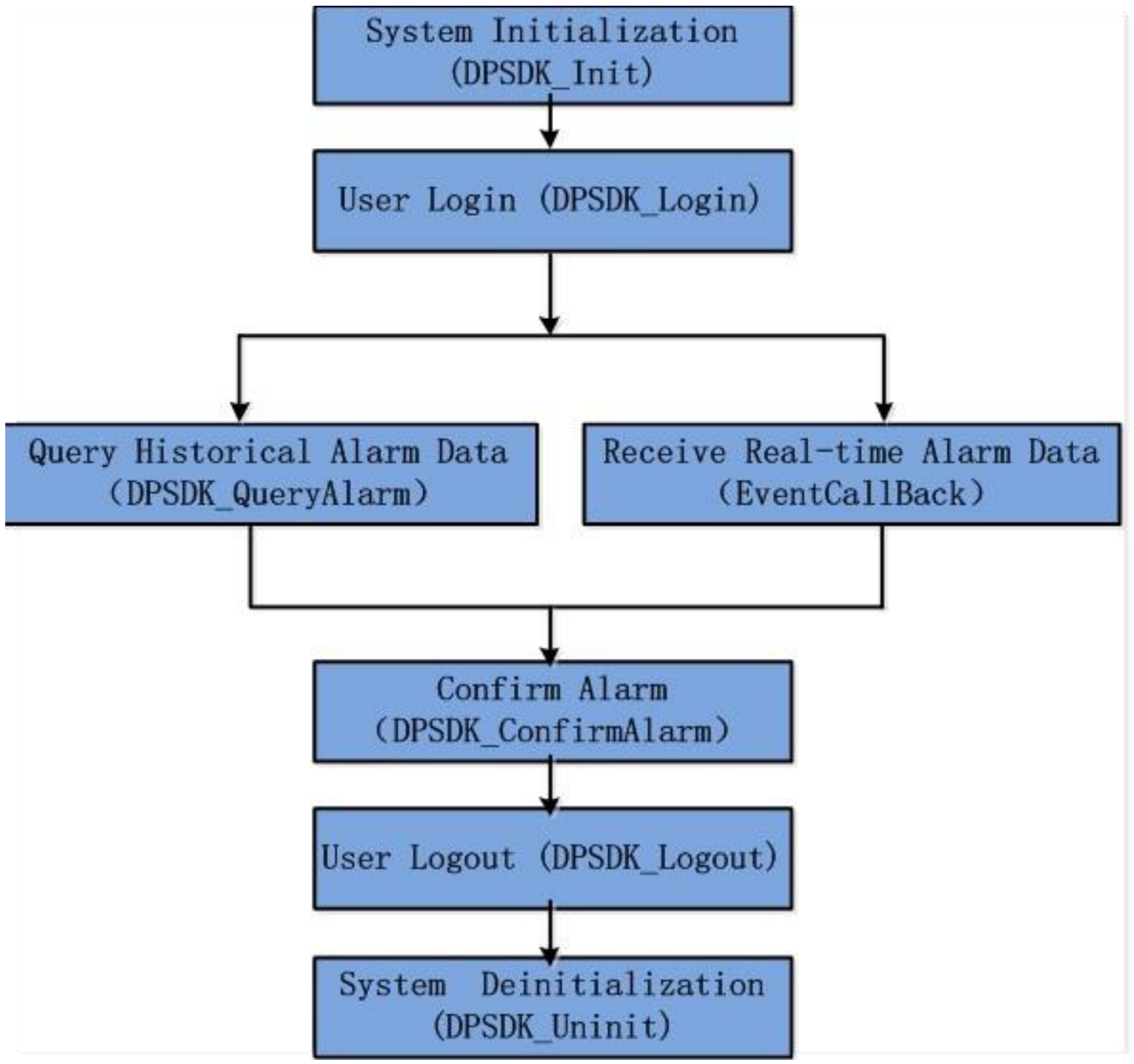
Control preset point (DPSDK_PtzOperatePresetPoint): Call this function port to position, set, delete and update PTZ preset point.

Lock and unlock PTZ (DPSDK_PtzArrangePtz): Call this function port to lock and unlock PTZ.

Parent Subject: Programming Guide

Alarm Flow

Fig. 1 Alarm Flow


System Initialization (DPSDK_Init)
User Login (DPSDK_Login)
Query Historical Alarm Data (DPSDK_QueryAlarm)
Receive Real-time Alarm Data (EventCallBack)
Confirm Alarm (DPSDK_ConfirmAlarm)
User Logout (DPSDK_Logout)
System Deinitialization (DPSDK_Uninit)

Get alarm info in two ways: Receive alarm event and query the alarm.

Receive real-time alarm data: Get real-time alarm event data according to event callback function (please refer to EventCallBack for details).

Query historical alarm data (DPSDK_QueryAlarm): Call the function port to query historical alarm data.

Confirm alarm (DPSDK_ConfirmAlarm): Call the function port to confirm alarm event according to alarm confirmation parameters (alarm code, handler's username and handling opinion etc.).

Parent Subject: Programming Guide

Header File Notes

Header File Name	Note
DPSDK.h	Header file of interface function.
DPSDK_Error.h	Error code definition.
DPSDK_Event.h	Event Definition.
DPSDKDefine.h	Constant, data struct, enum-type.

Data Type Definition

Data Type	DPSDK Data Type
typedef int	DPSDK_INT32;
typedef unsigned int	DPSDK_UINT32;
typedef void*	DPSDK_LPVOID;
typedef long	DPSDK_LONG;
typedef unsigned long	DPSDK_ULONG;
typedef bool	DPSDK_BOOL;
typedef char	DPSDK_CHAR;
typedef unsigned char	DPSDK_UCHAR;
typedef float	DPSDK_FLOAT;
typedef double	DPSDK_DOUBLE;
typedef void	DPSDK_VOID;
typedef short	DPSDK_SHORT;
typedef unsigned short	DPSDK_USHORT;
typedef size_t	DPSDK_SIZET;
typedef long long	DPSDK_TIMET;

Callback Function Definition

Event Callback Function EventCallBack

Name	Note
Description:	Event Callback Function

Function:	typedef DPSDK_VOID (DPSDK_CALL * EventCallBack)(
	    DPSDK_INT32 iEventType,
	    DPSDK_VOID* pEventBuf,
	    DPSDK_UINT32 uiBufSize,
	    DPSDK_VOID* pUserData
	);

Parameters:	iEventType
	   [in] Event Type
	pEventBuf
	  [in] Event Data
	uiBufSize
	  [in] Event Data Length
	pUserData
	  [in] User Data

Returned Value:	None
Samples:	None

Parent Subject: Definition of Callback Function

Media Stream Callback Function DPSDK_REALDATA_CALLBACK

Name	Note
Description:	Media Stream Callback Function
Function:	typedef DPSDK_INT32 (DPSDK_CALL * DPSDK_REALDATA_CALLBACK)(DPSDK_INT32 iMediaType, DPSDK_CHAR* pData, DPSDK_INT32 iDataLen, DPSDK_VOID* pUserParam);
Parameters:	iMediaType [in] Corresponding Business of Media Stream pData [in] Media Stream Data iDataLen [in] Data Length pUserParam [in] User Parameter
Returned Value:	Returned value is 0 in case of success.
Samples:	None

Parent Subject: Definition of Callback Function

Fisheye Data Callback Function DPSDK_FISHEYE_CALLBACK

Name	Note
Description:	Fisheye Data Callback Function
Function:	typedef DPSDK_VOID (DPSDK_CALL * DPSDK_FISHEYE_CALLBACK)(DPSDK_UCHAR uszCorrectMode, DPSDK_USHORT uRadius, DPSDK_USHORT uCircleX, DPSDK_USHORT uCircleY, DPSDK_UINT32 uWidthRatio, DPSDK_UINT32 uHeigthRatio, DPSDK_UCHAR uszGain, DPSDK_UCHAR uszDenoiseLevel, DPSDK_UCHAR uszInstallStyle, DPSDK_LPVOID pUserData);
Parameters:	uszCorrectMode [out] Correction Mode uRadius [out] Radius [0,8192) uCircleX [out] Abscissa of Circle Center uCircleY [out] Ordinate of Circle Center uWidthRatio [out] Width Ratio uHeigthRatio [out] Height Ratio uszGain [out] Gain uszDenoiseLevel [out] Noise Reduction Level uszInstallStyle [out] Fisheye Installation Type pUserData [out] User Data
Returned Value:	None
Samples:	None

Parent Subject: Definition of Callback Function

Video Drawing Callback Function DPSDK_DRAW_CALLBACK

Name	Note
Description:	Video Drawing Callback Function
Function:	typedef DPSDK_VOID (DPSDK_CALL * DPSDK_DRAW_CALLBACK)(DPSDK_HDC hDc, HCWND pWnd, DPSDK_LPVOID pUserData);
Parameters:	hDc [out] Drawing Handle pWnd [out] Window Handle pUserData [out] User Data
Returned Value:	None
Samples:	None

Parent Subject: Definition of Callback Function

Data Callback Function after Source Data Analysis DPSDK_DEMUXDEC_CALLBACK

Name	**Note**
Description:	Data Callback Function after Source Data Analysis
Function:	typedef DPSDK_VOID (DPSDK_CALL * DPSDK_DEMUXDEC_CALLBACK)(DPSDK_LPVOID pUserData, DPSDK_INT32 iEncode);
Parameters:	pUserData [out] User Data iEncode [out] MPEG4, H264, STDH264
Returned Value:	None
Samples:	None

Parent Subject: Definition of Callback Function

Callback Function of Playback on TV Wall DPSDK_TVWALL_PLAYBACK_CALLBACK

Name	**Note**
Description:	Callback Function of Playback on TV Wall

Function:	typedef DPSDK_INT32 (DPSDK_CALL * DPSDK_TVWALL_PLAYBACK_CALLBACK)(
	    DPSDK_CHAR* pData,
	    DPSDK_INT32 iDataLen,
	    DPSDK_VOID* pUserParam
	);

Parameters:	pData
	  [in] Media Stream Data
	iDataLen
	  [in] Data Length
	pUserParam
	    [in] User Parameter

Returned Value:	Returned value is 0 in case of success.
Samples:	None

Parent Subject: Definition of Callback Function

Media Event Callback Function DPSDK_EVENT_CALLBACK

Name	**Note**
Description:	Media Event Callback Function

Function:	typedef DPSDK_VOID (DPSDK_CALL * DPSDK_EVENT_CALLBACK)(
	    DPSDK_INT32 iEventType,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_VOID* pUserParam
	);

Parameters:	iEventType
	  [in] Event Type
	iMediaSessionID
	  [in] Media Session ID
	pUserParam
	  [in] User Parameter

Returned Value:	None
Samples:	None

Parent Subject: Definition of Callback Function

Picture Data Callback Function DPSDK_PICDATA_CALLBACK

Name	Note
Description:	Picture Data Callback Function

Function:	typedef DPSDK_INT32 (DPSDK_CALL * DPSDK_PICDATA_CALLBACK)(
	    DPSDK_INT32 iSession,
	    DPSDK_CHAR* pData,
	    DPSDK_INT32 iDataLen,
	    DPSDK_LPVOID pUserParam,
	    DPSDK_INT32 iPicEventType
	);

Parameters:	iSession
	  [in] Returned Session of Corresponding Request
	pData
	  [in] Picture Stream Data
	iDataLen
	  [in] Data Length
	pUserParam
	  [in] User Parameter
	iPicEventType
	  [in] Picture Event Type

Returned Value:	Returned value is 0 in case of success.
Samples:	None

Parent Subject: Definition of Callback Function

Event Definition

Macro Definition

Macro Definition	Value of Macro Definition	Returned Value
Message Definition (Message Report)		
DPSDK_EVENT_SERVER_OFFLINE	1	N/A
DPSDK_EVENT_RELOGIN_SUCCESS	2	N/A
DPSDK_EVENT_ALARM_ALARMEVENT	3	Returned structure: DPSDK_ALARMEVENT
DPSDK_EVENT_ALARM_CONFIRMALARM	4	Returned structure: DPSDK_ALARMCONFIR
DPSDK_EVENT_ALARM_ALARMPICTURE	5	Returned structure: DPSDK_ALARM_DETAI
DPSDK_EVENT_ALARM_EXPORTALARM	6	Returned structure: DPSDK_ALARMEXPOR
DPSDK_EVENT_DEVICE_STATUS	7	Returned structure: DPSDK_DEV_STATUS_N
DPSDK_EVENT_CHANNEL_STATUS	8	Returned structure: DPSDK_CHANNEL_STA
DPSDK_EVENT_ADD_ORG	9	Returned structure: DPSDK_ORG_BASE_INF
DPSDK_EVENT_MODIFY_ORG	10	Returned structure: DPSDK_ORG_BASE_INF
DPSDK_EVENT_DELETE_ORG	11	DPSDK_CHAR*
DPSDK_EVENT_MOVE_ORG	12	Returned structure: DPSDK_MOVE_ORG_N
DPSDK_EVENT_ADD_DEVICE	13	Returned structure: DPSDK_ADD_DEVICE_
DPSDK_EVENT_MODIFY_DEVICE	14	Returned structure: DPSDK_MODIFY_DEVI
DPSDK_EVENT_DELETE_DEVICE	15	Returned structure: DPSDK_DELETE_DEVI
DPSDK_EVENT_MOVE_DEVICE	16	Returned structure: DPSDK_MOVE_DEVICE
DPSDK_EVENT_ALERT_USER	17	N/A

DPSDK_EVENT_USER_LOCKED	18	N/A
DPSDK_EVENT_USER_PWD_CHANGED	19	N/A
DPSDK_EVENT_USER_OVER_DATA	20	N/A
DPSDK_EVENT_SYNC_TIME	21	N/A
DPSDK_EVENT_USER_UNLOCKED	22	N/A
DPSDK_EVENT_USERDATA_STATE_CHANGE	23	Returned structure: DPSDK_USERONLINES
DPSDK_EVENT_USERDATA_ADD	24	Returned structure: DPSDK_USERADD_NOT
DPSDK_EVENT_USERDATA_DELETE	25	Returned structure: DPSDK_USERDELETE_
DPSDK_EVENT_USER_DELETE	26	N/A
DPSDK_EVENT_VIEW_INFO_CHANGED	27	Returned structure: DPSDK_VIEWINFO_CH
DPSDK_EVENT_DEV_LOCATION_CHANGED	28	Returned structure: DPSDK_DEVICELOCATI
DPSDK_EVENT_LOCK_STATUS_CHANGED	29	Returned structure: DPSDK_LOCKSTATUS_
DPSDK_EVENT_RADER_FRAME_INFO	30	Returned structure: DPSDK_RADERFRAME_
DPSDK_EVENT_FACE_INFO	31	Returned structure: DPSDK_FACE_INFO_NC
DPSDK_EVENT_UPDATE_PERSONTYPE	32	Returned structure: DPSDK_PERSONTYPE_
DPSDK_EVENT_USERDEFINEDATA_ALERT	33	Returned structure: DPSDK_USERDEFINE_I
DPSDK_EVENT_POS_DATA_PUSH	34	Returned structure: DPSDK_POS_DATA_NO
DPSDK_EVENT_ADD_RELATION	35	Returned structure: DPSDK_ADD_RELATIO
DPSDK_EVENT_MODIFY_RELATION	36	Returned structure: DPSDK_MODIFY_RELA
DPSDK_EVENT_DELETE_RELATION	37	Returned structure: DPSDK_DELETE_RELA
DPSDK_EVENT_BITMAP_INFO_CHANGED	38	Returned structure: DPSDK_BITMAP_INFO_
DPSDK_EVENT_UPDATE_LICENSE	39	N/A

DPSDK_EVENT_ADD_TVWALL	50	Returned structure: DPSDK_TVWALL_NOTI
DPSDK_EVENT_MODIFY_TVWALL	51	Returned structure: DPSDK_TVWALL_NOTI
DPSDK_EVENT_DELETE_TVWALL	52	DPSDK_UINT32*
DPSDK_EVENT_MASTER_SLAVE	60	Returned structure: DPSDK_SMARTTRACK
DPSDK_EVENT_MEDIA_FIRST_RECEIVE	100	N/A
DPSDK_EVENT_MEDIA_MTSOFFLINE	101	N/A
DPSDK_EVENT_MEDIA_ENCHANGE	102	Returned structure: DPSDK_EVENT_PARAM
DPSDK_EVENT_MEDIA_SCREENSHOT	103	Returned structure: DPSDK_EVENT_PARAM
DPSDK_EVENT_MEDIA_RECORD_FINISH_FILE	104	Returned structure: DPSDK_EVENT_PARAM
DPSDK_EVENT_MEDIA_RECORD_ABNORMAL	105	Returned structure: DPSDK_EVENT_PARAM
DPSDK_EVENT_PLAYBACK_DATAOVER_RECEIVE	**Definition of General Constants**	112
DPSDK_EVENT_BAYONET_PICINFO_RECIVE		120
DPSDK_EVENT_BAYONET_PICDATA_RECIVE		121
DPSDK_EVENT_BAYONET_PICDATA_OVER	DPSDK_NAME_LEN	122
DPSDK_EVENT_BAYONET_RTP_CLOSE		123
DPSDK_EVENT_BAYONET_SURVEY_ALARM		124

	N/A	LL
64	For details, please refer callback function fo DPSDK_PICDATA_CAL	For details, please refer callback function fo DPSDK_PICDATA_CALL
	For details, please refer callback function for pictur DPSDK_PICDATA_CA	N/A
		Returned structure: DPSDK_CAR_SURVEY_
		N/A

DPSDK_PWD_LEN	64	N/A
DPSDK_IP_LEN	64	N/A
DPSDK_MACADDRESS_LEN	128	N/A
DPSDK_IMEI_LEN	128	N/A
DPSDK_VIDEO_PARAM_LEN	20	N/A
DPSDK_URL_LEN	256	N/A
DPSDK_FILE_PATH_LEN	256	N/A
MASTERSALVE_CLASS_LEN	16	N/A
DPSDK_ALARM_ALARMCODE_LEN	50	N/A
DPSDK_ALARM_HANDLERUSER_LEN	50	N/A
DPSDK_ALARM_HANDLEMESSAGE_LEN	255	N/A
DPSDK_ALARM_EMAILRECEIVER_LEN	320	N/A
DPSDK_ALARM_TIME_LEN	15	N/A
DPSDK_ALARM_DEVICEID_LEN	50	N/A
DPSDK_ALARM_CHANNELID_LEN	100	N/A
DPSDK_ALARM_ORGID_LEN	20	N/A
DPSDK_ALARM_ALARMID_LEN	20	N/A
DPSDK_ALARM_DEVICENAME_LEN	50	N/A
DPSDK_ALARM_CHANNELNAME_LEN	50	N/A
DPSDK_ALARM_ALARMPICTURE_LEN	255*16	N/A
DPSDK_ALARM_GROUPNAME_LEN	50	N/A
DPSDK_ALARM_ALARMSOURCE_LEN	100	N/A
DPSDK_ALARM_ALARMTYPENAME_LEN	255	N/A
DPSDK_ALARM_LANGUAGE_LEN	20	N/A

DPSDK_ALARM_ALARMEXPORTDOWNLOADPATH_LEN	255	N/A
DPSDK_ALARM_NODECODE_LEN	90	N/A
DPSDK_ALARM_ALARMMESSAGE_LEN	255	N/A
DPSDK_ALARM_EMAILRECEIVERLIST_SIZE	10	N/A
DPSDK_ALARM_LINKVEDIOINFOLIST_SIZE	10	N/A
DPSDK_ORG_CODE_LEN	96	N/A
DPSDK_ORG_SN_LEN	56	N/A
DPSDK_DEVICE_ID_LEN	56	N/A
DPSDK_SN_LEN	56	N/A
DPSDK_CHANNEL_ID_LEN	64	N/A
DPSDK_GPS_LEN	50	N/A
DPSDK_TYPE_LEN	20	N/A
DPSDK_DEVICE_NAME_LEN	50	N/A
DPSDK_CHANNEL_NAME_LEN	50	N/A
DPSDK_ORG_NAME_LEN	50	N/A
DPSDK_SERVER_CODE_LEN	64	N/A
DPSDK_PTZ_EXTEND_LEN	255	N/A
DPSDK_PTZ_TIME_LEN	15	N/A
DPSDK_PRESETPOINT_CODE_LEN	50	N/A
DPSDK_PRESETPOINT_NAME_LEN	50	N/A
DPSDK_CRUISE_PLAN_LEN	255	N/A
DPSDK_DEVICE_CODE_LEN	50	N/A
DPSDK_USER_LEVEL_LEN	20	N/A
DPSDK_BITMAP_FILE_HEADER_LEN	14	N/A

DPSDK_BITMAP_INFO_HEADER_LEN	40	N/A
DPSDK_TVWALL_NAME_LEN	50	N/A
DPSDK_TVWALL_OWNERCODE_LEN	90	N/A
DPSDK_LINKED_CHANNEL_SIZE	16	N/A
DPSDK_BIRTHDAY_LEN	20	N/A
DPSDK_TIME_LEN	15	N/A
DPSDK_PERSON_ID_LEN	50	N/A
DPSDK_PERSONTYPE_NAME_LEN	100	N/A
DPSDK_USERDEFINEDATA_FILENAME_LEN	256	N/A
DPSDK_PLATE_NUMBER_LEN	20	N/A
MEMURIGHT_LEN	64	N/A
DPSDK_ID_LEN	64	N/A
DPSDK_PWD_EXPIRY_LEN	32	N/A
DPSDK_USER_REMARK_LEN	256	N/A
DPSDK_MEMO_LEN	256	N/A
DPSDK_CONFIG_PARAM_LEN	64	N/A
DPSDK_KEYWORD_LEN	50	N/A

Parent Subject: Event Definition

Interface Function Definition

Resources Initialization & Uninitialization

Parent Subject: Interface Function Definition

Set Log DPSDK_SetLogInfo

Name	Note
Description:	Set log info
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_SetLogInfo(`
	`    const DPSDK_CHAR *pLogPath,`
	`    DPSDK_INT32 iLogLevel`
	`);`

Parameters:	PLogPath [in] Log file root directory iLogLevel [in] Log level, refer to DPSDK_LOG_LEVEL_TYPE definition
Returned value:	Success return 0, failure return Error_code.

Samples:	`DPSDK_CHAR szLogPath[25] = {0};`
	`DPSDK_INT32 iLogLevel = LOG_LEVEL_WARN; // Log level, warning`
	`strcpy(szLogPath, "log\\08-30-30");`
	`DPSDK_INT32 iRet = DPSDK_SetLogInfo(&szLogPath, iLogLevel);`
	`if(iRet == DPSDK_SUCCESS )`
	`{`
	`    //Success, set log`
	`}`

Parent subject: Source Initialization and Anti-initialization

System Initialization DPSDK_Init

Name	Note
Description:	System initialization
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit
Function:	DPSDK_INT32 DPSDK_Init();
Parameters:	None
Returned value:	Success return 0, failure return Error_code.

Samples:	DPSDK_INT32 iRet = DPSDK_Init();
	if(iRet == DPSDK_SUCCESS)
	{
	   //System initialization success
	}

Parent subject: Source Initialization and Anti initialization

System Anti Initialization DPSDK_Uninit

Name	Note
Description：	System anti initialization
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit
Function：	DPSDK_INT32 DPSDK_Uninit();
Parameters：	None
Returned value：	Success return 0, failure return Error code.

Samples：	DPSDK_INT32 iRet = DPSDK_Uninit();
	if(iRet == DPSDK_SUCCESS)
	{
	    //System anti initialization success
	}

Parent subject： Source initialization and anti initialization

Get SDK version number DPSDK_GetVersion

Name	**Note**
Description:	Get sdk version number
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit
Function:	DPSDK_INT32 DPSDK_GetVersion()
Parameters:	None
Returned value:	Success return 0, failure return Error_code.

Samples:	DPSDK_INT32 iRet = DPSDK_GetVersion ();
	if(iRet == DPSDK_SUCCESS)
	{
	  //Get SDK version number success
	}

Parent subject: Source initialization and anti initialization

Set Platform Data Compression Mode DPSDK_SetCompressType

Name	Note
Description:	It is to set platform data compression mode
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 位
Function:	DPSDK_INT32 DPSDK_SetCompressType(DPSDK_INT32 iCompressType);
Parameters:	iCompressType [in] Compression mode, refer to DPSDK_COMPRESS_TYPE definition
Returned value:	Success return 0, failure return Error_code.

Samples:	DPSDK_INT32 iCompressType = COMPRESS_DEFAULT; // Use default compression mode
	DPSDK_INT32 iRet = DPSDK_SetCompressType(iCompressType);
	if(iRet == DPSDK_SUCCESS)
	{
	//Success, set platform data compression mode
	}

Parent subject: Source initializaiton and anti initialization

User Login and Logout

Login DPSDK_Login

Logout DPSDK_Logout

Parent Subject: Interface Function Definition

Login DPSDK_Login

Name	Note
Description :	User login
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function :	`DPSDK_INT32 DPSDK_Login(`
	`    DPSDK_LOGIN_PARAM* pLoginParam,`
	`    DPSDK_INT32* pSessionID`
	`);`

Parameters :	pLoginParam
	  [in] Login parameters
	PSessionID
	  [out] User session ID

Returned value :	Success return 0, failure return Error_code.

Samples :	`DPSDK_LOGIN_PARAM struLoginParam;`
	`memset(&struLoginParam, 0, sizeof(struLoginParam));`
	`struLoginParam.bDomainUser = true;//Domain login`
	`struLoginParam.uiClientType = CLIENT_PC; //PC client`
	`strcpy(struLoginParam.szUserName, "system");`
	`strcpy(struLoginParam.struIP, "172.22.100.249");`
	`strcpy(struLoginParam.uiPort, "37777");`
	`DPSDK_INT32 iSessionID = -1;`
	`DPSDK_INT32 iRet = DPSDK_Login(&struLoginParam, &iSessionID);`
	`if(iRet == DPSDK_SUCCESS )`
	`{`
	`    //Success, users log in and set callback functions, refer to DPSDK_SetEventCallBack`
	`}`

Parent subject : User Login and Logout

Logout DPSDK_Logout

Name	Note
Description：	User log out
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 位

Function：	`DPSDK_INT32 DPSDK_Logout(`
	`    DPSDK_INT32 iSessionID`
	`);`

Parameters：	ISessionID [in] User session ID
Returned value：	Success return 0, failure return Error Code.

Samples：	`DPSDK_INT32 iRet = DPSDK_Logout(iSessionID);`
	`if(iRet == DPSDK_SUCCESS )`
	`{`
	`    //Users log out and set iSessionID as invalid value`
	`}`

Parent subject：Users Login and Logout

Organization Tree

- [Get Organization Tree DPSDK_GetOrganization]
- [Get Device Tree DPSDK_GetDevice]
- [Get All Organization Trees DPSDK_GetAllOrg]
- [Get Device Tree by Layer DPSDK_GetDeviceByLayered]

Parent Subject: Interface Function Definition

Get Organization Tree DPSDK_GetOrganization

Name	Note
Description:	Get organization data according to condition
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	DPSDK_INT32 DPSDK_GetOrganization(
	    DPSDK_INT32 iSessionID,
	    DPSDK_QUERY_ORG_INFO * pQueryOrgInfo,
	    DPSDK_UINT32 uiQueryLen,
	    DPSDK_DataCallback fDataCallBack,
	    DPSDK_VOID* pUserData
	);

Parameters:	iSessionID [in] User session ID pQueryOrgInfo [in] Organization query condition, refer to DPSDK_QUERY_ORG_INFO definition UiQueryLen [in] Organization query condition length fDataCallBack [in] Data sync callback function, data type refer to DPSDK_DATA_TYPE, structure body, refer to DPSDK_ORG_INFO pUserData [in] User data
Returned value:	Success return 0, failure return Error_code.

Samples:	vector<int> vecChannelType;
	GetCheckChannel(vecChannelType);
	int iChannelTypeCount = vecChannelType.size();
	DPSDK_UINT32 uiQueryLen = sizeof(DPSDK_QUERY_ORG_INFO) + (iChannelTypeCount - 1) * sizeof(DPSDK_INT32);
	DPSDK_QUERY_ORG_INFO* pQueryOrgInfo = (DPSDK_QUERY_ORG_INFO*)(new DPSDK_CHAR[uiQueryLen]);
	memset(pQueryOrgInfo, 0, uiQueryLen);
	strcpy(pQueryOrgInfo->szOrgCode, "");
	for (int i = 0; i < iChannelTypeCount; ++i)
	{
	    pQueryOrgInfo->iChannelTypeList[i] = vecChannelType[i];
	}
	DPSDK_INT32 iRet = DPSDK_GetOrganization(CAppData::m_iLoginID, pQueryOrgInfo, uiQueryLen, &DataCallback, &m_struDepInfoAll);
	if(iRet == DPSDK_SUCCESS)
	{

	`    //Successfully get organization data`
	`}`

Parent subject: Organization tree

-

Get Device Tree DPSDK_GetDevice

Name	Note
Description :	Get device data
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit
Function :	DPSDK_INT32 DPSDK_GetDevice(DPSDK_INT32 iSessionID, DPSDK_CHAR* pDeviceList, DPSDK_DEV_ALL_INFO_LIST** pDevAllInfoList);
Parameters :	iSessionID [in] User session ID pDeviceList [in] Device ID list pDevAllInfoList [out] Device data
Returned value :	Success return 0, failure return Error code.

Samples	DPSDK_CHAR* pDeviceList = NULL;
	DPSDK_DEV_ALL_INFO_LIST* pDevAllInfoList = NULL;
	DPSDK_INT32 iRet = DPSDK_GetDevice(iSessionID, pDeviceList, &pDevAllInfoList);
	if (iRet == DPSDK_SUCCESS)
	{
	//Success, get device data
	}
	DPSDK_ReleaseDevBuffer(pDevAllInfoList);

Parent subject : Organization tree

Get All Organization Tree DPSDK_GetAllOrg

Name	Note
Description:	Get all organization tree (exclude device, channel)
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit. • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_GetAllOrg (`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iOrgType,`
	`    DPSDK_DataCallback fDataCallBack,`
	`    DPSDK_VOID* pUserData`
	`);`

Parameters:	iSessionID [in] User session ID iOrgType [in] Organization type, it is a by default, basic organization fDataCallBack [in] Data sync callback function, data type, refer to DPSDK_DATA_TYPE, structure body, refer to DPSDK_ALL_ORG_INFO pUserData [in] User data
Returned value:	Success return 0, failure return Error_code

Samples:	`Dep_Info_All_t struDepInfoAll;`
	`DPSDK_INT32 iRet = DPSDK_GetAllOrg(CAppData::m_iLoginID, 1, &DataCallback, &struDepInfoAll);`
	`if (iRet == DPSDK_SUCCESS)`
	`{`
	`    //Successfully get all organization trees`
	`}`

Parent subject: Organization tree

Get device tree by layers DPSDK_GetDeviceByLayered

Name	Note
Description:	Get device tree by layers
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit
Function:	DPSDK_INT32 DPSDK_GetDeviceByLayered (DPSDK_INT32 iSessionID, DPSDK_GET_DEVICE_LAYERED_PARAM* pParam, DPSDK_PAGE_INFO* pPageInfo, DPSDK_UINT32* pTotal, DPSDK_DataCallback fDataCallBack, DPSDK_VOID* pUserData);
Parameters:	iSessionID [in] User session ID pParam [in] Layered query condition, refer to structure body DPSDK_GET_DEVICE_LAYERED_PARAM pPageInfo [in] Page info, refer to structure body DPSDK_PAGE_INFO pTotal [in] total record number fDataCallBack [in] Data sync callback function, data type, refer to DPSDK_DATA_TYPE, structure body, refer to DPSDK_LAYERED_RESULT_LIST pUserData [in] User data
Returned value:	Success return 0, failure return Error_code.

Samples:	DPSDK_PAGE_INFO struPageInfo;
	memset(&struPageInfo, 0, sizeof(struPageInfo));
	struPageInfo.uiPage = 1;
	struPageInfo.uiPageSize = 100;
	
	DPSDK_UINT32 uiTotal = 0;
	DPSDK_GET_DEVICE_LAYERED_PARAM struParam;
	memset(&struParam, 0, sizeof(struParam));
	struParam.iOrgType = 1;
	struParam.iShowDev = 1;
	struParam.iDeep = 3;
	struParam.iNodeType = nNodeType;
	strcpy(struParam.szID, pParentId);

	`Dep_Info_All* depChild = FinOrgInfo(&m_struDepInfoAll, pParentId);`
	`if (depChild == NULL)`
	`{`
	`    depChild = &m_struDepInfoAll;`
	`}`
	`DPSDK_INT32 iRet = DPSDK_GetDeviceByLayered(CAppData::m_iLoginID, &struParam, &struPageInfo, &uiTotal, &DataCallback, depChild);`
	`if (iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success, get device tree by layers`
	`}`

Parent subject: Organization tree

Video Preview

Parent Subject: Interface Function Definition

Start Unicast Video Preview DPSDK_StartRealPlay

Name	Note
Description :	Start Unicast Video Preview.
OS :	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function :	`DPSDK_INT32 DPSDK_StartRealPlay(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_REALPLAY_PARAM* pRealPlayParam,`
	`    DPSDK_INT32* pMediaSessionID`
	`);`

Parameter :	iSessionID [in] User Session ID pRealPlayParam [in] Video Preview Parameters. For details, please refer to DPSDK_REALPLAY_PARAM structure. pMediaSessionID [out] Media Session ID
Returned Value :	Success returns 0. Failure returns error_code.

Sample :	`//Unicast`
	`DPSDK_REALPLAY_PARAM struRealParam;`
	`memset(&struRealParam, 0, sizeof(struRealParam));`
	`struRealParam.iUsedVcs = 0; // Marks whether VCS transcoding is needed or not. 0 means no transcoding is required.`
	`struRealParam.struMediaBaseParam.iDataType = 1; // Video Type. 1 means video.`
	`struRealParam.struMediaBaseParam.iStreamType = 1; // Code Stream Type. 1 means the main code stream.`
	`struRealParam.struMediaBaseParam.iDecodeType = DPSDK_DECODE_HW; //Decoding Type. 1 means CPU decoding.`
	`struRealParam.struMediaBaseParam.iStreamMode = DPSDK_STREAM_REAL_MODE; // Play Mode. 0 means real-play priority mode.`
	`DPSDK_INT32 iMediaSessionID = -1;`
	`DPSDK_INT32 iRet = DPSDK_StartRealPlay(iSessionID, &struRealParam, &iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS )`
	`{`
	`    //Success`
	`}`

Parent Subject : Video Preview

Start Multi-view video Preview DPSDK_StartMulitviewRealPlay

Name	Note
Description:	Start multi-view video preview.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function:	DPSDK_INT32 DPSDK_StartMulitviewRealPlay(
	    DPSDK_INT32 iSessionID,
	    DPSDK_MULITVIEW_REALPLAY_PARAM* pMulitviewParam,
	    DPSDK_INT32* pMediaSessionID
	);

Parameter:	iSessionID [in] User Session ID pMulitviewParam [in] Video Preview Parameters. For details, please refer to DPSDK_MULITVIEW_REALPLAY_PARAM structure. pMediaSessionID [out] Media Session ID
Returned Value:	Success returns 0. Failure returns error_code.

Sample:	DPSDK_MULITVIEW_REALPLAY_PARAM struMulitRealParam;
	memset(&struMulitRealParam, 0, sizeof(struMulitRealParam));
	struMulitRealParam.struMediaBaseParam.iDecodeType = DPSDK_DECODE_HW;
	DPSDK_INT32 pMediaSessionID = -1;
	DPSDK_INT32 iRet = DPSDK_StartMulitviewRealPlay(iSessionID, &struMulitRealParam, &pMediaSessionID);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success. Start multi-view video preview.
	}

Parent Subject: Video Preview

Start Multicast Video preview DPSDK_StartMulitcastRealPlay

Name	Note
Description:	Start multicast video preview.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.
Function:	DPSDK_INT32 DPSDK_StartMulitcastRealPlay (DPSDK_INT32 iSessionID, DPSDK_MULITCAST_REALPLAY_PARAM* pMulitcastParam, DPSDK_INT32* pMediaSessionID);
Parameter:	iSessionID [in] User Session ID pMulitcastParam [in] Video Preview Parameters. For details, please refer to DPSDK_MULITCAST_REALPLAY_PARAM structure. pMediaSessionID [out] Media Session ID
Returned Value:	Success returns 0. Failure returns error_code.

Sample:	DPSDK_MULITCAST_REALPLAY_PARAM struMulitcastParam;
	memset(&struMulitcastParam, 0, sizeof(struMulitcastParam));
	strcpy(struMulitcastParam.szTrackId, "");
	struMulitcastParam.struMediaBaseParam.iStreamType = 1; // Code Stream Type. 1=Main Stream;2= Auxiliary Stream
	struMulitcastParam.struMediaBaseParam.iDataType = 1; //Video Type. 1=Video; 2=Audio; 3=Audio / video
	struMulitcastParam.struMediaBaseParam.iDecodeType = 1; //Decoding Type. See DPSDK_DECODE_TYPE definition.
	struMulitcastParam.struMediaBaseParam.iStreamMode = 1; //Play Mode. See DPSDK_STREAM_MODE definition.
	struMulitcastParam.struMediaBaseParam.uiDelayTime = 10; //Play Delay Time. Be effective when iStreamMode is 3. Unit: ms
	DPSDK_INT32 iMediaSessionID = -1;
	DPSDK_INT32 iRet = DPSDK_StartMulitcastRealPlay (iSessionID, & struMulitcastParam, &iMediaSessionID);
	if(iRet == DPSDK_SUCCESS)
	{
	//Success
	}

Parent Subject: Video Preview

Stop Video Preview DPSDK_StopRealPlay

Name	Note
Description:	Stop video preview.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function:	`DPSDK_INT32 DPSDK_StopRealPlay(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID`
	`);`

Parameter:	iSessionID
	  [in] User Session ID
	iMediaSessionID
	  [in] Media Session ID

Returned Value:	Success returns 0. Failure returns error_code.

Sample:	`DPSDK_INT32 iRet = DPSDK_StopRealPlay(iSessionID, iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success. Set the Media Session ID to an invalid value.`
	`    iMediaSessionID = 0;`
	`}`

Parent Subject: Video Preview

Get Play Stream Mode DPSDK_GetPlayStreamMode

Name	**Note**
Description:	Get play stream mode.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function:	DPSDK_INT32 DPSDK_GetPlayStreamMode(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_UINT32* pStreamMode
	);

Parameter:	iSessionID
	  [in] User Session ID
	IMediaSessionID
	  [in] Media Session ID
	PStreamMode
	  [out] Play Mode. Please refer to DPSDK_STREAM_MODE

Returned Value:	Success returns 0. Failure returns error_code.

Sample:	DPSDK_UINT32 iStreamMode = -1;
	DPSDK_INT32 iRet = DPSDK_GetPlayStreamMode(iSessionID, iMediaSessionID, &iStreamMode);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent Subject: Video preview

Set Play Stream Mode DPSDK_SetPlayStreamMode

Name	Note
Description:	Set play stream mode.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1(2.6.16.21 higher) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function:	`DPSDK_INT32 DPSDK_SetPlayStreamMode(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_UINT32 uiStreamMode,`
	`    DPSDK_UINT32 uiDelayTime`
	`);`

Parameter:	iSessionID [in] User Session ID iMediaSessionID [in] Media Session ID UiStreamMode [in] Play Stream Mode. For details, please refer to DPSDK_STREAM_MODE enumeration. uiDelayTime [in] Delay Time. Be effective when uiStreamMode is STREAM_CUSTOM_MODE.
Returned Value:	Success returns 0. Failure returns error_code.

Sample:	`DPSDK_UINT32 uiStreamMode = DPSDK_STREAM_REAL_MODE;    // Real-Time Priority Mode`
	`DPSDK_UINT32 uiDelayTime = 0;    // Be effective when uiStreamMode is STREAM_CUSTOM_MODE.`
	`DPSDK_INT32 iRet = DPSDK_SetPlayStreamMode(iSessionID, iMediaSessionID, uiStreamMode, uiDelayTime)`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success. Set the play stream mode.`
	`}`

Parent Subject: Video Preview

Operate RealPlay DPSDK_OperateRealPlay

Name	Note
Description：	Operate (lock, unlock) the video.
OS：	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function：	DPSDK_INT32 DPSDK_OperateRealPlay(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_CHAR* pCodeID,
	    DPSDK_INT32 iOperateType
	);

Parameter：	iSessionID [in] User Session ID iMediaSessionID [in] Media Session ID pCodeID [in] Video Channel ID iOperateType [in] Operation Type. For Details, please refer to DPSDK_VIDEO_LOCK_TYPE
Returned Value：	Success returns 0. Failure returns error_code.

Sample：	DPSDK_CHAR szCodeID[DPSDK_CHANNEL_ID_LEN] = {0} ;
	DPSDK_INT32 iOperateType = DPSDK_VIDEO_CMD_LOCK; // Lock the current camera.
	strcpy(szCodeID, "1000001$1$0$1");
	DPSDK_INT32 iRet = DPSDK_SetPlayStreamMode(iSessionID, iMediaSessionID, szCodeID, iOperateType);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent Subject：Video Preview

Video Playback

Parent Subject: Interface Function Definition

Get Channel Record Information DPSDK_GetRecordStatus

Name	Note
Description:	Get Channel Record Information.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits • SUSE Linux 11 SP1(2.6.16.21以上) 64 Bits • SUSE Linux 10 32 Bits • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits

Function:	`DPSDK_INT32 DPSDK_GetRecordStatus(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_CHAR* pChannelID,`
	`    DPSDK_RECORD_STATUS_INFO* pRecordInfo`
	`);`

Parameter:	iSessionID [in] User Session ID pChannelID [in] Channel ID pRecordInfo [out] Channel Record Information. For details, please refer to DPSDK_RECORD_STATUS_INFO structure.
Returned Value:	Success returns. Failure returns error code.

Sample:	`DPSDK_RECORD_STATUS_INFO struRecordInfo;`
	`memset(&struRecordInfo, 0, sizeof(DPSDK_RECORD_STATUS_INFO));`
	`DPSDK_CHAR* pChannelID = new DPSDK_CHAR;`
	`strcpy(pChannelID, "168383947B19V88R2VM0DOT");`
	`DPSDK_INT32 iRet = DPSDK_GetRecordStatus(CAppData::m_iLoginID, pChannelID, &struRecordInfo);`
	`if(iRet == DPSDK_SUCCESS )`
	`{`
	`    //Success`
	`}`
	`delete pChannelID;`
	`pChannelID = NULL;`

Parent Subject: Video Playback

Query Record DPSDK_QueryRecord

Name	Note
Description:	Query record.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1(2.6.16.21 above) 64 Bits • SUSE Linux 10 32 Bits • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits

Function:	DPSDK_INT32 DPSDK_QueryRecord(
	    DPSDK_INT32 iSessionID,
	    DPSDK_QUERY_RECORD_PARAM* pQueryRecord,
	    DPSDK_RECORD_INFO_LIST* pRecordList,
	    DPSDK_UINT32 uiBufLen
	);

Parameter:	iSessionID [in] User Session ID pQueryRecord [in] Conditions of Record Date Query. For details, please refer to DPSDK_QUERY_RECORD_PARAM structure. pRecordList [out] Query Result. For details, please refer to DPSDK_RECORD_INFO_LIST uiBufLen [in] Buffer Size
Returned Value:	Success returns. Failure returns error code.

Sample:	DPSDK_QUERY_RECORD_PARAM struQueryRecord;
	memset(&struQueryRecord, 0, sizeof(&struQueryRecord));
	strcpy(struQueryRecord.szCameraId, "168383947B19V88R2VM0DOT");
	struQueryRecord.iStreamType = STREAM_MAIN_STREAM;
	struQueryRecord.iRecordType = DPSDK_RECORD_TYPE_ALL;
	
	struQueryRecord.tBeginTime = ParseDateTime(ui.StartdateTimeEdit);
	struQueryRecord.tEndTime = ParseDateTime(ui.EnddateTimeEdit);
	
	DPSDK_UINT32 uiNum = 100;
	DPSDK_UINT32 uiLen = sizeof(DPSDK_RECORD_INFO_LIST) + (uiNum - 1) * sizeof(DPSDK_SINGLE_RECORD_INFO);
	DPSDK_RECORD_INFO_LIST* pList = (DPSDK_RECORD_INFO_LIST*)(new DPSDK_CHAR[uiLen]);
	memset(pList, 0, uiLen);
	
	int iRet = DPSDK_QueryRecord(CAppData::m_iLoginID, &struQueryRecord, pList, uiLen);
	if(iRet == DPSDK_SUCCESS)
	{

	//Success }

Parent Subject: Video Playback

Query Record Date DPSDK_QueryRecordDate

Name	Note
Description:	Query record date.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits • SUSE Linux 11 SP1(2.6.16.21 above) 64 Bits • SUSE Linux 10 32 Bits • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits

Function:	DPSDK_INT32 DPSDK_QueryRecordDate(
	    DPSDK_INT32 iSessionID,
	    DPSDK_QUERY_RECORD_DATE_PARAM* pQueryDateInfo,
	    DPSDK_RECORD_DATE_INFO* pRecordDate
	);

Parameter:	iSessionID [in] User Session ID pQueryDateInfo [in] Condition of Record Date Query. For details, please refer to DPSDK_QUERY_RECORD_DATE_PARAM structure. pRecordDate [out] Query Result. Detailed parameters please refer to DPSDK_RECORD_DATE_INFO
Returned Value:	Success returns 0. Failure returns error_code.

Sample:	DPSDK_QUERY_RECORD_DATE_PARAM struQueryDateInfo;
	memset(&struQueryDateInfo, 0, sizeof(DPSDK_QUERY_RECORD_DATE_PARAM));
	strcpy(struQueryDateInfo.szCameraId, "168383947B19V88R2VM0DOT");
	struQueryDateInfo.iSourceType = DPSDK_SOURCE_TYPE_ALL;
	
	DPSDK_RECORD_DATE_INFO struRecordDate;
	memset(&struRecordDate, 0, sizeof(DPSDK_RECORD_DATE_INFO));
	
	DPSDK_INT32 iRet = DPSDK_QueryRecordDate(CAppData::m_iLoginID, &struQueryDateInfo, &struRecordDate);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent Subject: Video Playback

Lock Record File DPSDK_LockRecordFile

Name	**Note**
Description :	Lock record file.
OS :	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1(2.6.16.21 above) 64 Bits • SUSE Linux 10 32 Bits • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits

Function :	DPSDK_INT32 DPSDK_LockRecordFile(
	    DPSDK_INT32 iSessionID,
	    DPSDK_LOCK_RECORD_FILE_PARAM* pLockFileInfo,
	    DPSDK_LOCK_RECORD_FILE_RESULT* pResult
	);

Parameter :	iSessionID [in] User Session ID pLockFileInfo [in] Lock Record File Parameter. For details, please refer to DPSDK_LOCK_RECORD_FILE_PARAM structure. pResult [out] Lock Record File Result. Detailed parameters please see DPSDK_LOCK_RECORD_FILE_RESULT
Returned Value :	Success returns 0. Failure returns error_code.

Sample :	DPSDK_LOCK_RECORD_FILE_PARAM struLockFileInfo;
	memset(&struLockFileInfo, 0, sizeof(struLockFileInfo));
	strcpy(struLockFileInfo.szFilename, "");
	strcpy(struLockFileInfo.szCameraId, "168383947B19V88R2VM0DOT");
	DPSDK_LOCK_RECORD_FILE_RESULT struResult;
	memset(&struResult, 0, sizeof(struResult));
	
	int iRet = DPSDK_LockRecordFile(CAppData::m_iLoginID, &struLockFileInfo, &struResult);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent Subject: Video Playback

Unlock Record File DPSDK_UnlockRecordFile

Name	**Note**
Description:	Unlock record file.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits • SUSE Linux 11 SP1(2.6.16.21 above) 64 Bits • SUSE Linux 10 32 Bits • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits

Function:	DPSDK_INT32 DPSDK_UnlockRecordFile(
	    DPSDK_INT32 iSessionID,
	    DPSDK_UNLOCK_RECORD_FILE_PARAM* pUnlockFileInfo
	    DPSDK_LOCK_RECORD_FILE_RESULT* pResult
	);

Parameter:	iSessionID [in] User Session ID pUnlockFileInfo [in] Unlock Record File Parameter. For details please refer to DPSDK_UNLOCK_RECORD_FILE_PARAM structure. pResult [out] Unlock Result. Detailed parameters please see DPSDK_LOCK_RECORD_FILE_RESULT
Returned Value:	Success returns 0. Failure returns error code.

Sample:	DPSDK_UNLOCK_RECORD_FILE_PARAM struUnlockFileInfo;
	memset(&struUnlockFileInfo, 0, sizeof(struUnlockFileInfo));
	strcpy(struUnlockFileInfo.szFilename, "");
	strcpy(struUnlockFileInfo.szCameraId, " 168383947B19V88R2VM0DOT ");
	if (ui.checkBox_IsForce->isChecked())
	{
	        struUnlockFileInfo.bForce = true;
	}
	else
	{
	        struUnlockFileInfo.bForce = false;
	}
	DPSDK_LOCK_RECORD_FILE_RESULT struResult;
	memset(&struResult, 0, sizeof(struResult));
	int iRet = DPSDK_UnlockRecordFile(CAppData::m_iLoginID, &struUnlockFileInfo, &struResult);
	if(iRet == DPSDK_SUCCESS)
	{
	     //Success
	}

Parent Subject: Video Playback

Start Playback by Time DPSDK_StartPlaybackByTime

Name	Note
Description:	Start playback by time.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.
Function:	DPSDK_INT32 DPSDK_StartPlaybackByTime(DPSDK_INT32 iSessionID, DPSDK_PLAYBACK_BY_TIME_PARAM* pPlaybackParam, DPSDK_INT32* pMediaSessionID);
Parameter:	iSessionID [in] User Session ID pPlaybackParam [in] Playback by Time Parameter. For details, please refer to DPSDK_PLAYBACK_BY_TIME_PARAM structure. pMediaSessionID [out] Media Session ID
Returned Value:	Success returns 0. Failure returns error_code.

Sample:	DPSDK_PLAYBACK_BY_TIME_PARAM struRealParam;
	memset(&struRealParam, 0, sizeof(struRealParam));
	
	struRealParam.pHWnd = (HCWND)ui.widgetVideoWindow->winId();
	struRealParam.iStreamType = STREAM_MAIN_STREAM;
	struRealParam.iRecordType = DPSDK_RECORD_TYPE_ALL;
	struRealParam.iRecordSource = DPSDK_SOURCE_TYPE_ALL;
	
	struRealParam.tBeginTime = ParseDateTime(ui.StartdateTimeEdit);
	struRealParam.tEndTime = ParseDateTime(ui.EnddateTimeEdit);
	
	DPSDK_INT32 iRet = DPSDK_StartPlaybackByTime(CAppData::m_iLoginID, &struRealParam, &m_iMediaSessionID);
	if(iRet == DPSDK_SUCCESS)
	{
	//Success
	}

Parent Subject: Video Playback

Start Playback by Record File DPSDK_StartPlaybackByFile

Name	Note
Description：	Start playback by record file.
OS：	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits • SUSE Linux 11 SP1(2.6.16.21 above) 64 Bits • SUSE Linux 10 32 Bits • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits

Function：	`DPSDK_INT32 DPSDK_StartPlaybackByFile(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_PLAYBACK_BY_FILE_PARAM* pPlaybackParam,`
	`    DPSDK_INT32* pMediaSessionID`
	`);`

Parameter：	iSessionID [in] User Session ID pPlaybackParam [in] Playback by File Parameter. For details, please refer to DPSDK_PLAYBACK_BY_FILE_PARAM structure. pMediaSessionID [out] Media Session ID
Returned Value：	Success returns 0. Failure returns error_code.

Sample：	`DPSDK_PLAYBACK_BY_FILE_PARAM struPlaybackParam;`
	`memset(&struPlaybackParam, 0, sizeof(struPlaybackParam));`
	
	`struPlaybackParam.pHWnd = (HCWND)ui.widgetVideoWindow->winId();`
	`strcpy(struPlaybackParam.szCodeId, "68383947B19V88R2VM0DOT");`
	`struPlaybackParam.iRecordSource = DPSDK_SOURCE_TYPE_ALL;`
	`struPlaybackParam.tBeginTime = ParseDateTime(ui.StartdateTimeEdit);`
	`struPlaybackParam.tEndTime = ParseDateTime(ui.EnddateTimeEdit);`
	
	`DPSDK_INT32 iRet = DPSDK_StartPlaybackByFile(CAppData::m_iLoginID, &struPlaybackParam, &m_iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS )`
	`{`
	`    //Success`
	`}`

Parent Subject：Video Playback

Stop Playback DPSDK_StopPlayback

Name	Note
Description:	Stop Playback.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1(2.6.16.21 above) 64 Bits • SUSE Linux 10 32 Bits • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits

Function:	`DPSDK_INT32 DPSDK_StopPlayback(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID`
	`);`

Parameter:	iSessionID
	  [in] User Session ID
	iMediaSessionID
	  [in] Media Session ID

Returned Value:	Success returns 0. Failure returns error code.

Sample:	`if (m_iMediaSessionID > 0)`
	`{`
	`    DPSDK_INT32 iRet = DPSDK_StopPlayback(CAppData::m_iLoginID, m_iMediaSessionID);`
	`        if(iRet == DPSDK_SUCCESS )`
	`        {`
	`            //Success`
	`        }`
	`}`

Parent Subject: Video Playback

Playback Pause DPSDK_PlaybackPause

Name	Note
Description：	Playback pause.
OS：	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function：	`DPSDK_INT32 DPSDK_PlaybackPause(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID`
	`);`

Parameter：	iSessionID
	  [in] User Session ID
	iMediaSessionID
	  [in] Media Session ID

Returned Value：	Success returns 0. Failure returns error_code.

Sample：	`DPSDK_INT32 iRet = DPSDK_PlaybackPause(CAppData::m_iLoginID, m_iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS )`
	`{`
	`    //Success`
	`}`

Parent Subject：Video Playback

Resume Playback DPSDK_PlaybackResume

Name	**Note**
Description：	Resume Playback.
OS：	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function：	`DPSDK_INT32 DPSDK_PlaybackResume(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID`
	`);`

Parameter：	iSessionID
	  [in] User Session ID
	iMediaSessionID
	  [in] Media Session ID

Returned Value：	Success returns 0. Failure returns error_code.

Sample：	`DPSDK_INT32 iRet = DPSDK_PlaybackResume (CAppData::m_iLoginID, m_iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS )`
	`{`
	`    //Success`
	`}`

Parent Subject：Video Playback

Playback by Frame Step DPSDK_PlaybackFrameStep

Name	Note
Description：	Playback pause.
OS：	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function：	`DPSDK_INT32 DPSDK_PlaybackFrameStep(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID`
	`);`

Parameter：	iSessionID
	  [in] User Session ID
	iMediaSessionID
	  [in] Media Session ID

Returned Value：	Success returns 0. Failure returns error_code.

Sample：	`DPSDK_INT32 iRet = DPSDK_PlaybackFrameStep (CAppData::m_iLoginID, m_iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS )`
	`{`
	`    //Success`
	`}`

Parent Subject: Video Playback

Playback Seek DPSDK_PlaybackSeek

Name	**Note**
Description :	Playback seeking.
OS :	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function :	`DPSDK_INT32 DPSDK_PlaybackSeek(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_PLAYBACK_SEEK_PARAM* pPlaybackSeekParam`
	`);`

Parameter :	iSessionID [in] User Session ID iMediaSessionID [in] Media Session ID pPlaybackSeekParam [in] seek Playback Parameters. For details, please refer to DPSDK_PLAYBACK_SEEK_PARAM structure.
Returned Value :	Success returns 0. Failure returns error_code.

Sample :	`DPSDK_PLAYBACK_SEEK_PARAM struPlaybackSeekParam;`
	`memset(&struPlaybackSeekParam, 0, sizeof(DPSDK_PLAYBACK_SEEK_PARAM));`
	`struPlaybackSeekParam.iDirection = 1;`
	`struPlaybackSeekParam.iSpeed = DPSDK_PB_NORMAL;`
	`struPlaybackSeekParam.tBeginTime = ParseDateTime(ui.StartdateTimeEdit);`
	`struPlaybackSeekParam.tEndTime = ParseDateTime(ui.EnddateTimeEdit);`
	`DPSDK_INT32 iRet = DPSDK_SUCCESS;`
	`iRet = DPSDK_PlaybackSeek(CAppData::m_iLoginID, m_iMediaSessionID, &struPlaybackSeekParam);`
	`if(iRet == DPSDK_SUCCESS )`
	`{`
	`    //Success`
	`}`

Parent Subject : Video Playback

Set Playback Speed DPSDK_SetPlaybackSpeed

Name	Note
Description:	Set playback speed.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function:	`DPSDK_INT32 DPSDK_SetPlaybackSpeed(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_PLAYBACK_SPEED iSpeed`
	`);`

Parameter:	iSessionID [in] User Session ID iMediaSessionID [in] Media Session ID iSpeed [in] Playback Speed. For details, please refer to DPSDK_PLAYBACK_SPEED structure.
Returned Value:	Success returns 0. Failure returns error_code.

Sample:	`int iParam = 1;`
	`DPSDK_PLAYBACK_SPEED iSpeed = SetSpeed(iParam);`
	`DPSDK_INT32 iRet = DPSDK_SetPlaybackSpeed(CAppData::m_iLoginID, m_iMediaSessionID, iSpeed);`
	`if(iRet == DPSDK_SUCCESS )`
	`{`
	`    //Success`
	`}`

Parent Subject: Video Playback

Get Current Play Time DPSDK_GetPlayedTime

Name	Note
Description:	Get current play time.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.
Function:	DPSDK_INT32 DPSDK_GetPlayedTime(DPSDK_INT32 iSessionID, DPSDK_INT32 iMediaSessionID, DPSDK_TIMET* pTime);
Parameter:	iSessionID [in] User Session ID iMediaSessionID [in] Media Session ID pTime [out] Play Time
Returned Value:	Success returns 0. Failure returns error_code.

Sample:	DPSDK_TIMET tTime = 0;
	DPSDK_INT32 iRet = DPSDK_GetPlayedTime(CAppData::m_iLoginID, m_iMediaSessionID, &tTime);
	if(iRet == DPSDK_SUCCESS)
	{
	//Success
	}

Parent Subject: Video Playback

Get Stream Provider Type DPSDK_GetProviderType

Name	**Note**
Description：	Get stream provider type.
OS：	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function：	`DPSDK_INT32 DPSDK_GetProviderType(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_INT32 *pProviderType`
	`);`

Parameter：	iSessionID
	  [in] User Session ID
	iMediaSessionID
	  [in] Media Session ID
	pProviderType
	  [out] Provider Type

Returned Value：	Success returns 0. Failure returns error_code.

Sample：	`DPSDK_INT32 iProviderType = 0;`
	`DPSDK_INT32 iRet = DPSDK_GetProviderType(CAppData::m_iLoginID, m_iMediaSessionID, &iProviderType);`
	`if(iRet == DPSDK_SUCCESS )`
	`{`
	`    //Success`
	`}`

Parent Subject： Video Playback

Start Manual Record DPSDK_StartRemoteRecord

Name	**Note**
Description:	Start Manual Record
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function:	`DPSDK_INT32 DPSDK_StartRemoteRecord(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_PTZOPERATE_STARTREMOTERECORD_PARAM* pStartRemoteRecordParam,`
	`    DPSDK_PTZOPERATE_REMOTERECORD_RESULT* pStartRemoteRecordResult`
	`);`

Parameter:	iSessionID [in] User Session ID pStartRemoteRecordParam [in] Parameters to Start Manual Record. For details, please refer to DPSDK_PTZOPERATE_STARTREMOTERECORD_PARAM structure. pStartRemoteRecordResult [out] Operation Result. Details please see DPSDK_PTZOPERATE_REMOTERECORD_RESULT structure.
Returned Value:	Success returns 0. Failure returns error_code.

Sample:	`DPSDK_PTZOPERATE_STARTREMOTERECORD_PARAM struStartRemoteRecordParam;`
	`memset(&struStartRemoteRecordParam, 0, sizeof(DPSDK_PTZOPERATE_STARTREMOTERECORD_PARAM));`
	`strcpy(struStartRemoteRecordParam.szChannelId, "168383947B19V88R2VM0DOT");`
	`struStartRemoteRecordParam.iStreamType = 1;`
	`struStartRemoteRecordParam.iRecordDuration = 3600;`
	`DPSDK_PTZOPERATE_REMOTERECORD_RESULT struStartRemoteRecordResult;`
	`memset(&struStartRemoteRecordResult, 0, sizeof(DPSDK_PTZOPERATE_REMOTERECORD_RESULT));`
	`DPSDK_INT32 iRet = DPSDK_StartRemoteRecord(CAppData::m_iLoginID, &struStartRemoteRecordParam, &struStartRemoteRecordResult);`
	`if(iRet == DPSDK_SUCCESS )`
	`{`
	`    //Success`
	`}`

Parent Subject: Video Playback

Stop Manual Record DPSDK_StopRemoteRecord

Name	**Note**
Description：	Stop manual record.
OS：	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1(2.6.16.21以上) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.
Function：	DPSDK_INT32 DPSDK_StopRemoteRecord(DPSDK_INT32 iSessionID, DPSDK_PTZOPERATE_STOPREMOTERECORD_PARAM* pStopRemoteRecordParam, DPSDK_PTZOPERATE_REMOTERECORD_RESULT* pStopRemoteRecordResult);
Parameter：	iSessionID [in] User Session ID pStopRemoteRecordParam [in] Parameters to Stop Manual Record. For details, please refer to DPSDK_PTZOPERATE_STOPREMOTERECORD_PARAM structure. pStartRemoteRecordResult [out] Operation Result. Details please see DPSDK_PTZOPERATE_REMOTERECORD_RESULT structure.
Returned Value：	Success returns 0. Failure returns error_code.

Sample：	DPSDK_PTZOPERATE_STOPREMOTERECORD_PARAM struStopRemoteRecordParam;
	memset(&struStopRemoteRecordParam, 0, sizeof(DPSDK_PTZOPERATE_STOPREMOTERECORD_PARAM));
	strcpy(struStopRemoteRecordParam.szChannelId, "168383947B19V88R2VM0DOT");
	struStopRemoteRecordParam.iStreamType = 1;
	DPSDK_PTZOPERATE_REMOTERECORD_RESULT struStopRemoteRecordResult;
	memset(&struStopRemoteRecordResult, 0, sizeof(DPSDK_PTZOPERATE_REMOTERECORD_RESULT));
	DPSDK_INT32 iRet = DPSDK_StopRemoteRecord(CAppData::m_iLoginID, &struStopRemoteRecordParam, &struStopRemoteRecordResult);
	if(iRet == DPSDK_SUCCESS)
	{
	//Success
	}

Parent Subject：Video Playback

Download

Parent Subject: Interface Function Definition

Download Record by Time DPSDK_StartDownloadRecordByTime

Name	Note
Description:	Download by time.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function:	`DPSDK_INT32 DPSDK_StartDownloadRecordByTime(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_DOWNLOAD_BY_TIME_PARAM* pDownloadByTimeParam,`
	`    DPSDK_INT32* pMediaSessionID`
	`);`

Parameter:	iSessionID [in] User Session ID pDownloadByTimeParam [in] Detailed meaning please refer to DPSDK_DOWNLOAD_BY_TIME_PARAM pMediaSessionID [out] Media Session ID
Returned Value:	Success returns 0. Failure returns error_code.

Sample:	`DPSDK_DOWNLOAD_BY_TIME_PARAM struDownloadByTimeParam;`
	`memset(&struDownloadByTimeParam, 0, sizeof(struDownloadByTimeParam));`
	`struDownloadByTimeParam.tBeginTime = ParseDateTime(ui.StartdateTimeEdit);`
	`struDownloadByTimeParam.tEndTime = ParseDateTime(ui.EnddateTimeEdit);`
	`struDownloadByTimeParam.iFileFormat = DPSDK_FILE_FORMAT_NORMAL;`
	`struDownloadByTimeParam.iNameRule =DPSDK_NAME_RULE_TIME_CHANNELID;`
	`struDownloadByTimeParam.iSourceType = DPSDK_SOURCE_TYPE_ALL;`
	`struDownloadByTimeParam.iStreamType = STREAM_MAIN_STREAM;`
	`struDownloadByTimeParam.iRecordType = DPSDK_RECORD_TYPE_MANUAL;`
	`strcpy(struDownloadByTimeParam.szChannelID, "");`
	`strcpy(struDownloadByTimeParam.szChannelName, "");`
	`strcpy(struDownloadByTimeParam.szDownloadPath, "");`
	`strcpy(struDownloadByTimeParam.szDownloadFileName, "");`
	`struDownloadByTimeParam.fEventCallBack = QPlayback::EventCallBack;`
	`DPSDK_INT32 iRet = DPSDK_StartDownloadRecordByTime(CAppData::m_iLoginID,`
	`&struDownloadByTimeParam, &m_iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent Subject: Download

Download Record by File DPSDK_StartDownloadRecordByFile

Name	**Note**
Description :	Download by file.
OS :	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function :	DPSDK_INT32 DPSDK_StartDownloadRecordByFile(
	    DPSDK_INT32 iSessionID,
	    DPSDK_DOWNLOAD_BY_FILE_PARAM* pDownloadByFileParam
	    DPSDK_INT32* pMediaSessionID
	);

Parameter :	iSessionID [in] User Session ID pDownloadByFileParam [in] Detailed meaning please refer to DPSDK_DOWNLOAD_BY_FILE_PARAM pMediaSessionID [out] Media Session ID
Returned Value :	Success returns 0. Failure returns error_code.

Sample :	DPSDK_DOWNLOAD_BY_FILE_PARAM struDownloadByFileParam;
	memset(&struDownloadByFileParam, 0, sizeof(struDownloadByFileParam));
	struDownloadByFileParam.tBeginTime = ParseDateTime(ui.StartdateTimeEdit);
	struDownloadByFileParam.tEndTime = ParseDateTime(ui.EnddateTimeEdit);
	struDownloadByFileParam.iFileFormat = DPSDK_FILE_FORMAT_NORMAL;
	struDownloadByFileParam.iNameRule =DPSDK_NAME_RULE_TIME_CHANNELID;
	struDownloadByFileParam.iSourceType = DPSDK_SOURCE_TYPE_ALL;
	strcpy(struDownloadByFileParam.szChannelID, "168383947B19V88R2VM0DOT");
	strcpy(struDownloadByFileParam.szChannelName, "");
	strcpy(struDownloadByFileParam.szDownloadPath, "");
	strcpy(struDownloadByFileParam.szDownloadFileName, "");
	struDownloadByFileParam.fEventCallBack = QPlayback::EventCallBack;
	strcpy(struDownloadByFileParam.szFilename, "");
	DPSDK_INT32 iRet = DPSDK_StartDownloadRecordByFile(CAppData::m_iLoginID, &struDownloadByFileParam, &m_iMediaSessionID);
	 if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent Subject : Download

Stop Record Download DPSDK_StopDownloadRecord

Name	Note
Description:	Stop record download.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function:	`DPSDK_INT32 DPSDK_StopDownloadRecord(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32* pMediaSessionID`
	`);`

Parameter:	iSessionID
	  [in] User Session ID
	pMediaSessionID
	  [in] Media Session ID

Returned Value:	Success returns 0. Failure returns error_code.

Sample:	`DPSDK_INT32 iRet = DPSDK_StopDownloadRecord(CAppData::m_iLoginID, m_iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent Subject: Download

Pause Record Download DPSDK_PauseDownloadRecord

Name	Note
Description：	Pause record download.
OS：	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function：	`DPSDK_INT32 DPSDK_PauseDownloadRecord(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32* pMediaSessionID`
	`);`

Parameter：	iSessionID
	  [in] User Session ID
	pMediaSessionID
	  [in] Media Session ID

Returned Value：	Success returns 0. Failure returns error_code.

Sample：	`DPSDK_INT32 iRet = DPSDK_PauseDownloadRecord(CAppData::m_iLoginID, m_iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent Subject: Download

Resume Record Download DPSDK_ResumeDownloadRecord

Name	Note
Description:	Resume record download.
OS	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function:	`DPSDK_INT32 DPSDK_ResumeDownloadRecord(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32* pMediaSessionID`
	`);`

Parameter:	iSessionID
	  [in] User Session ID
	pMediaSessionID
	  [in] Media Session ID

Returned Value:	Success returns 0. Failure returns error_code.

Sample	`DPSDK_INT32 iRet = DPSDK_ResumeDownloadRecord(CAppData::m_iLoginID, m_iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent Subject: Download

Get Record Download Information DPSDK_GetDownloadRecordInfo

Name	Note
Description:	Get record download information.
OS:	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function:	DPSDK_INT32 DPSDK_GetDownloadRecordInfo(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32* pMediaSessionID,
	    DPSDK_DOWNLOAD_RECORD_INFO* pDownloadInfo,
	    DPSDK_UINT32 uiBufLen
	);

Parameter:	iSessionID [in] User Session ID pMediaSessionID [in] Media Session ID pDownloadInfo [out] Record Download Information. Details please refer to DPSDK_DOWNLOAD_RECORD_INFO uiBufLen [in] Buffer Size
Returned Value:	Success returns 0. Failure returns error_code.

Sample:	DPSDK_UINT32 uiNum = 256;
	DPSDK_INT32 uiBufLen = sizeof(DPSDK_DOWNLOAD_RECORD_INFO) + (uiNum - 1) * sizeof(DPSDK_CHAR);
	DPSDK_DOWNLOAD_RECORD_INFO* pDownloadInfo = (DPSDK_DOWNLOAD_RECORD_INFO*)(new DPSDK_CHAR[uiBufLen]);
	DPSDK_INT32 iRet = DPSDK_GetDownloadRecordInfo(CAppData::m_iLoginID, m_iMediaSessionID, pDownloadInfo, uiBufLen);if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent Subject: Download

Record

Parent Subject: Interface Function Definition

Start to Record DPSDK_StartRecord

Name	Note
Description：	Start to record.
OS：	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1 (2.6.16.21 above) 64 Bits. • SUSE Linux 10 32 Bits. • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits.

Function：	`DPSDK_INT32 DPSDK_StartRecord(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_CHAR* pFile,`
	`    DPSDK_UINT32 uiSplitRecordLen`
	`);`

Parameter：	iSessionID
	  [in] User Session ID.
	iMediaSessionID
	  [in] Media Session ID.
	pFile
	  [in] Record File Name with Full Path.
	uiSplitRecordLen
	  [in] Split Record Length.

Returned Value：	Success returns 0 and failure returns error_code.

Sample：	`DPSDK_CHAR szFile[DPSDK_FILE_PATH_LEN] = {0};`
	`strcpy(szFile, "E:\\Download.dav");`
	`DPSDK_UINT32 uiSplitRecordLen = 10;`
	`DPSDK_INT32 iRet = DPSDK_StartRecord(iSessionID, iMediaSessionID, szFile, uiSplitRecordLen);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent Subject：Record

Stop Recording DPSDK_StopRecord

Name	Note
Descriptin :	Stop recording.
OS :	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits • SUSE Linux 11 SP1(2.6.16.21 above) 64 Bits • SUSE Linux 10 32 Bits • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits

Function :	DPSDK_INT32 DPSDK_StopRecord(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_FILE_STORE_LIST* pRecordFile,
	    DPSDK_UINT32 uiBufLen
	);

Parameter :	iSessionID [in] User Session ID iMediaSessionID [in] Media Session ID pRecordFile [out] Generated Record File List. NULL means no need to get record result. uiBufLen [in] Buffer Size
Returned value :	Success returns 0 and failure returns error code.

Sample :	DPSDK_SIZET ulBufSize = 10;
	DPSDK_UINT32 uiBufLen = sizeof(DPSDK_FILE_STORE_LIST) + (ulBufSize-1) * sizeof(DPSDK_FILE_STORE_INFO);
	DPSDK_FILE_STORE_LIST* pRecordFile = (DPSDK_FILE_STORE_LIST*)malloc(uiBufLen);
	memset(pRecordFile, 0, uiBufLen);
	DPSDK_INT32 iRet = DPSDK_StopRecord(iSessionID, iMediaSessionID, pRecordFile, uiBufLen);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}
	free(pRecordFile);
	pRecordFile = NULL;

Parent Subject : Record

Taking a Video or not DPSDK_IsRecordState

Name	Note
Descriptin：	Judge if the video is being taken.
OS：	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits • SUSE Linux 11 SP1(2.6.16.21 above) 64 Bits • SUSE Linux 10 32 Bits • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits

Function：	DPSDK_INT32 DPSDK_IsRecordState(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_BOOL* pIsRecord
	);

Parameter：	iSessionID
	  [in] User Session ID
	iMediaSessionID
	  [in] Media Session ID
	pIsRecord
	  [out] true: Taking a Video, false: not Taking a Video.

Returned Value：	Success returns 0 and failure returns error code.

Sample：	DPSDK_BOOL bIsRecord = false;
	DPSDK_INT32 iRet = DPSDK_IsRecordState(iSessionID, iMediaSessionID, &bIsRecord);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	    if(bIsRecord == true)
	    {
	        //Taking a Video
	    }
	}

Parent Subject： Record

Set Split Record Length DPSDK_SetSplitRecordLen

Name	**Note**
Description :	Set split record length.
OS :	• Windows 7 Professional 32 Bits, Windows Server 2008 R2 64 Bits. • SUSE Linux 11 SP1(2.6.16.21 above) 64 Bits • SUSE Linux 10 32 Bits • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 Bits

Function :	`DPSDK_INT32 DPSDK_SetSplitRecordLen(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_UINT32 uiSplitRecordLen`
	`);`

Parameter :	iSessionID
	  [in] User Session ID
	iMediaSessionID
	  [in] Media Session ID
	uiSplitRecordLen
	  [in] Split Record Length. Unit: M

Returned Value :	Success returns 0 and failure returns error_code.

Sample :	`DPSDK_UINT32 uiSplitRecordLen = 10;`
	`DPSDK_INT32 iRet = DPSDK_SetSplitRecordLen(iSessionID, iMediaSessionID, uiSplitRecordLen);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent Subject: Record

Screenshot

Parent Subject: Interface Function Definition

Save the picture into the file DPSDK_Get24BitPictureFile

Name	Note
Description:	Snapshot, save the picture into the file.
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32位 • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_Get24BitPictureFile(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_UINT32 uiPicFormat,`
	`    DPSDK_CHAR* pPath`
	`);`

Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID uiPicFormat [in] Picture format, refer to DPSDK_PIC_FORMAT pPath [in] File path where picture data is saved.
Returned value:	Success return 0, failure return Error code。

Samples:	`DPSDK_UINT32 uiPicFormat = DPSDK_PIC_FORMAT_BMP; // BMP type`
	`DPSDK_CHAR szPath[DPSDK_FILE_PATH_LEN] = {0};`
	`strcpy(szPath, "D:\\ Download\\file.bmp");//File path where picture data is saved`
	`DPSDK_INT32 iRet = DPSDK_Get24BitPictureFile(iSessionID, iMediaSessionID, uiPicFormat, szPath)`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Snapshot

Save to buffer address of image data DPSDK_GetPictureBuf

Name	Note
Description:	Snapshot, save it to the buffer address of image data.
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	DPSDK_INT32 DPSDK_GetPictureBuf(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_CHAR* pPicBuf,
	    DPSDK_INT32 iBufsize,
	    DPSDK_INT32* pPicSize,
	    DPSDK_UINT32 uiPicFormat
	);

Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID pPicBuf [out] Save it to buffer address of image data, allocated by users, no less than image size iBufsize [in] Buffer zone size pPicSize [out] Get the actual image size uiPicFormat [in] Picture format, refer to DPSDK_PIC_FORMAT
Returned value:	Success return 0, failure return Error code.

Samples:	DPSDK_INT32 iWidth = 0;
	DPSDK_INT32 iHeight = 0;
	DPSDK_UINT32 uiPicFormat = DPSDK_PIC_FORMAT_BMP; // BMP type
	DPSDK_GetPictureSize(iSessionID, iMediaSessionID, &iWidth, &iHeight);
	DPSDK_LONG lBufSize = GetPicBuffSize(uiPicFormat, iHeight, iWidth);
	DPSDK_CHAR* pPicBuf = new DPSDK_CHAR[lBufSize];
	memset(pPicBuf, 0, lBufSize);
	DPSDK_INT32 iFactPicSize = -1;
	DPSDK_INT32 iRet = DPSDK_GetPictureBuf(iSessionID, iMediaSessionID, pPicBuf, lBufSize, &iFactPicSize, uiPicFormat);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}
	delete []pPicBuf;

	pPicBuf = NULLL;

Parent subject: Snapshot

Data conversion bmp format picture DPSDK_ConvertToBmpFile

Name	**Note**
Description:	Convert the picture data to the bmp format picture
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	DPSDK_INT32 DPSDK_ConvertToBmpFile(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_CONVERT_BMP* pConvertBMP
	);

Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID pConvertBMP [in] Picture data which needs to be converted, refer to DPSDK_CONVERT_BMP
Returned value:	Success return 0, failure return Error_code.

Samples:	DPSDK_CONVERT_BMP struConvertBMP;
	struConvertBMP.pBuf = m_struPictureData.pBuf;
	struConvertBMP.lSize = m_struPictureData.lSize;
	struConvertBMP.lHeight = m_struPictureData.lHeight;
	struConvertBMP.lWidth = m_struPictureData.lWidth;
	struConvertBMP.lType = m_struPictureData.lType;
	strcpy(struConvertBMP.szFileName, "D:\\file.bmp");
	DPSDK_INT32 iRet = DPSDK_ConvertToBmpFile(iSessionID, iMediaSessionID, &struConvertBMP);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent subject: Snapshot

Fisheye

Parent Subject: Interface Function Definition

Enable Fisheye DPSDK_StartFisheyeEx

Name	Note
Description:	Enable fisheye
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit
Function:	DPSDK_INT32 DPSDK_StartFisheyeEx(DPSDK_INT32 iSessionID, DPSDK_INT32 iMediaSessionID, DPSDK_UINT32 uiFishType, DPSDK_MHFPTZ_INIT_PARAM* pPtzChannelParam);
Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID uiFishType [in] Fisheye enable type, refer to DPSDK_FISH_TYPE definition pPtzChannelParam [in] Smart track (Fisheye + PTZ) initialization channel parameter
Returned value:	Success return 0, failure return Error code.

Samples:	DPSDK_UINT32 uiFishTyp = 1; // 1 means smart tracking (fisheye + PTZ) dewarping
	DPSDK_MHFPTZ_INIT_PARAM struParam;
	memset(&struParam, 0, sizeof(struParam));
	struParam.iHimgWidth = 1280;
	struParam.iHimgHeight = 720;
	struParam.iZoomListSize = 4;
	DPSDK_INT32 iRet = DPSDK_StartFisheyeEx(iSessionID, iMediaSessionID, uiFishTyp, &struParam);
	if(iRet == DPSDK_SUCCESS)
	{
	//Success
	}

Parent subject: Fisheye

Disable Fisheye DPSDK_CloseFisheye

Name	Note
Description:	Disable fisheye
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_CloseFisheye(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	iMediaSessionID
	  [in] Media session ID

Returned value:	Success return 0, failure return Error Code.

Samples:	`DPSDK_INT32 iRet = DPSDK_ CloseFisheye(iSessionID, iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Fisheye

Fisheye parameter initialization DPSDK_InitFisheyeOptParam

Name	Note
Description:	Fisheye parameter initialization
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_InitFisheyeOptParam(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_FISH_OPTPARAM* pOptParam`
	`);`

Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID pOptParam [out] Fisheye parameter
Returned value:	Success return 0, failure return Error_code.

Samples:	`DPSDK_FISH_OPTPARAM struParam;`
	`memset(&struParam, 0, sizeof(struParam));`
	`DPSDK_INT32 iRet = DPSDK_InitFisheyeOptParam(iSessionID, iMediaSessionID, &struParam);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Fisheye

Update Fisheye Parameters DPSDK_UpdateFisheyeOptParam

Name	Note
Description:	Update fisheye parameters
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	DPSDK_INT32 DPSDK_UpdateFisheyeOptParam(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_FISH_UPDATE_PARAM* iMediaSessionID
	);

Parameters:	iSessionID
	  [in] User session ID
	iMediaSessionID
	  [in] Media session ID
	iMediaSessionID
	  [in] Fisheye parameters

Returned value:	Success return 0, failure return Error_code.

Samples:	DPSDK_FISH_UPDATE_PARAM struParam;
	memset(&struParam, 0, sizeof(struParam));
	struParam.iCircleX = 1; //Input X coordinate of fisheye circle center in the image
	struParam.iCircleY = 1; //Input Y coordinate of fisheye circle center in the image
	struParam.iRadius = 2; //Input fisheye radius in the image
	struParam.lHeightRatio = 720; // Original width of corresponding main stream
	struParam.lWidthRatio = 1280; // Original height of corresponding main stream
	DPSDK_INT32 iRet = DPSDK_UpdateFisheyeOptParam(iSessionID, iMediaSessionID, &struParam);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent subject: Fisheye

Get Fisheye Parameters DPSDK_GetFisheyeInfo

Name	Note
Description:	It is to get fisheye parameters
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_GetFisheyeInfo(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_FISH_OPTPARAM* pOptParam`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	iMediaSessionID
	  [in] Media session ID
	pOptParam
	  [out] Fisheye parameters

Returned value:	Success return 0, failure return Error code.

Samples:	`DPSDK_FISH_OPTPARAM struResult;`
	`memset(&struResult, 0, sizeof(struResult));`
	`DPSDK_INT32 iRet = DPSDK_ GetFisheyeInfo(iSessionID, iMediaSessionID, &struResult);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success, get fisheye parameters`
	`}`

Parent subject: Fisheye

Set and get fisheye parameters DPSDK_SetFisheyeInfo

Name	**Note**
Description:	Set fisheye parameters
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit
Function:	DPSDK_INT32 DPSDK_SetFisheyeInfo(DPSDK_INT32 iSessionID, DPSDK_INT32 iMediaSessionID, DPSDK_FISH_OPTPARAM* pOptParam);
Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID pOptParam [in] Fisheye parameters
Returned value:	Success return 0, failure return Error code

Samples:	DPSDK_FISH_OPTPARAM struParam;
	memset(&struParam, 0, sizeof(struParam));
	struParam.uiMainMountMode = DPSDK_EMOUNT_MODE_CEIL; // Main mounting mode, 1 means ceiling mount
	struParam.uiMainCalibrateMode = DPSDK_SHOW_MODE_ORIGINAL; // Image main calibration mode, 2 means original mode (square), with zoom ratio
	DPSDK_INT32 iRet = DPSDK_ SetFisheyeInfo(iSessionID, iMediaSessionID, &struParam);
	if(iRet == DPSDK_SUCCESS)
	{
	//Success
	}

Parent subject: Fisheye

Enable or disable the second fisheye window in floating mode DPSDK_ShowFisheyeSecondRegion

Name	**Note**
Description:	Enable or disable the second fisheye window in floating mode
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit
Function:	DPSDK_INT32 DPSDK_ShowFisheyeSecondRegion(DPSDK_INT32 iSessionID, DPSDK_INT32 iMediaSessionID, HCWND hDestWnd, DPSDK_FISH_OPTPARAM* pOptParam, DPSDK_BOOL bEnable);
Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID hDestWnd [in] Window handle pOptParam [in] Fisheye parameters B Enable [in] Enable or disable second window in floating mode
Returned value:	Success return 0, failure return Error code.

Samples:	HCWND hDestWnd;
	DPSDK_BOOL bEnable = 1; //Enable the second window in floating mode
	DPSDK_FISH_OPTPARAM struParam;
	memset(&struParam, 0, sizeof(struParam));
	struParam.uiMainMountMode = DPSDK_EMOUNT_MODE_CEIL; // Main mounting mode, 1 means ceiling mount
	struParam.uiMainCalibrateMode = DPSDK_SHOW_MODE_ORIGINAL; // Image main calibration mode, 2 means original mode(square), with zoom ratio
	DPSDK_INT32 iRet = DPSDK_ShowFisheyeSecondRegion(iSessionID, iMediaSessionID, hDestWnd, &struParam, bEnable);
	if(iRet == DPSDK_SUCCESS)
	{
	//Success
	}

Parent subject: Fisheye

Control Fisheye Device DPSDK_ControlFishEye

Name	Note
Description:	It is to control fisheye device to make zoom/ get fisheye and PTZ info
OS:	• Windows 7 Professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit
Function:	DPSDK_INT32 DPSDK_ControlFishEye(DPSDK_INT32 iSessionID, DPSDK_INT32 iMediaSessionID, DPSDK_FISH_EPTZPARAM* pFishBasePtzInfo);
Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID pFishBasePtzInfo [in out] ePTZ zoom option
Returned value:	Success return 0, failure return Error code.

Samples:	DPSDK_FISH_EPTZPARAM struParam;
	memset(&struParam, 0, sizeof(struParam));
	struParam.uiPtzCmd = DPSDK_EPTZ_CMD_ZOOM_IN; // PTZ operation, 1 means zoom in
	DPSDK_INT32 iRet = DPSDK_ControlFishEye(iSessionID, iMediaSessionID, &struParam);
	if(iRet == DPSDK_SUCCESS)
	{
	//Success
	}

Parent subject: Fisheye

Get Fisheye PTZ Parameters DPSDK_GetFishEyePtzInfo

Name	Note
Description:	It is to get fisheye PTZ parameters
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_GetFishEyePtzInfo(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_FISH_EPTZPARAM* pEptzFrameInfo,`
	`    DPSDK_BOOL bSecondRegion`
	`);`

Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID PEptzFrameInfo [in out] Fisheye PTZ relevant parameters storage structure bSecondRegion [in] Input 1 in the second window in the floating mode (It is 0 by default)
Returned value:	Success return 0, failure return Error code.

Samples:	`DPSDK_FISH_EPTZPARAM struParam;`
	`memset(&struParam, 0, sizeof(struParam));`
	`struParam.uiPtzCmd = DPSDK_EPTZ_CMD_ZOOM_IN;    //ePTZ movement option, 1 means zoom in`
	`DPSDK_BOOL bSecondRegion = 1;    //Input 1 in the second window in floating mode`
	`DPSDK_INT32 iRet = DPSDK_GetFishEyePtzInfo(iSessionID, iMediaSessionID, &struParam, bSecondRegion);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Fisheye

Set Fisheye Parameters DPSDK_SetFisheyeParams

Name	Note
Description:	Set fisheye parameters
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	DPSDK_INT32 DPSDK_SetFisheyeParams(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_FISH_PARAMS* pFishParams
	);

Parameters:	iSessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID
	PFishParams
	  [in] Fisheye parameters

Returned value:	Success return 0, failure return Error_code.

Samples:	DPSDK_FISH_PARAMS struParam;
	memset(&struParam, 0, sizeof(struParam));
	struParam.struSubCamConfigParam.uiHCamType = DPSDK_IPCTYPE_FE; //Fisheye
	DPSDK_INT32 iRet = DPSDK_SetFisheyeParams(iSessionID, iMediaSessionID, &struParam);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent subject: Fisheye

Get Fisheye Parameters DPSDK_GetFisheyeParams

Name	Note
Description:	Get fisheye parameters
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	DPSDK_INT32 DPSDK_GetFisheyeParams(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_FISH_OPTPARAM* pFishOptParamBase
	);

Parameters:	iSessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID
	PFishOptParamBase
	   [in out] Fisheye parameters

Returned value:	Success return 0, failure return error_code.

Samples:	DPSDK_FISH_OPTPARAM struParam;
	memset(&struParam, 0, sizeof(struParam));
	struParam.uiMainMountMode = DPSDK_EMOUNT_MODE_CEIL; // Main mounting mode, 1 means ceiling mount
	struParam.uiMainCalibrateMode = DPSDK_SHOW_MODE_ORIGINAL; // Image main calibration mode, 2 means original mode(square), with zoom ratio
	DPSDK_INT32 iRet = DPSDK_GetFisheyeParams(iSessionID, iMediaSessionID, &struParam);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent subject: Fisheye

Video Base Interface

Parent Subject: Interface Function Definition

Enable Hardware Acceleration DPSDK_SetDecode

Name	Note
Description:	Enable hardware acceleration
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	DPSDK_INT32 DPSDK_SetDecode(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_INT32 iDecodeType
	);

Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID iDecodeType [in] Decode type, refer to DPSDK_DECODE_TYPE definition
Returned value:	Success return 0, failure return Error code.

Samples:	DPSDK_INT32 iDecodeType = DPSDK_DECODE_SW; //Decode type, DPSDK_DECODE_SW means CPU decode
	DPSDK_INT32 iRet = DPSDK_SetDecode（iSessionID, iMediaSessionID, iDecodeType）;
	if(iRet == DPSDK_SUCCESS)
	{
	    //成功
	}

Parent subject: Video Basic Port

If it is to enable HD picture internal adjustment DPSDK_EnableLargePicAdjustment

Name	Note
Description:	If it is to enable HD image internal adjustment
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_EnableLargePicAdjustment(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_BOOL bEnable`
	`);`

Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID bEnable [in] Function sign true enable false disable
Returned value:	Success return 0, failure return Error code

Samples:	`DPSDK_INT32 iRet = DPSDK_EnableLargePicAdjustment(iSessionID, iMediaSessionID, true);//Enable function sign`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Video Basic Port

Get Original Picture Size DPSDK_GetPictureSize

Name	Note
Description:	Get original picture size
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_GetPictureSize(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_INT32* pWidth,`
	`    DPSDK_INT32* pHeight`
	`);`

Parameter:	iSessionID
	  [in] User session ID
	iMediaSessionID
	  [in] Media session ID
	pWidth
	  [out] Width
	pHeight
	  [out] Height

Returned value:	Success return 0, failure return Error code

Samples:	`DPSDK_INT32 iWidth = -1;`
	`DPSDK_INT32 iHeight = -1;`
	`DPSDK_INT32 iRet = DPSDK_GetPictureSize(iSessionID, iSessionID, &iWidth, &iHeight);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success, get original picture size`
	`}`

Parent subject: Video Basic Port

Get Volume DPSDK_GetVolume

Name	Note
Description:	Get volume
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_GetVolume(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_UINT32* pVolume`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	iMediaSessionID
	  [in] Media session ID
	PVolume
	  [out] Volume

Returned value:	Success return 0, failure return Error_code

Samples:	`DPSDK_UINT32 uiVolume = -1;`
	`DPSDK_INT32 iRet = DPSDK_GetVolume(iSessionID, iMediaSessionID, &uiVolume);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Video Basic Port

Set Volume DPSDK_SetVolume

Name	Note
Description:	Set volume
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_SetVolume(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_UINT32 uiVolume`
	`);`

Parameters:	iSessionID [in] User session ID IMediaSessionID [in] Media session ID uiVolume [in] Volume
Returned value:	Success return 0, failure return Error code.

Samples:	`DPSDK_UINT32 uiVolume = 2;`
	`DPSDK_INT32 iRet = DPSDK_SetVolume(iSessionID, iMediaSessionID, uiVolume);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Video Basic Port

Open Sound DPSDK_OpenSound

Name	Note
Description:	Open sound via private mode
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_OpenSound(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID

Returned value:	Success return 0, failure return Error_code.

Samples:	`DPSDK_INT32 iRet = DPSDK_OpenSound(iSessionID, iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Video Basic Port

Close Sound DPSDK_CloseSound

Name	Note
Description:	Close sound
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_CloseSound(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID`
	`);`

Parameter:	iSessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID

Returned value:	Success return 0, failure return Error code.

Samples:	`DPSDK_INT32 iRet = DPSDK_CloseSound(iSessionID, iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Video Basic Port

Open Sound via Sharing Mode DPSDK_OpenSoundShare

Name	Note
Description:	Open sound via sharing mode
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_OpenSoundShare(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID

Returned value:	Success return 0, failure return Error_code.

Samples:	`DPSDK_INT32 iRet = DPSDK_OpenSoundShare(iSessionID, iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Video Basic Port

Close Sound of Sharing Mode DPSDK_CloseSoundShare

Name	Note
Description:	Close sound of sharing mode
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_CloseSoundShare(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID`
	`);`

Parameters:	ISessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID

Returned value:	Success return 0, failure return Error code 。

Samples:	`DPSDK_INT32 iRet = DPSDK_CloseSoundShare(iSessionID, iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Video Basic Port

Sound Enable Status DPSDK_IsOpenSoundState

Name	Note
Description:	Sound enable status
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_IsOpenSoundState(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_BOOL* pIsOpenSound`
	`);`

Parameters:	iSessionID [in] User session ID IMediaSessionID [in] Media session ID pIsOpenSound [out] true: enable, false: disabled
Returned value:	Success return 0, failure return Error code

Samples:	`DPSDK_BOOL bIsOpen = false;`
	`DPSDK_INT32 iRet = DPSDK_IsOpenSoundState(iSessionID, iMediaSessionID, &bIsOpen);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`    if(bIsOpen == true)`
	`    {`
	`        //Sound enable status`
	`    }`
	`}`

Parent subject: Video Basic Port

Set Video Parameters DPSDK_SetColor

Name	Note
Description:	Set video parameters
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_SetColor(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_INT32 iBrightness,`
	`    DPSDK_INT32 iContrast,`
	`    DPSDK_INT32 iSaturation,`
	`    DPSDK_INT32 iHue`
	`);`

Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID IBrightness [in] Brightness Default: 64 Range 0-128 iContrast [in] Contrast Default 64: Range 0-128 iSaturation [in] Saturation Default 64: Range 0-128 iHue [in] Hue Default 64: Range 0-128
Returned Value:	Success return 0, failure return Error_Code

Samples:	`DPSDK_INT32 iBrightness = 64;`
	`DPSDK_INT32 iContrast = 64;`
	`DPSDK_INT32 iSaturation = 64;`
	`DPSDK_INT32 iHue = 64;`
	`DPSDK_INT32 iRet = DPSDK_SetColor(iSessionID, iMediaSessionID, iBrightness,`
	`        iContrast, iSaturation, iHue);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Video Basic Port

Get Video Parameters DPSDK_GetColor

Name	Note
Description:	Get video parameters
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_GetColor(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_INT32* pBrightness,`
	`    DPSDK_INT32* pContrast,`
	`    DPSDK_INT32* pSaturation,`
	`    DPSDK_INT32* pHue`
	`);`

Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID pBrightness [out] Brightness Default 64: Range 0-128 pContrast [out] Contrast Default 64: Range 0-128 pSaturation [out] Saturation Default 64: Range 0-128 pHue [out] Hue Default 64: Range 0-128
Returned value:	Success return 0, failure return Error Code.

Samples:	`DPSDK_INT32 iBrightness = -1;`
	`DPSDK_INT32 iContrast = -1;`
	`DPSDK_INT32 iSaturation = -1;`
	`DPSDK_INT32 iHue = -1;`
	`DPSDK_INT32 iRet = DPSDK_GetColor(iSessionID, iMediaSessionID, &iBrightness, &iContrast, &iSaturation, &iHue);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Video Basic Port

Set or add display region, regional amplified display DPSDK_SetDisplayRegion

Name	Note
Description：	It is to set or add display region, it can also make regional amplification display
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function：	DPSDK_INT32 DPSDK_SetDisplayRegion(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_RECT* pRECT,
	    HCWND hDestWnd,
	    DPSDK_BOOL bEnable
	);

Parameters：	iSessionID [in] User session ID IMediaSessionID [in] Media session ID Prect [in] Regional display hDestWnd [in] Display window handle bEnable [in] Enable or disable display area true enable false disable
Returned value：	Success return 0, failure return Error code。

Samples：	DPSDK_RECT struRECT;
	memset(struRECT, 0, sizeof(struRECT));
	struRECT.left = 1;
	struRECT.right = 5;
	struRECT.top = 10;
	struRECT.bottom = 5;
	HCWND hDestWnd;
	DPSDK_INT32 iRet = DPSDK_SetDisplayRegion(iSessionID, iMediaSessionID, &struRECT, hDestWnd, true);//Open display region
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success,
	}

Parent subject：Video Basic Port

Enable video enhancement algorithm function DPSDK_StartIVSE

Name	Note
Description:	Enable video enhancement algorithm function, it needs to include IvseDll.dll library
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_StartIVSE(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID

Returned value:	Success return 0, failure return Error code.

Samples:	`DPSDK_INT32 iRet = DPSDK_StartIVSE(iSessionID, iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Video Basic Port

Stop video enhancement algorithm function DPSDK_StopIVSE

Name	**Note**
Description：	Stop video enhancement algorithm function
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function：	`DPSDK_INT32  DPSDK_StopIVSE(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32  iMediaSessionID`
	`);`

Parameters：	ISessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID

Returned value：	Success return 0, failure return Error code 。

Samples：	`DPSDK_INT32 iRet = DPSDK_StopIVSE(iSessionID, iMediaSessionID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    // Success`
	`}`

Parent subject：Video Basic Port

Set video enhancement parameters DPSDK_SetIVSE

Name	Note
Description:	It is to set video enhancement parameters, it can make several call to use several types of IVSE library.
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	DPSDK_INT32 DPSDK_SetIVSE(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_IVSE_INFO* pIVSEInfo,
	    DPSDK_BOOL bEnable
	);

Parameters:	ISessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID
	PIVSEInfo
	  [in] Video enhancement parameters
	bEnable
	  [in] Enable switch

Returned value:	Success return 0, failure return Error_code

Samples:	DPSDK_IVSE_INFO struIVSEInfo;
	memset(&struIVSEInfo, 0, sizeof(struIVSEInfo));
	struIVSEInfo.struRoi.iX =0;
	struIVSEInfo.struRoi.iY =0;
	struIVSEInfo.struRoi.iWidth = 4;
	struIVSEInfo.struRoi.iHeight = 4;
	struIVSEInfo.iMode = 1; //Video Mode
	struIVSEInfo.uiFuncType = DPSDK_IVSE_DEHAZE; // Defog
	DPSDK_BOOL bEnable = 1; //Enable
	DPSDK_INT32 iRet = DPSDK_SetIVSE(iSessionID, iMediaSessionID, &struIVSEInfo, bEnable);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent subject: Video Basic Port

Get current stream data length DPSDK_GetFrameDataLen

Name	**Note**
Description:	It is to get current stream data length
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit
Function:	DPSDK_INT32 DPSDK_GetFrameDataLen(DPSDK_INT32 iSessionID, DPSDK_INT32 iMediaSessionID, DPSDK_UINT32* pFrameDataLen);
Parameters:	iSessionID [in] User session ID IMediaSessionID [in] Media session ID PFrameDataLen [out] Stream data length
Returned value:	Success return 0, failure return Error code.

Samples:	DPSDK_UINT32 uiFrameDataLen = -1;
	DPSDK_INT32 iRet = DPSDK_GetBufferValue(iSessionID, iMediaSessionID, &uiFrameDataLen);
	if(iRet == DPSDK_SUCCESS)
	{
	//成功
	}

Parent subject: Video Basic Port

Get current frame time DPSDK_GetFrameTime

Name	Note
Description:	Get current frame time
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_GetFrameTime(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_TIMET* pFrameTime`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID
	PFrameTime
	  [out] Frame time

Returned value:	Success return 0, failure return error code 。

Samples:	`DPSDK_TIMET tFrameTime;`
	`DPSDK_INT32 iRet = DPSDK_GetFrameTime(iSessionID, iMediaSessionID, &tFrameTime);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Video Basic Port

Get video size from stream DPSDK_GetVideoFrameSize

Name	Note
Description:	Get video size from stream
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_GetVideoFrameSize(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_INT32* pWidth,`
	`    DPSDK_INT32* pHeight`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID
	PWidth
	  [out] Width
	pHeight
	  [out] Height

Returned value:	Success return 0, failure return Error_code.

Samples:	`DPSDK_INT32 iWidth = -1;`
	`DPSDK_INT32 iHeigth = -1;`
	`DPSDK_INT32 iRet = DPSDK_GetVideoFrameSize(iSessionID, iMediaSessionID, &iWidth, &iHeigth);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Video Basic Port

Get current frame number DPSDK_GetFrameNumber

Name	Note
Description:	Get current frame number
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	DPSDK_INT32 DPSDK_GetFrameNumber(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_INT32* pFrameNum
	);

Parameters:	iSessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID
	PFrameNum
	  [out] Frame number

Returned value:	Success return 0, failure return Error code.

Samples:	DPSDK_INT32 iFrameNum = -1;
	DPSDK_INT32 iRet = DPSDK_GetFrameNumber(iSessionID, iMediaSessionID, &iFrameNum);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent subject: Video Basic Port

Get remained data size from source buffer zone DPSDK_GetSourceBufferRemain

Name	Note
Description:	Get remained data size from source buffer zone
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	DPSDK_INT32 DPSDK_GetSourceBufferRemain(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_INT32* pBufferRemain
	);

Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID pBufferRemain [out] Remained data size of buffer zone, unit BYTE
Returned value:	Success return 0, failure return Error code.

Samples:	DPSDK_INT32 iBufferRemain = -1;
	DPSDK_INT32 iRet = DPSDK_GetSourceBufferRemain(iSessionID, iMediaSessionID, &iBufferRemain);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent subject : Video Basic Port

Get designated buffer zone size DPSDK_GetBufferValue

Name	Note
Description:	Get the designated buffer zone size
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_GetBufferValue(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_UINT32 uiVaxBufType,`
	`    DPSDK_INT32* pBufferSize`
	`);`

Parameters:	ISessionID [in] User session ID IMediaSessionID [in] Media session ID UiVaxBufType [in] Buffer type, refer to DPSDK_VAX_BUF_TYPE enumeration definition pBufferSize [out] Buffer zone size
Returned value:	Success return 0, failure return Error code.

Samples:	`DPSDK_UINT32 uiVaxBufType = 1; // Video source buffer`
	`DPSDK_INT32 iBufferSize = -1;`
	`iRet = DPSDK_GetBufferValue(iSessionID, iMediaSessionID, uiVaxBufType, &iBufferSize);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: Video Basic Port

Intelligent Interface

Parent Subject: Interface Function Definition

Set if it is to display intelligent info DPSDK_SetIvsShowFlag

Name	Note
Description:	Set if it is to display intelligent info
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_SetIvsShowFlag(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_UINT32 uiType,`
	`    DPSDK_BOOL bVisible`
	`);`

Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID uiType [in] enumType type, refer to DPSDK_IVS_VISIBLE definition bVisible [in] bVisible display sign, true display false not display
Returned value:	Success return 0, failure return Error_code.

Samples:	`DPSDK_UINT32 uiType = DPSDK_IVS_RULE_VISIBLE; // DPSDK_IVS_RULE_VISIBLE means rules;`
	`DPSDK_INT32 iRet = DPSDK_SetIvsShowFlag(iSessionID, iMediaSessionID, uiType, true);//Display signs`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent Subject: Intelligent Interface

Get target classification ID according to location DPSDK_GetIvsClassId

Name	Note
Description:	It is to get target classification ID according to location
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_GetIvsClassId(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_INT32 iPortX,`
	`    DPSDK_INT32 iPortY,`
	`    DPSDK_INT32* pClassID`
	`);`

Parameters:	ISessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID
	IPortX
	  [in] Point X value
	IPortY
	  [in] Point Y value
	PClassID
	  [out] Target classification id

Returned value:	Success return 0, failure return Error code.

Samples:	`DPSDK_INT32 iClassID;`
	`DPSDK_INT32 iPortX = 1;`
	`DPSDK_INT32 iPortY = 1;`
	`DPSDK_INT32 iRet = DPSDK_GetIvsClassId(iSessionID, iMediaSessionID, iPortX, iPortY, &iClassID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`      //Success`
	`}`

Parent subject: Intelligent Interface

Get designated target ID according to location DPSDK_GetIvsObjectId

Name	Note
Description:	Get designated target ID according to location
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	DPSDK_INT32 DPSDK_GetIvsObjectId(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_INT32 iPortX,
	    DPSDK_INT32 iPortY,
	    DPSDK_INT32* pObjectID
	);

Parameters:	ISessionID
	  [in] User session ID
	IMediaSessionID
	  [in] Media session ID
	IPortX
	  [in] Point X value
	IPortY
	  [in] Point Y value
	PObjectID
	  [out] Designated target id

Returned value:	Success return 0, failure return Error code

Samples:	DPSDK_INT32 iObjectID;
	DPSDK_INT32 iPortX = 1;
	DPSDK_INT32 iPortY = 1;
	DPSDK_INT32 iRet = DPSDK_GetIvsObjectId(iSessionID, iMediaSessionID, iPortX, iPortY, &iObjectID);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent subject: Intelligent Interface

Set designated target ID DPSDK_SetIvsObjectId

Name	Note
Description:	Set designated target id
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function	DPSDK_INT32 DPSDK_SetIvsObjectId(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iMediaSessionID,
	    DPSDK_INT32 iClassID,
	    DPSDK_INT32 iObjectID,
	    DPSDK_BOOL bAttach
	);

Parameters	iSessionID [in] User session ID iMediaSessionID [in] Media session ID iClassID [in] Classification id iObjectID [in] Designated target id bAttach [in] If it is to draw tracking shape and color object
Returned value:	Success return 0, failure return Error_code.

Samples:	DPSDK_INT32 iClassID;
	DPSDK_INT32 iObjectID;
	DPSDK_BOOL bAttach = true;// draw the tracking shape and color object
	DPSDK_INT32 iRet = DPSDK_SetIvsObjectId(iSessionID, iMediaSessionID, iClassID, iObjectID, bAttach);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success, it is to set designated target id
	}

Parent subject: Intelligent Interface

Get In & Out People Number DPSDK_GetIvsPCInOutValue

Name	Note
Description:	Get in and out people number
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_GetIvsPCInOutValue(`
	`    DPSDK_INT32  iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_INT32* pInValue,`
	`    DPSDK_INT32* pOutValue`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	iMediaSessionID
	  [in] Media session ID
	pInValue
	  [out] In
	pOutValue
	  [out] Out

Returned value:	Success return 0, failure return Error code.

Samples:	`DPSDK_INT32 iInValue = 0, iOutValue =  0;`
	`DPSDK_INT32 iRet = DPSDK_GetIvsPCInOutValue(iSessionID, iMediaSessionID, &iInValue, &iOutValue);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //成功`
	`}`

Parent subject: Intelligent Interface

Split Screen Interface

Parent Subject: Interface Function Definition

Splicing algorithm DPSDK_SplitProc

Name	**Note**
Description:	Splicing algorithm. It is for 4K video wall display.
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	`DPSDK_INT32DPSDK_SplitProc(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_UINT32 iMode`
	`);`

Parameters:	iSessionID [in] User session ID iMediaSessionID [in] Media session ID iMode [in] Mode, 0=general mode, 1=1+3 mode, 2=1+5 mode and so on, refer to DPSDK_SPLIT_TRECE_TYPE
Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	`DPSDK_UINT32 iMode = DPSDK_SPLIT_ORG; //General mode`
	`DPSDK_INT32iRet = DPSDK_SplitProc(iSessionID, iMediaSessionID, iMode);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Successful`
	`}`

Parent Subject: Splicing screen interface

Splicing algorithm DPSDK_SplitProcUpdate

Name	Note
Description:	Splicing algorithm, refresh the rectangle area to be zoomed in.
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	`DPSDK_INT32 DPSDK_SplitProcUpdate(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMediaSessionID,`
	`    DPSDK_DISPLAY_RECT* pAreaRect`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	iMediaSessionID
	  [in] Media session ID
	pAreaRect
	  [in] Start address of the rectangle coordinates array
	    If nMode is 0, it is NULL;
	    If nMode is 1 or 4, the array is 3;
	    If nMode is 2, the array is 5;
	    If nMode is 5, the array is 6

Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	`DPSDK_DISPLAY_RECT struAreaRect;`
	`memset(&struAreaRect, 0, sizeof(struAreaRect));`
	`struAreaRect.iTop = 10;`
	`struAreaRect.iBottom = 1;`
	`struAreaRect.iX = 5;`
	`struAreaRect.iY = 5;`
	`struAreaRect.iPicWidth = 3;`
	`struAreaRect.iPicHeight = 4;`
	`DPSDK_INT32iRet =DPSDK_SplitProcUpdate(iSessionID,iMediaSessionID, &struAreaRect);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    // Successful`
	`}`

Parent Subject: Splicing screen interface

Bayonet

Parent Subject: Interface Function Definition

Start ANPR image surveillance DPSDK_StartBayonetPicture

Name	Note
Description：	Start ANPR image surveillance
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function：	DPSDK_INT32DPSDK_StartBayonetPicture(
	    DPSDK_INT32 iSessionID,
	    DPSDK_PICTURE_MONITOR* pBayPicParam,
	    DPSDK_INT32* pMonitorSessionID
	);

Parameters：	ISessionID
	  [in] User session ID
	PBayPicParam
	  [in] ANPR image parameters
	PMonitorSessionID
	  [out] Monitor session ID

Returned Value：	Return 0 if succeeded, Return Error code if failed.

Samples：	DPSDK_INT32iMonitorSessionID = -1;
	DPSDK_PICTURE_MONITOR struBayPic;
	memset(&struBayPic, 0, sizeof(struBayPic));
	strcpy(struBayPic.szCodeId, "172.2.10.33");
	struBayPic.uiDataType = 1; // vehicle information
	struBayPic.uiStreamType = 1;//main stream
	DPSDK_INT32iRet = DPSDK_StartBayonetPicture(iSessionID, &struBayPic,&iMonitorSessionID);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Successful
	}

Parent Subject： ANPR

Stop ANPR image surveillance DPSDK_StopBayonetPicture

Name	Note
Description：	Stop ANPR image surveillance
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function：	`DPSDK_INT32 DPSDK_StopBayonetPicture(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iMonitorSessionID`
	`);`

Parameters：	ISessionID [in] User session ID IMonitorSessionID [in] Monitor session ID
Returned Value：	Return 0 if succeeded， Return Error code if failed.

Samples：	`DPSDK_INT32 iRet = DPSDK_StopBayonetPicture(iSessionID, iMonitorSessionID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    // Successful`
	`}`

Parent Subject： ANPR

Getting ANPR image surveillance data DPSDK_GetBayonetDictionary

Name	Note
Description:	Getting ANPR image surveillance dictionary data
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	DPSDK_INT32 DPSDK_GetBayonetDictionary(
	    DPSDK_INT32 iSessionID,
	    DPSDK_CHAR* pLanguage,
	    DPSDK_INT32 iDictionaryType,
	    DPSDK_CHAR** pDicListXml
	);

Parameters:	ISessionID [in] User session ID PLanguage [in] Language, such as : zh_CN IDictionaryType [in] Dictionary type,refer to DPSDK_BAYONET_DICTIONARY_TYPE for definition. PDicListXml [out] Dictionary data list xml
Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	DPSDK_CHAR szLanguage[DPSDK_ALARM_LANGUAGE_LEN] = {0};
	strcpy(szLanguage, "zh_CN");
	DPSDK_INT32 iDictionaryType = DPSDK_LANE_NUMBER; //ANPR surveillance dictionary type. 19 is the lane number.
	DPSDK_CHAR* pDicListXml = NULL;
	DPSDK_INT32 iRet = DPSDK_GetBayonetDictionary(iSessionID, &szLanguage, iDictionaryType, &pDicListXml);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Successful, getting ANPR image surveillance dictionary data
	}
	DPSDK_ReleaseDataBuffer(pDicListXml);

Parent Subject: ANPR

Alarm

Parent Subject: Interface Function Definition

Confirm alarm DPSDK_ConfirmAlarm

Name	**Note**
Description:	Confirm alarm
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	`DPSDK_INT32DPSDK_ConfirmAlarm(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_CONFIRMALARM_PARAM* pConfirmAlarmParam`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	pConfirmAlarmParam
	  [in] Alarm confirm parameters

Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	`DPSDK_CONFIRMALARM_PARAM struParam;`
	`memset(&struParam, 0, sizeof(struParam));`
	`struParam.iHandleStatus = DEALWITH_PENDING; //Processing`
	`struParam.uiEmailRevceiverNumber = 5;`
	`for(unsigned inti=0; i<struParam.uiEmailRevceiverNumber; i++)`
	`{`
	`    strcpy(struParam.struEmailReceiverList[i].szEmailAddr, "http://");// E-mail address`
	`}`
	`DPSDK_INT32iRet = DPSDK_ConfirmAlarm(iSessionID, &struParam);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Successful. Confirm alarm`
	`}`

Parent Subject: Alarm

Search alarm DPSDK_QueryAlarm

Name	Note
Description:	Search alarm
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	DPSDK_INT32 DPSDK_QueryAlarm(
	    DPSDK_INT32 iSessionID,
	    DPSDK_QUERYALARM_PARAM* pQueryAlarmParam,
	    DPSDK_UINT32 uiBufLen,
	    DPSDK_ALARM_DETAILINFO_LIST* pAlarmDetailInfoList
	);

Parameters:	ISessionID [in] User session ID PQueryAlarmParam [in] Alarm search parameters UiBufLen [in] Alarm list buffer size PAlarmDetailInfoList [out] Alarm list
Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	DPSDK_QUERYALARM_PARAM struParam;
	memset(&struParam, 0, sizeof(struParam));
	struParam.pAlarmType = ALARM_TYPE_VIDEO_LOST; //Video loss
	struParam.pAlarmGrade = ALARM_LEVEL_ONE; //Alarm level
	struParam.pAlarmStatus = ALARM_EVENT_OCCUR; //Alarm occurs
	struParam.pHandleStatus= DEALWITH_PENDING; //Processing alarm
	// Calculate Output size
	DPSDK_UINT32UiBufLen = sizeof(DPSDK_ALARM_DETAILINFO_LIST) + sizeof(DPSDK_ALARM_DETAILINFO)*(struParam.iPageSize-1);
	DPSDK_ALARM_DETAILINFO_LIST* pAlarmDetailInfoList = (DPSDK_ALARM_DETAILINFO_LIST*)malloc(UiBufLen);
	memset(pAlarmDetailInfoList, 0, UiBufLen);
	DPSDK_INT32 iRet = DPSDK_QueryAlarm(iSessionID, &struParam, UiBufLen, pAlarmDetailInfoList);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Successful. Search alarm
	}
	free(pAlarmDetailInfoList);
	pAlarmDetailInfoList = NULL;

Parent Subject : Alarm

Search alarm total amount DPSDK_QueryAlarmCount

Name	Note
Description:	Search alarm total amount
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	DPSDK_INT32 DPSDK_QueryAlarmCount(
	    DPSDK_INT32 iSessionID,
	    DPSDK_QUERYALARMCOUNT_PARAM* pQueryAlarmCountParam,
	    DPSDK_UINT32* pAlarmCount
	);

Parameters:	ISessionID [in] User session ID PQueryAlarmCountParam [in] Alarm total amount search parameter PAlarmCount [out] Alarm total amount
Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	DPSDK_UINT32 uiAlarmCount = 0;
	DPSDK_QUERYALARMCOUNT_PARAM struQueryAlarmCountParam;
	memset(&struQueryAlarmCountParam, 0, sizeof(struQueryAlarmCountParam));
	struQueryAlarmCountParam.pAlarmType = ALARM_TYPE_VIDEO_LOST; //Video loss
	struQueryAlarmCountParam.pAlarmGrade = ALARM_LEVEL_ONE; //Alarm level
	struQueryAlarmCountParam.pAlarmStatus = ALARM_EVENT_OCCUR; //Alarm occurs
	struQueryAlarmCountParam.pHandleStatus= DEALWITH_PENDING; //Processing alarm
	struQueryAlarmCountParam.uiAlarmTypeNumber = 100; //Alarm type amount
	struQueryAlarmCountParam.uiAlarmGradeNumber = 6; //Alarm level amount
	struQueryAlarmCountParam.uiAlarmStatusNumber = 3; //Alarm status amount
	struQueryAlarmCountParam.uiHandleStatusNumber= 5; //Alarm processing status amount
	DPSDK_INT32 iRet = DPSDK_QueryAlarmCount(iSessionID,&struQueryAlarmCountParam,&uiAlarmCount);
	if(iRet == DPSDK_SUCCESS)
	{
	    // Successful.
	}

Parent Subject: Alarm

Search alarm processing flows DPSDK_QueryAlarmProcessFlow

Name	Note
Description:	Search alarm processing flows
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	DPSDK_INT32 DPSDK_QueryAlarmProcessFlow(
	    DPSDK_INT32 iSessionID,
	    DPSDK_CHAR* pAlarmCode,
	    DPSDK_UINT32 uiBufLen,
	    DPSDK_ALARMPROCESS_DETAILINFO_LIST* pAlarmProcessInfoList
	);

Parameters:	ISessionID [in] User session ID PAlarmCode [in] Alarm code UiBufLen [in] Alarm processing flow list buffer size PAlarmProcessInfoList [out] Alarm processing flow list
Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	DPSDK_CHARszAlarmCode[DPSDK_ALARM_ALARMCODE_LEN] = {0};
	strcpy(szAlarmCode, "0F8CF723-FC01-49C9-A5A2-97984E9E2548");
	DPSDK_UINT32 uiBufLen= sizeof(DPSDK_ALARMPROCESS_DETAILINFO_LIST) + sizeof(DPSDK_ALARMPROCESS_DETAILINFO)*2; // Just take 3 records
	DPSDK_ALARMPROCESS_DETAILINFO_LIST* pAlarmPFList = (DPSDK_ALARMPROCESS_DETAILINFO_LIST*)malloc(uiBufLen);
	Memset(pAlarmPFList,0,uiBufLen);
	DPSDK_INT32iRet = DPSDK_QueryAlarmProcessFlow(iSessionID,szAlarmCode,uiBufLen,pAlarmPFList);
	if(iRet == DPSDK_SUCCESS)
	{
	    // Successful.
	}
	free(pAlarmPFList);
	pAlarmPFList = NULL;

Parent Subject: Alarm

Block an alarm DPSDK_BlockAlarm

Name	Note
Description:	Block an alarm
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	`DPSDK_INT32 DPSDK_BlockAlarm(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_BLOCKALARM_PARAM* pBlockAlarmParam`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	pBlockAlarmParam
	  [in] Block alarm parameters

Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	`DPSDK_BLOCKALARM_PARAM struParam;`
	`memset(&struParam, 0, sizeof(struParam));`
	`struParam.iAlarmType = ALARM_TYPE_VIDEO_LOST;// Video loss`
	`struParam.iDuration = 10;`
	`strcpy(struParam.szAlarmCodeSource, "1000021");`
	`DPSDK_INT32 iRet = DPSDK_BlockAlarm(iSessionID, &struParam);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    // Successful.`
	`}`

Parent Subject: Alarm

Getting alarm type group information DPSDK_GetAlarmTypeGroupInfo

Name	Note
Description:	Getting alarm type group information
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	`DPSDK_INT32 DPSDK_GetAlarmTypeGroupInfo(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_CHAR* pLanguage,`
	`    DPSDK_CHAR** pInfoXml`
	`);`

Parameters:	ISessionID [in] User session ID PLanguage [in] Language PInfoXml [out] Alarm type group information Xml stream
Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	`DPSDK_CHARszLanguage[DPSDK_ALARM_LANGUAGE_LEN] = {0};`
	`DPSDK_CHAR* pInfoXml = NULL;`
	`strcpy(szLanguage, "en_us");`
	`DPSDK_INT32 iRet = DPSDK_GetAlarmTypeGroupInfo(iSessionID, szLanguage, &pInfoXml);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    // Successful.`
	`}`
	`DPSDK_ReleaseDataBuffer(pInfoXml);`

Parent Subject: Alarm

Export alarm DPSDK_ExportAlarms

Name	Note
Description :	Export alarm
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function :	DPSDK_INT32 DPSDK_ExportAlarms(
	    DPSDK_INT32 iSessionID,
	    DPSDK_ALARMEXPORT_PARAM* pAlarmExportParam,
	    DPSDK_INT32 iSessionId
	);

Parameters :	ISessionID
	  [in] User session ID
	PAlarmExportParam
	  [in] Alarm export parameters
	ISessionId
	  [in] Alarm export session ID

Returned Value :	Return 0 if succeeded, Return Error code if failed.

Samples :	DPSDK_ALARMEXPORT_PARAM struParam;
	memset(&struParam, 0, sizeof(struParam));
	struParam.pAlarmType = ALARM_TYPE_VIDEO_LOST; //Video loss
	struParam.pAlarmGrade = ALARM_LEVEL_ONE; //Alarm level
	struParam.pAlarmStatus = ALARM_EVENT_OCCUR; //Alarm occurs
	struParam.pHandleStatus= DEALWITH_PENDING; //Processing alarm
	DPSDK_INT32 iRet = DPSDK_ExportAlarms(iSessionID, &struParam, iSessionId);
	if(iRet == DPSDK_SUCCESS)
	{
	    // Successful.
	}
	free(struParam.pAlarmType);
	struParam.pAlarmType = NULL;
	free(struParam.pHandleStatus);
	struParam.pHandleStatus = NULL;
	free(struParam.pAlarmStatus);
	struParam.pAlarmStatus = NULL;
	free(struParam.pAlarmGrade);
	struParam.pAlarmGrade = NULL;

Parent Subject : Alarm

PTZ

Parent Subject: Interface Function Definition

PTZ Function Operation DPSDK_PtzOperateFunction

Name	Note
Description:	PTZ function operation
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_PtzOperateFunction(`
	`DPSDK_INT32 iSessionID,`
	`DPSDK_PTZOPERATE_FUNCTION_PARAM* pPtzOperateFunctionParam,`
	`DPSDK_PTZOPERATE_RESULT* pPtzOperateFunctionResult`
	`);`

Parameters:	iSessionID [in] User session ID pPtzOperateFunctionParam [in] PTZ function operation parameters pPtzOperateFunctionResult [out] Operation result
Returned value:	Success return 0, failure return Error_code.

Samples:	`DPSDK_PTZOPERATE_FUNCTION_PARAM struParam;`
	`DPSDK_PTZOPERATE_RESULT struResult;`
	`memset(&struParam, 0, sizeof(struParam));`
	`memset(&struResult, 0, sizeof(struResult));`
	`struParam.PtzOperateFunction_e = PtzOF_Show_PtzMenu; //Display “PTZ Menu”`
	`struParam.iSwitchMode = 1;     // Enable`
	`struParam.iBorderType = 16;     //Left limit`
	`struParam.iAssisentType= 23;    // BLC`
	`struParam.iMoveType = 25;       //Move upward`
	`struParam.iSwitchPtzMenu = 22; //Open PTZ menu`
	`DPSDK_INT32 iRet = DPSDK_PtzOperateFunction(iSessionID, &struParam, &struResult);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: PTZ

Operate PTZ Camera DPSDK_PtzOperateCamera

Name	Note
Description :	Operate PTZ camera
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit
Function :	DPSDK_INT32 DPSDK_PtzOperateCamera(DPSDK_INT32 iSessionID, DPSDK_PTZOPERATE_CAMERA_PARAM* pPtzOperateCamereParam, DPSDK_PTZOPERATE_RESULT* pPtzOperateResult);
Parameters :	iSessionID [in] User session ID pPtzOperateCamereParam [in] PTZ camera operation parameters pPtzOperateResult [out] Operation result
Returned value :	Success return 0, failure return Error_code.

Samples :	DPSDK_PTZOPERATE_CAMERA_PARAM struParam;
	DPSDK_PTZOPERATE_RESULT struResult;
	memset(&struParam, 0, sizeof(struParam));
	memset(&struResult, 0, sizeof(struResult));
	struParam.iDirect_e = 1; // Add
	struParam.iCommand = 1; // Enable
	struParam.iOperateType = 1; // Zoom
	struParam.iStep = 1; // Step
	DPSDK_INT32 iRet = DPSDK_PtzOperateCamera(iSessionID, &struParam, &struResult);
	if(iRet == DPSDK_SUCCESS)
	{
	//Success
	}

Parent subject : PTZ

PTZ Direction Control DPSDK_PtzOperateDirect

Name	Note
Description :	PTZ direction control
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function :	DPSDK_INT32 DPSDK_PtzOperateDirect(
	    DPSDK_INT32 iSessionID,
	    DPSDK_PTZOPERATE_DIRECT_PARAM* pPtzOperateDirectParam,
	    DPSDK_PTZOPERATE_RESULT* pPtzOperateResult
	);

Parameters :	iSessionID
	   [in] User session ID
	pPtzOperateDirectParam
	   [in] PTZ direction control parameters
	pPtzOperateResult
	   [out] Operation result

Returned value :	Success return 0, failure return Error_code.

Samples :	DPSDK_PTZOPERATE_DIRECT_PARAM struParam;
	DPSDK_PTZOPERATE_RESULT struResult;
	memset(&struParam, 0, sizeof(struParam));
	memset(&struResult, 0, sizeof(struResult));
	struParam.iStepY = 1; //Vertical step
	struParam.iStepX = 1; // Horizontal step
	struParam.iDirect = 1; //Upward
	struParam.iCommand = 0; // Stop
	DPSDK_INT32 iRet = DPSDK_PtzOperateDirect(iSessionID, &struParam, &struResult);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent subject : PTZ

Motorized Focus Control DPSDK_PtzOperateFocus

Name	Note
Description:	Motorized focus control
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_PtzOperateFocus(`
	`DPSDK_INT32 iSessionID,`
	`DPSDK_PTZOPERATE_FOCUS_PARAM* pPtzOperateFocusParam,`
	`DPSDK_PTZOPERATE_RESULT* pPtzOperateResult`
	`);`

Parameters:	ISessionID [in] User session PPtzOperateFocusParam [in] Motorized focus control parameters PPtzOperateResult [out] Operation results
Returned value:	Success return 0, failure return Error_code.

Samples:	`DPSDK_PTZOPERATE_FOCUS_PARAM struParam;`
	`DPSDK_PTZOPERATE_RESULT struResult;`
	`memset(&struParam, 0, sizeof(struParam));`
	`memset(&struResult, 0, sizeof(struResult));`
	`struParam.iOperateType = 1; // Continuous focus`
	`struParam.fZoom = 2;         // 2x`
	`struParam.Focus= 15;         //Focal length is 15`
	`DPSDK_INT32 iRet = DPSDK_PtzOperateFocus(iSessionID, &struParam, &struResult);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: PTZ

Control Preset DPSDK_PtzOperatePresetPoint

Name	Note
Description :	Control preset
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function :	DPSDK_INT32 DPSDK_PtzOperatePresetPoint(
	    DPSDK_INT32 iSessionID,
	    DPSDK_PTZOPERATE_PRESETPOINT_PARAM* pPtzOperatePrePointParam,
	    DPSDK_PTZOPERATE_RESULT* pPtzOperateResult
	);

Parameters :	iSessionID
	  [in] User session ID
	pPtzOperatePrePointParam
	  [in] Preset control parameters
	pPtzOperateResult
	  [out] Operation results

Returned value :	Success return 0, failure return Error_code

Samples :	DPSDK_PTZOPERATE_PRESETPOINT_PARAM struParam;
	DPSDK_PTZOPERATE_RESULT struResult;
	memset(&struParam, 0, sizeof(struParam));
	memset(&struResult, 0, sizeof(struResult));
	struResult.iOperateType = 1;// Positioning
	DPSDK_INT32 iRet = DPSDK_PtzOperatePresetPoint(iSessionID, &struParam, &struResult);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent subject : PTZ

3D Positioning DPSDK_PtzSitPosition

Name	Note
Description:	3D positioning
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	`DPSDK_INT32 DPSDK_PtzSitPosition(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_PTZOPERATE_SITPOSITION_PARAM* pPtzOperateSitPositionParam,`
	`    DPSDK_PTZOPERATE_RESULT* pPtzOperateResult`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	pPtzOperateSitPositionParam
	  [in] 3D positioning parameters
	pPtzOperateResult
	  [out] Operation results

Returned value:	Success return 0, failure return Error_code.

Samples:	`DPSDK_PTZOPERATE_SITPOSITION_PARAM struParam;`
	`DPSDK_PTZOPERATE_RESULT struResult;`
	`memset(&struParam, 0, sizeof(struParam));`
	`memset(&struResult, 0, sizeof(struResult));`
	`struParam.iPointX = 8000;// Horizontal coordinate is 8000`
	`struParam.iPointY = 8000;// Vertical coordinate is 8000`
	`struParam.iPointZ = 3;    // 3x zoom rate`
	`DPSDK_INT32 iRet = DPSDK_PtzSitPosition(iSessionID, &struParam, &struResult);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: PTZ

Lock, Unlock DPSDK_PtzArrangePtz

Name	Note
Description：	Lock, unlock
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32位 • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 位

Function：	`DPSDK_INT32 DPSDK_PtzArrangePtz(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_PTZOPERATE_ARRANGEPTZ_PARAM* pPtzOperateArrangePtzParam,`
	`    DPSDK_PTZOPERATE_RESULT* pPtzOperateResult`
	`);`

Parameters：	iSessionID
	  [in] User session ID
	pPtzOperateArrangePtzParam
	  [in] Lock, unlock parameters
	pPtzOperateResult
	  [out] Operation results

Returned value：	Success return 0, failure return Error code。

Samples：	`DPSDK_PTZOPERATE_ARRANGEPTZ_PARAM struParam;`
	`DPSDK_PTZOPERATE_RESULT struResult;`
	`memset(&struParam, 0, sizeof(struParam));`
	`memset(&struResult, 0, sizeof(struResult));`
	`struParam.iOperateType = 1; //1-lock current camera`
	`struParam.uiLockTime = 0;   //0 means it is locked all the time`
	`strcpy(struParam.szChannelId, "1000001$1$0$0");`
	`DPSDK_INT32 iRet = DPSDK_PtzArrangePtz(iSessionID, &struParam, &struResult);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject：PTZ

Alarm Output Control DPSDK_AlarmActionOut

Name	Note
Description:	Alarm output control
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function:	DPSDK_INT32 DPSDK_AlarmActionOut(
	    DPSDK_INT32 iSessionID,
	    DPSDK_PTZOPERATE_ALARMOUT_PARAM* pAlarmOutParam,
	   DPSDK_PTZOPERATE_RESULT* pPtzOperateResult
	);

Parameters:	iSessionID [in] User session ID pAlarmOutParam [in] Alarm output control parameters, refer to DPSDK_PTZOPERATE_ALARMOUT_PARAM structure for details pPtzOperateResult [out] Operation result, for exact parameters, please refer to DPSDK_PTZOPERATE_RESULT
Returned value:	Success return 0, failure return Error_code.

Samples:	DPSDK_PTZOPERATE_ALARMOUT_PARAM struParam;
	memset(&struParam, 0, sizeof(struParam));
	strcpy(struParam.szChannelId, "168383947B19V88R2VM0DOT");
	struParam.iOperateType = 1;
	struParam.iCommand = 1;
	
	DPSDK_PTZOPERATE_RESULT struResult;
	memset(&struResult, 0, sizeof(struResult));
	DPSDK_INT32 iRet = DPSDK_AlarmActionOut(CAppData::m_iLoginID, &struParam, &struResult);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Success
	}

Parent subject: PTZ

Preset Query DPSDK_PtzGetPresetPoints

Name	Note
Description :	Preset query
OS:	• Windows 7 professional version 32 bit, Windows Server 2008 R2 64 bit • SUSE Linux 11 SP1(2.6.16.21 or above) 64 bit • SUSE Linux 10 32 bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32 bit

Function :	`DPSDK_INT32 DPSDK_PtzGetPresetPoints(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_CHAR* pChannelId,`
	`    DPSDK_PTZ_PRESETPOINT_LIST* pPresetPointList,`
	`   DPSDK_UINT32 uiBufLen`
	`);`

Parameters :	iSessionID [in] User session ID pChannelId [in] Channel ID pPresetPointList [out] Preset info list, for exact parameters, refer to DPSDK_PTZ_PRESETPOINT_LIST uiBufLen [in] Buffer size
Returned value :	Success return 0, failure return Error_code.

Samples :	`DPSDK_CHAR szChannelId[DPSDK_CHANNEL_ID_LEN];`
	`memset(&szChannelId, 0, sizeof(szChannelId));`
	`strcpy(szChannelId, "168383947B19V88R2VM0DOT");`
	
	`DPSDK_UINT32 uiNum = 100;`
	`DPSDK_UINT32 uiLen = sizeof(DPSDK_PTZ_PRESETPOINT_LIST) + (uiNum - 1) * sizeof(DPSDK_PTZ_PRESETPOINT_INFO);`
	`DPSDK_PTZ_PRESETPOINT_LIST* pList = (DPSDK_PTZ_PRESETPOINT_LIST*)(new DPSDK_CHAR[uiLen]);`
	`memset(pList, 0, uiLen);`
	`DPSDK_INT32 iRet = DPSDK_PtzGetPresetPoints(CAppData::m_iLoginID, szChannelId, pList, uiLen);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Success`
	`}`

Parent subject: PTZ

TV Wall

Parent Subject: Interface Function Definition

Getting video wall list DPSDK_GetTVWallList

Name	Note
Description:	Getting video wall list
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	`DPSDK_INT32DPSDK_GetTVWallList(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_TVWALL_LIST* pTVWallList,`
	`    DPSDK_UINT32 uiBufLen`
	`);`

Parameters:	iSessionID [in] User session ID pTVWallList [out] Video wall list, refer to DPSDK_TVWALL_LIST structure uiBufLen [in] Buffer size
Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	`DPSDK_UINT32 uiTVWallNum = 100;          //Search 100 video walls.`
	`DPSDK_UINT32 uiLen = sizeof(DPSDK_TVWALL_LIST) + (uiTVWallNum - 1) * sizeof(DPSDK_TVWALL_BASE_INFO);`
	`DPSDK_TVWALL_LIST* pTVWallList = (DPSDK_TVWALL_LIST*)(new DPSDK_CHAR[uiLen]);`
	`memset(pTVWallList, 0, uiLen);`
	`DPSDK_INT32 iRet = DPSDK_GetTVWallList(CAppData::m_iLoginID, pTVWallList, uiLen);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    // Successful`
	`}`
	`delete[]pTVWallList;`
	`pTVWallList = NULL;`

Parent Subject: Video wall

Getting video wall information DPSDK_GetTVWallInfo

Name	**Note**
Description:	Getting video wall information
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	`DPSDK_INT32DPSDK_GetTVWallInfo(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iTVWallID,`
	`   DPSDK_TVWALL_INFO* pTVWallInfo`
	`);`

Parameters:	iSessionID
	  [in] User session ID
	iTVWallID
	  [in] Video wall ID
	pTVWallInfo
	  [out] Video wall information， refer to DPSDK_TVWALL_INFO structure

Returned Value:	Return 0 if succeeded， Return Error code if failed.

Samples:	`DPSDK_TVWALL_INFO strParam;`
	`memset(&strParam, 0, sizeof(strParam));`
	`DPSDK_INT32 iRet = DPSDK_GetTVWallInfo(CAppData::m_iLoginID, m_iCurTVWallID, &strParam);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    // Successful`
	`}`

Parent Subject: Video wall

Getting video wall task list DPSDK_GetTVWallTaskList

Name	Note
Description：	Getting video wall task list
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function：	`DPSDK_INT32DPSDK_GetTVWallTaskList(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iTVWallID,`
	`    DPSDK_TVWALL_TASK_LIST* pTaskList,`
	`   DPSDK_UINT32 uiBufLen`
	`);`

Parameters：	iSessionID
	  [in] User session ID
	iTVWallID
	  [in] Video wall ID
	pTaskList
	  [out] Video wall information， refer to DPSDK_TVWALL_TASK_LIST structure
	uiBufLen
	  [in] Buffer size

Returned Value：	Return 0 if succeeded， Return Error code if failed.

Samples：	`DPSDK_UINT32 uiNum = 100;          //Search 100 tasks`
	`DPSDK_UINT32 uiLen = sizeof(DPSDK_TVWALL_TASK_LIST) + (uiNum - 1) * sizeof(DPSDK_TVWALL_TASK_BASE_INFO);`
	`DPSDK_TVWALL_TASK_LIST* pList = (DPSDK_TVWALL_TASK_LIST*)(new DPSDK_CHAR[uiLen]);`
	`memset(pList, 0, uiLen);`
	`DPSDK_INT32 iRet = DPSDK_GetTVWallTaskList(CAppData::m_iLoginID, m_iCurTVWallID, pList, uiLen);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    // Successful`
	`}`

Parent Subject： Video wall

Getting video wall task information DPSDK_GetTVWallTaskInfo

Name	Note
Description:	Getting video wall task information
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	DPSDK_INT32DPSDK_GetTVWallTaskInfo(
	    DPSDK_INT32 iSessionID,
	    DPSDK_INT32 iTVWallID,
	    DPSDK_INT32 iTaskID,
	    DPSDK_DataCallback fDataCallBack,
	    DPSDK_VOID* pUserData
	);

Parameters:	iSessionID [in] User session ID iTVWallID [in] Video wall ID iTaskID [in] task ID fDataCallBack [in] Data synchronization call function, refer to DPSDK_DATA_TYPE for data type, refer to DPSDK_TVWALL_TASK_INFO for structure pUserData [in] User data
Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	DPSDK_TVWALL_TASK_INFO strParam;
	memset(&strParam, 0, sizeof(strParam));
	DPSDK_INT32 iRet = DPSDK_GetTVWallTaskInfo(CAppData::m_iLoginID, m_iCurTVWallID, m_iCurTaskID, &DataCallback, &strParam);
	if(iRet == DPSDK_SUCCESS)
	{
	  //Successfully got organization data
	}

Parent Subject: Video wall

Getting video wall running task information DPSDK_GetCurrentTVWallTaskInfo

Name	Note
Description:	Getting video wall running task information
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	DPSDK_INT32DPSDK_GetCurrentTVWallTaskInfo(
	    DPSDK_INT32 iSessionID,
	    DPSDK_CURRENT_TVWALL_TASK_LIST* pTaskList,
	    DPSDK_UINT32 uiBufLen
	);

Parameters:	iSessionID [in] User session ID pTaskList [out] Video wall task list, refer to DPSDK_CURRENT_TVWALL_TASK_LIST structure uiBufLen [in] Buffer size
Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	DPSDK_UINT32 uiTVWallNum = 100;
	DPSDK_UINT32 uiLen = sizeof(DPSDK_CURRENT_TVWALL_TASK_LIST) + (uiTVWallNum - 1) * sizeof(DPSDK_CURRENT_TVWALL_TASK_INFO);
	DPSDK_CURRENT_TVWALL_TASK_LIST* pTaskList= (DPSDK_CURRENT_TVWALL_TASK_LIST*)(new DPSDK_CHAR[uiLen]);
	memset(pTaskList, 0, uiLen);
	DPSDK_INT32 iRet = DPSDK_GetCurrentTVWallTaskInfo(CAppData::m_iLoginID, pTaskList, uiLen);
	if(iRet == DPSDK_SUCCESS)
	{
	    //Successfuu=l
	}

Parent Subject: Video wall

Delete video wall task DPSDK_DeleteTVWallTask

Name	Note
Description：	Delete video wall task
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function：	`DPSDK_INT32DPSDK_DeleteTVWallTask(`
	`    DPSDK_INT32 iSessionID,`
	`    DPSDK_INT32 iTVWallId,`
	`   DPSDK_INT32 iTaskId`
	`);`

Parameters：	iSessionID
	  [in] User session ID
	iTVWallId
	  [in] Video wall ID
	iTaskId
	  [in] Task ID

Returned Value：	Return 0 if succeeded, Return Error code if failed.

Samples：	`DPSDK_INT32 iRet = DPSDK_DeleteTVWallTask(CAppData::m_iLoginID, m_iCurTVWallID, m_iCurTaskID);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`    //Successful`
	`}`

Parent Subject： Video wall

Add or save video wall task DPSDK_AddTVWallTaskEx

Name	**Note**
Description :	Add or save video wall task
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function :	`DPSDK_INT32DPSDK_AddTVWallTaskEx(`
	`      DPSDK_INT32 iSessionID,`
	`      DPSDK_TVWALL_TASK_INFO* pTaskInfo`
	`);`

Parameters :	iSessionID [in] User session ID pTaskInfo [in] Video wall task, refer to DPSDK_TVWALL_TASK_INFO structure
Returned Value :	Return 0 if succeeded, Return Error code if failed.

Samples :	`DPSDK_TVWALL_TASK_INFO strParam;`
	`memset(&strParam, 0, sizeof(strParam));`
	`strParam.struBaseInfo.iTaskId = m_iTaskIDInc;`
	`m_iTaskIDInc++;`
	`strParam.struBaseInfo.iTVWallId = m_iCurTVWallID;`
	`strcpy(strParam.struBaseInfo.szTaskName, "task " + strParam.struBaseInfo.iTaskId);`
	`strcpy(strParam.struBaseInfo.szTaskDesc, "taskDes " + strParam.struBaseInfo.iTaskId);`
	`DPSDK_INT32 iRet = DPSDK_AddTVWallTaskEx(CAppData::m_iLoginID, &strParam);`
	`if(iRet == DPSDK_SUCCESS)`
	`{`
	`      //Successful`
	`}`

Parent Subject : Video wall

Open new screen window DPSDK_TVWallOpenWindows

Name	Note
Description:	Open new screen window
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit

Function:	DPSDK_INT32DPSDK_TVWallOpenWindows(
	    DPSDK_INT32 iSessionID,
	    DPSDK_TVWALL_OPEN_WINDOW* pOpenWndList,
	    DPSDK_INT32 uiBufLen
	);

Parameters:	iSessionID [in] User session ID pOpenWndList [in] New screen window list,refer to DPSDK_TVWALL_OPEN_WINDOW structure uiBufLen [in] Buffer size
Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	DPSDK_UINT32 uiTVWallNum = 100;
	DPSDK_UINT32 uiLen = sizeof(DPSDK_TVWALL_OPEN_WINDOW) + (uiTVWallNum - 1) * sizeof(DPSDK_TVWALL_WINDOW_INFO);
	DPSDK_TVWALL_OPEN_WINDOW * pOpenWndLis = (DPSDK_TVWALL_OPEN_WINDOW *)(new DPSDK_CHAR[uiLen]);
	memset(pOpenWndLis, 0, uiLen);
	DPSDK_INT32 iRet = DPSDK_TVWallOpenWindows(CAppData::m_iLoginID, pOpenWndLis, uiLen);
	if(iRet == DPSDK_SUCCESS)
	{
	     //Successful
	}

Parent Subject: Video wall

Output to the video wall DPSDK_MapToTVWallEx

Name	Note
Description:	Output to the video wall
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit
Function:	DPSDK_INT32DPSDK_MapToTVWallEx(DPSDK_INT32 iSessionID, DPSDK_TVWALL_CONTROL_INFO* pCtrlInfo, DPSDK_TVWALL_TASK_INFO_LIST* pTaskInfoList);
Parameters:	iSessionID [in] User session ID pCtrlInfo [in] Output to the video wall control information, refer to DPSDK_TVWALL_CONTROL_INFO structure pTaskInfoList [in] Video wall task list, refer to DPSDK_TVWALL_TASK_INFO_LIST structure
Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	DPSDK_TVWALL_CONTROL_INFO struCtrlInfo;
	memset(&struCtrlInfo, 0, sizeof(struCtrlInfo));
	struCtrlInfo.iControlType = TVWALL_PLAN_TASK;
	struCtrlInfo.iSplitNum = 1;
	struCtrlInfo.iTvType = 0;
	struCtrlInfo.iZoder = 0;
	struCtrlInfo.struScreenPos.fHeight = 0.5;
	struCtrlInfo.struScreenPos.fWidth = 0.5;
	struCtrlInfo.struScreenPos.fLeft = 0.1;
	struCtrlInfo.struScreenPos.fTop = 0.1;
	DPSDK_TVWALL_TASK_INFO_LIST struTaskInfoList;
	memset(&struTaskInfoList, 0, sizeof(struTaskInfoList));
	struTaskInfoList.iTVWallTaskNum = 100;
	strcpy(struTaskInfoList.pTVWallTaskList->struBaseInfo.szTaskName, "");
	strcpy(struTaskInfoList.pTVWallTaskList->struBaseInfo.szTaskDesc, "");
	struTaskInfoList.pTVWallTaskList->struChannelExtList.uiTotal = 10;
	struTaskInfoList.pTVWallTaskList->struChannelExtList.pChannelExtList->iChannelNum = 100;
	struTaskInfoList.pTVWallTaskList->struChannelExtList.pChannelExtList->iPort = 3777;
	struTaskInfoList.pTVWallTaskList->struChannelExtList.pChannelExtList->iType = 0;
	strcpy(struTaskInfoList.pTVWallTaskList->struChannelExtList.pChannelExtList->szPassword, "admin123");
	strcpy(struTaskInfoList.pTVWallTaskList->struChannelExtList.pChannelExtList-

```
>szUserName, "admin");
struTaskInfoList.pTVWallTaskList->struScreenOperList.uiTotal = 10;
struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->fHeight = 0.8;
struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->fWidth = 0.8;
struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->fLeft = 0.2;
struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->fTop = 0.1;
struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->iScreenMode =
1;
struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->iTvIdx = -1;
struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->iVisitorMode =
1;
struTaskInfoList.struProjectList.uiCount = 10;
struTaskInfoList.struProjectList.pProjList->iTaskNum = 100;
strcpy(struTaskInfoList.struProjectList.pProjList->szProjName, "");
DPSDK_INT32 iRet = DPSDK_MapToTVWallEx(CAppData::m_iLoginID, &struCtrlInfo,
&struTaskInfoList);
if(iRet == DPSDK_SUCCESS)
{
    //Successful
}
```

Parent Subject : Video wall

Save video wall task file DPSDK_SaveTVWallProjectFile

Name	Note
Description:	Save video wall task file
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit
Function:	DPSDK_INT32DPSDK_SaveTVWallProjectFile(DPSDK_INT32 iSessionID, DPSDK_CHAR* szFileName, DPSDK_TVWALL_TASK_INFO_LIST* pTaskInfoList);
Parameters:	iSessionID [in] User session ID szFileName [in] File name pTaskInfoList [in] Video wall task list, Refer to DPSDK_TVWALL_TASK_INFO_LIST structure
Returned Value:	Return 0 if succeeded, Return Error code if failed.

Samples:	DPSDK_CHAR szFileName[DPSDK_NAME_LEN];
	Strcpy(szFileName, "");
	DPSDK_TVWALL_TASK_INFO_LIST struTaskInfoList;
	memset(&struTaskInfoList, 0, sizeof(struTaskInfoList));
	struTaskInfoList.iTVWallTaskNum = 100;
	strcpy(struTaskInfoList.pTVWallTaskList->struBaseInfo.szTaskName, "");
	strcpy(struTaskInfoList.pTVWallTaskList->struBaseInfo.szTaskDesc, "");
	struTaskInfoList.pTVWallTaskList->struChannelExtList.uiTotal = 10;
	struTaskInfoList.pTVWallTaskList->struChannelExtList.pChannelExtList->iChannelNum = 100;
	struTaskInfoList.pTVWallTaskList->struChannelExtList.pChannelExtList->iPort = 3777;
	struTaskInfoList.pTVWallTaskList->struChannelExtList.pChannelExtList->iType = 0;
	strcpy(struTaskInfoList.pTVWallTaskList->struChannelExtList.pChannelExtList->szPassword, "admin123");
	strcpy(struTaskInfoList.pTVWallTaskList->struChannelExtList.pChannelExtList->szUserName, "admin");
	struTaskInfoList.pTVWallTaskList->struScreenOperList.uiTotal = 10;
	struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->fHeight = 0.8;
	struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->fWidth = 0.8;
	struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->fLeft = 0.2;
	struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->fTop = 0.1;
	struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->iScreenMode = 1;
	struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->iTvIdx = -1;

```
struTaskInfoList.pTVWallTaskList->struScreenOperList.pScreenOperList->iVisitorMode =
1;
struTaskInfoList.struProjectList.uiCount = 10;
struTaskInfoList.struProjectList.pProjList->iTaskNum = 100;
strcpy(struTaskInfoList.struProjectList.pProjList->szProjName, "");
DPSDK_INT32 iRet = DPSDK_SaveTVWallProjectFile(CAppData::m_iLoginID,
szFileName, &struTaskInfoList);
if(iRet == DPSDK_SUCCESS)
{
    //Successful
}
```

Parent Subject : Video wall

Getting video wall task file DPSDK_GetTVWallProjectFile

Name	Note
Description :	Getting video wall task file
OS:	• Windows 7 Pro 32-bit, Windows Server 2008 R2 64-bit • SUSE Linux 11 SP1(2.6.16.21 or higher) 64-bit • SUSE Linux 10 32-bit • Red Hat Enterprise Linux Server Release 5.4(2.6.18-164.el5xen) 32-bit
Function :	DPSDK_INT32DPSDK_GetTVWallProjectFile(DPSDK_INT32 iSessionID, DPSDK_CHAR* szFileName, DPSDK_DataCallback fDataCallBack, DPSDK_VOID* pUserData);
Parameters :	iSessionID [in] User session ID szFileName [in] File name fDataCallBack [in] Data synchronization function, refer to DPSDK_DATA_TYPE for data type, refer to DPSDK_TVWALL_TASK_INFO_LIST for structure pUserData [in] User data
Returned Value :	Return 0 if succeeded, Return Error code if failed.

Samples :	DPSDK_CHAR szFileName[DPSDK_NAME_LEN];
	Strcpy(szFileName, "");
	DPSDK_TVWALL_TASK_INFO_LISTstruTaskInfoList;
	memset(&struTaskInfoList, 0, sizeof(struTaskInfoList));
	DPSDK_INT32 iRet = DPSDK_GetTVWallProjectFile(CAppData::m_iLoginID, szFileName, &DataCallback, &struTaskInfoList);
	if(iRet == DPSDK_SUCCESS)
	{
	//Successful
	}

Parent Subject : Video wall

Error Codes

Error Code	Error Description
-107700	Log uploading function of the module is closed.
-107603	Task name is the same.
-107602	Task name is blank.
-107601	Task description is too long.
-107600	Task name is too long.
-107315	Download writing file failed.
-107314	Query busy.
-107313	No record locked information available.
-107312	Record locking query failed.
-107311	Too quick record label query.
-107310	No record label available.
-107309	Failed record label query.
-107308	Query busy.
-107307	Query timeout.
-107306	Failed to send message.
-107305	Device offline.
-107304	Subscriber absent.
-107303	No record available.
-107302	Record query failed.
-107301	No record.
-107202	XML memory of the distributed plan is insufficient.
-107201	Time period of the plan global configuration is repeated.
-107115	Failed login - not in the term of validity.
-107114	Failed to change the password.

-107113	Failed login - low client version.
-107112	Port is blank.
-107111	IP address is blank.
-107110	Username is blank.
-107109	Buffer is too small.
-107108	Failed login - the user is already logged in.
-107107	Failed login - user locked.
-107106	Failed login - username or password error.
-107105	Failed login - username or password error.
-107104	Failed login - network closed.
-107103	Repeated login.
-107102	Failed login - invalid MAC address.
-107101	Connection timeout.
-107100	Failed login - unknown error.
-802	Device offline.
-801	No channel authority.
-701	Intercom parameter mismatching.
-606	Call curl and return “release error”.
-605	Call curl and return “remote file operation error”.
-604	Call curl and return “local file open error”.
-603	Call curl and return “local file not found”.
-602	Call curl and return “error”.
-601	FTP unknown error.
-502	REST service offline.
-501	Occupied by some other user.
-500	Camera is locked by some other user.
-406	RTSP play timeout.

-405	RTSP setup timeout.
-404	Port analysis failed.
-403	Order is cancelled.
-402	Data transmission failed.
-401	Link service failed.
-400	RTSP monitor failed.
-301	Playback snapshot failed.
-300	No ss service available.
-203	Media type mismatching.
-202	Open audio failed. In intercom with other device.
-201	Open audio failed.
-200	Invalid media session ID (including preview, playback, intercom).
-102	Invalid user token.
-101	Rest request timeout. This error code keeps the same as that inside Rest.
-100	Rest library initialization failed.
-17	Close database failed.
-16	Open database failed.
-15	Error code query failed.
-14	Data absent.
-13	Invalid XML data.
-12	Memory distribution failed.
-11	Invalid data.
-10	MQ initialized.
-9	Buffer insufficient.
-8	User not logged in.
-7	Invalid user. User session ID is invalid.
-6	User is logged in.

-5	System uninitialized.
-4	Illegal parameter.
-3	Internal function calling error.
-2	Unknown error.
-1	Failed.
0	Successful.
404	Internal error.
409	No permission
500	Internal error
1103	No authority.
2001	Username or password error.
2002	User locked.
2003	User is frozen.
2004	User is already logged in.
2005	Current you cannot log in.
2006	Currently you cannot log in.
2007	Please enter file name.
2008	Please enter the original password.
2009	Please enter the new password.
2010	User does not exist.
2011	The new password and the old password cannot be the same.
2012	Please enter headline.
2013	User shall not be blank.
2014	Video shall not be blank.
2015	Please enter device code.
2016	This device does not exist.
2017	This channel does not exist.

2018	Invalid channel ID.
2019	TV wall ID must be an unsigned long integer.
2020	Task ID must be an unsigned long integer.
2021	Please enter task name.
2022	Task does not exist.
2023	TV Wall does not exist.
2024	Label ID must be an unsigned long integer.
2025	Record source must be an unsigned integer.
2026	Please enter headline.
2027	Label time must be an unsigned long integer.
2028	Begin time must be an unsigned long integer.
2029	End time must be an unsigned long integer.
2030	Loop-testing ID must be an unsigned long integer.
2031	Please select cruise duration.
2032	This cruise plan does not exist.
2033	Call does not exist.
2034	The called object does not exist.
2035	Time of duration must be an unsigned long integer.
2036	Status must be an unsigned integer.
2037	Calling time must be an unsigned long integer.
2038	Operation type must be an unsigned integer.
2039	Operation time must be an unsigned long integer.
2040	Service time must be an unsigned integer.
2041	Device or service does not exist.
2042	Calling type must be an unsigned integer.
2043	This calling type is not supported.
2044	Channel S/N must be an unsigned integer.

2045	No broadcasting channel.
2046	Audio type must be an unsigned integer.
2047	Intercom type must be an unsigned integer.
2048	Intercom audio unit must be an unsigned integer.
2049	Sampling rate must be an unsigned integer.
2050	Record type must be an unsigned integer.
2051	Audio intercom is released.
2052	In device intercom.
2053	In calling, please wait for answer.
2054	Data type must be an unsigned integer.
2055	This data type is not supported.
2056	Code stream type must be an unsigned integer.
2057	This code stream is not supported.
2058	Please enter hardware ID.
2059	Code stream ID must be an unsigned long integer.
2060	The code stream ID does not exist.
2061	Alarm ID must be an unsigned long integer.
2062	Alarm status must be an unsigned integer.
2063	Please select record month.
2064	Record duration must be an unsigned long integer.
2065	Please enter alarm code.
2066	This record source is not supported.
2067	This record type is not supported.
2068	This alarm does not exist.
2069	User ID must be an unsigned integer.
2070	Please select channel.
2071	Please enter ID.

2072	ID must be an unsigned long integer.
2073	Record lock information does not exist.
2074	This operation type is not supported.
2075	Locking time must be an unsigned long integer.
2076	Command must be an unsigned integer.
2077	Direction must be an unsigned integer.
2078	Preset point x must be an integer.
2079	Preset point y must be an integer.
2080	Preset point z must be an integer.
2081	Step size x must be an unsigned integer.
2082	Step size y must be an unsigned integer.
2083	Please enter the handler.
2084	You are not authorized with this operation.
2085	PTZ zoom parameter must be a floating point number.
2086	PTZ focus parameter must be a floating point number.
2087	The preset point code shall not be blank.
2088	Please enter the preset point name.
2089	Please enter page number.
2090	Please enter page size.
2091	Please enter command value.
2092	Alarm status must be an unsigned integer.
2093	Alarm type must be an unsigned integer.
2094	Alarm level must be an unsigned integer.
2095	Handling status must be an unsigned integer.
2096	Alarm begin time must be an unsigned long integer.
2097	Alarm end time must be an unsigned long integer.
2098	Handling begin time must be an unsigned long integer.

2099	Handling end time must be an unsigned long integer.
2100	Page number must be an unsigned long integer.
2101	Page size must be an unsigned integer.
2102	Command size must be an unsigned integer.
2103	The handling status shall not be blank.
2104	Please enter matrix ID.
2105	Please select operation type.
2106	The index of TV wall shall not be blank.
2107	TV Wall index must be an unsigned integer.
2108	Screen index shall not be blank.
2109	Screen index must be an unsigned integer.
2110	Sub TV wall index shall not be blank.
2111	Sub TV wall index must be an unsigned integer.
2112	Please enter the number of partition.
2113	The number of partition must be an unsigned integer.
2114	Please enter TV wall ID.
2115	TV wall ID must be an unsigned integer.
2116	Please enter direction information value.
2117	Please enter left margin.
2118	Please enter top margin.
2119	Please enter the width.
2120	Please enter the height.
2121	TV wall type shall not be blank.
2122	TV wall type shall not be blank.
2123	Please select alarm level.
2124	Alarm level must be an unsigned integer.
2125	Array shall not be blank.

2126	Array must be an unsigned integer.
2127	The sub-window shall not be blank.
2128	Sub-window must be an integer.
2129	Email is not available.
2130	This command is not supported.
2131	Alarm handling record does not exist.
2132	This user role does not exist.
2133	You are not authorized for this channel.
2134	You are not authorized for this device.
2135	Please enter the receiver.
2136	Please enter the email content.
2137	Word has reached its limit.
2138	Target must be an unsigned integer.
2139	Please select the alarm source.
2140	Time of duration must be a positive integer.
2141	Azimuth must be a digit / digits.
2142	Please select data type.
2143	Please enter data name.
2144	Data does not exist.
2145	Window information does not exist.
2146	Screen type does not exist.
2147	Screen type must be an unsigned integer.
2148	Left margin must be an unsigned floating-point number.
2149	Top margin must be an unsigned floating-point number.
2150	Width must be an unsigned floating-point number.
2151	Height must be an unsigned floating-point number.
2152	Window ID shall not be blank.

2153	Window ID must be an unsigned integer.
2154	Please enter information.
2155	The MAC address of this computer is disabled.
2156	Domain certification center is disabled.
2157	Processing information is out of allowed length.
2178	Get current task failed from VMS
3000	Dialogue error or timeout.
3001	Starting service.
3002	Starting service.
3003	Starting service.
3004	Starting service.
3100	Service time may go wrong.
3101	Service time may go wrong.
10803	Service unavailable.
101001	Organization code is empty.
101002	Organization not exist.
101003	Organization is root node.
101004	Organization has child nodes.
101005	Organization has child devices or channels.
101006	Organization name already exist.
101007	Organization name slop.
101008	Parent code is empty.
101009	Parent not exist.
101010	Organization level has reached level 10.
101011	Parent has 999 child nodes.
101012	Root node cannot be modified.
101013	Child node cannot to be root.

101014	Can not set child node.
101015	Can not set self.
101016	The organization to move to not exist.
101017	Batch add organization failed.
101101	Batch add device failed.
101102	Device code is empty.
101103	No permission.
101104	Resource not exist.
101105	Encoder channel not exist.
101106	Intelligence channel is enough.
101107	Fail to request intelligence service.
101108	Update channel intelliState error.
101109	Fail to request DSE service.
101110	Fail to operator intelliState server.
101501	The account has been locaked.
101502	The account has not exist.
101503	The account has been exist.
101504	Username or password wrong.
101505	Token has been lose efficacy,please relogin.
101506	Create random key case exception.
101507	Encryption type is wrong.
101508	Random key can not be null.
101509	Signature can not be null.
101510	Signature is not right.
101511	Password need to update.
101512	Create token exception.
101513	Save token to redis failed.

101514	The account has expired.
101515	The new password is equal to old password.
101516	The old password incorrect.
101517	User's department can not be null.
101518	Mamager account can not be deleted.
101519	Can not delete online user.
101520	Password questios not exist.
101521	The number of roleis touch upper limit.
101522	This role is existed.
101523	Default role can not be deleted.
101524	Relate user can not be deleted.
101525	User can not update owner role.
101601	The name of role is empty.
101602	The name of role is slop.
101603	The memo of role is slop.
101604	Role grade slop.
101605	Role code is empty.
101606	User code can not be empty.
101607	User name can not be empty.
101608	User password can not be empty.
101609	User organization can not be empty.
101610	User type can not be empty.
101611	Please check whether the user could be multiplexing .
101612	User can not change the password which is not himself.
101801	File name is empty.
101802	Date type error.
101803	BeginTime later than endTime.

101804	No video over period.
101805	Start date empty.
101806	Start date type error.
101807	Days beyond 1-31.
101808	The type of day error.
101809	The type of subType error.
101810	The type of scheme error.
101811	The type of duration error.
101812	The duration less than 1.
101813	The duration more than 600.
101814	The action is empty.
101815	The type of subType slop.
101816	Url type error.
101817	Url type slop.
101818	Session empty.
101819	Video stream type empty.
101820	Video stream type slop,only support main,extra1,extra2,extra3.
101821	Remote ip empty.
101822	Remote port empty.
101823	Protocol empty.
101824	Protocol type error,only support UDP,TCP,TLS.
101825	Packet type error, only support RTP.
101826	Stream type empty.
101827	Stream type error,only support PS、TS.
101828	Start time empty.
101829	Start time type error.
101830	End time empty.

101831	End time type error.
101832	Time interval more than 24 hours.
101833	Quota limit error.
101834	Get realtime minotor uri failed.
101835	Session exists when push stream.
101836	Push stream error.
101837	Start push stream error.
101838	Stop push stream error.
101839	Get snapshot error.
101840	Get push stream status error.
101841	Time template resource has depend.
101842	Record Has Locked Can not Lock Again.
101847	The channel is not record quota.
101848	Remote record interval less then 60s.
101850	The channel is not in manual recording.
101851	Remote record duration must more then 60s and less then 24h.
101901	Update alarm plan status fail.
101902	Create alarm paln fail.
101903	Update alarm plan fail.
101904	Delete alarm plan fail.
101905	Login fail.
101906	Login timeout.
101907	UID is empty.
101908	Can not achiece this message.
102101	Domain name can not be null.
102102	Domain code can not be null.
102103	Domain IP can not be same.

102104	Domain name can not be same.
102201	Ptz preset index must be integer.
102202	Channel code empty.
102203	Preset index less than one.
102204	Ptz focus change arg empty.
102205	Ptz focus change arg slop.
102206	Ptz preset name empty.
102207	Ptz move speed slop.
102208	Ptz span empty.
102209	ptz tilt empty.
102210	Ptz zoom empty.
102211	Ptz span slop.
102212	Ptz tilt slop.
102213	Ptz zoom slop.
102214	Ptz location horizontal empty.
102215	Ptz location vertical empty.
102216	Ptz location zoom empty.
102217	Ptz location horizontal slop.
102218	Ptz location vertical slop.
102219	Ptz location zoom slop.
102220	Ptz time out slop.
102234	Ptz location device not supported.
102301	TvWall name exists.
102302	TvWall id empty.
102303	TvWall id type error.
102304	TvWall task id empty.
102305	TvWall task id type error.

102306	TvWall screen id empty.
102307	TvWall screen id type error.
102308	Open windows args is empty.
102309	Window id empty.
102310	Window id type error.
102311	Window z empty.
102312	Window z type error.
102313	Window height empty.
102314	Window height type error.
102315	Window left empty.
102316	Window left type error.
102317	Window top empty.
102318	Window top type error.
102319	Window width empty.
102320	Window width type error.
102321	Request tvwall service failed.
102401	Client file name empty.
102402	Client file name slop.
102403	Client file data empty.
102451	Region id empty.
102452	Request method empty.
102453	Dst url empty.

System Initialization DPSDK_Init

Name	**Note**
Description：	System initialization
Function：	DPSDK_INT32 DPSDK_Init();
Parameters：	None
Returned value：	Success return 0, failure return Error_code.

Samples：	if (DPSDK_Init () == DPSDK_SUCCESS)
	{
	   CAppData::ShowMsgInfo("Init system success", CAppData::TITLE_PROMPT);
	}
	else
	{
	   CAppData::ShowMsgInfo("Init system failed");
	}

Setup Event Callback Function DPSDK_SetEventCallBack

Name	**Note**
Description:	Setup Event Callback Function

Function:	DPSDK_INT32 DPSDK_SetEventCallBack(
	    DPSDK_INT32 iSessionID,
	    EventCallBack fEventCallBack = NULL,
	    DPSDK_VOID* pUserData = NULL
	);

Parameters:	iSession
	  [in] User Session
	fEventCallBack
	  [in] Event Callback Function
	User Data
	  [in] Data Length

Returned Value:	Returned value is 0 in case of success.
Samples:	None

Parent Subject: Definition of Callback Function

DPSDK_ALARMEVENT_NOTIFY

Alarm event (notice)

```
Typedef Struct
{
    DPSDK_CHAR SzAlarmCode[DPSDK_ALARM_ALARMCODE_LEN];          // Alarm code
    DPSDK_CHAR  SzAlarmNodeCode[DPSDK_ALARM_NODECODE_LEN];          //  Alarm
source code
    DPSDK_CHAR  SzAlarmTime[DPSDK_ALARM_TIME_LEN];                      // Alarm time
Yyyymmddhhmmss
    DPSDK_INT32 IAlarmGrade;                                    // Alarm level (Reference resources
AlarmLevel_e)
    DPSDK_INT32 IAlarmStatus;                                   // Alarm state (Reference resources
AlarmState_e)
    DPSDK_INT32 IAlarmObjType;                                      // Alarm object type (Reference
resources AlarmObject_e)
    DPSDK_INT32 IAlarmType;                                     // Alarm type (Reference resources
Alarm_type_e)
    DPSDK_INT32 IAlarmCategory;                                // Type of alarm (Reference resources
AlarmCategory_e)
    DPSDK_CHAR   SzAlarmMessage[DPSDK_ALARM_ALARMMESSAGE_LEN];  //   Alarm
extension information (for example,GPSThe extended information of the alarm includes the latitude
and longitude, the heightEtc.
    DPSDK_UINT32   UiAlarmLinkVedioListSize;  //   Alarm   video   linkage   information   list
number(Not greater than
    DPSDK_ALARM_LINKVEDIOINFOLIST_SIZE)
    DPSDK_ALARMLINKVEDIO_INFO
StruAlarmLinkVedioList[DPSDK_ALARM_LINKVEDIOINFOLIST_SIZE];   //    Alarm    video
linkage information list(Most returnDPSDK_ALARM_LINKVEDIOINFOLIST_SIZEVideo linkage
information)
}DPSDK_ALARMEVENT_NOTIFY;
```

Father theme:structural morphology

DPSDK_ALARMCONFIRM_NOTIFY

Alarm confirmation (notice)

```
Typedef Struct
{
    DPSDK_CHAR SzAlarmCode[DPSDK_ALARM_ALARMCODE_LEN];                // Alarm code
    DPSDK_CHAR SzAlarmTime[DPSDK_ALARM_TIME_LEN];                     // Alarm time Yyyymmddhhmmss
    DPSDK_INT32 IHandleStatus;                              // Processing state
    DPSDK_CHAR  SzHandleMessage[DPSDK_ALARM_HANDLEMESSAGE_LEN];           // Handling opinions
    DPSDK_CHAR  SzHandleUser[DPSDK_ALARM_HANDLERUSER_LEN];  // Handling human username
    DPSDK_UINT32 UiEmailReceiverListSize;                              // The actual number of notification mailbox lists is not greater than that of the alarm. DPSDK_EMAILRECEIVERLIST_SIZE)
    DPSDK_EMAILADDRESS StruEmailReceiverList[DPSDK_ALARM_EMAILRECEIVERLIST_SIZE]; // Alarm processing notification mailbox list(Most return DPSDK_EMAILRECEIVERLIST_SIZEA mail address)
}DPSDK_ALARMCONFIRM_NOTIFY;
```

Father theme:structural morphology

DPSDK_ALARM_DETAILINFO_NOTIFY

Alarm information (notice)

```
Typedef Struct
{
    DPSDK_CHAR SzAlarmCode[DPSDK_ALARM_ALARMCODE_LEN];        // Alarm code
    DPSDK_CHAR  SzAlarmTime[DPSDK_ALARM_TIME_LEN];                    // Alarm time
Yyyymmddhhmmss
    DPSDK_CHAR   SzAlarmPicture[DPSDK_ALARM_ALARMPICTURE_LEN];      //   Alarm
snapshot path
    DPSDK_UINT32 UiAlarmPictureSize;                           // Alarm snapshot size
}DPSDK_ALARM_DETAILINFO_NOTIFY;
```

Father theme:structural morphology

DPSDK_ALARMEXPORT_RESULT_NOTIFY

Alarm export results (notice)

```
Typedef Struct
{
    DPSDK_UINT32 UiSessionId;                                   // Session marking
    DPSDK_CHAR
SzDownloadPath[DPSDK_ALARM_ALARMEXPORTDOWNLOADPATH_LEN];                //
Downloading path
}DPSDK_ALARMEXPORT_RESULT_NOTIFY;
```

Father theme:structural morphology

DPSDK_DEV_STATUS_NOTIFY

Device status change notification

```
Typedef Struct
{
    DPSDK_CHAR SzDeviceID[DPSDK_DEVICE_ID_LEN];   // equipment ID
    DPSDK_INT32  IStatus;                          // Equipment state See DPSDK_DEV_STATUS
Definition
}DPSDK_DEV_STATUS_NOTIFY;
```

Father theme:structural morphology

DPSDK_CHANNEL_STATUS_NOTIFY

Channel state change notification

```
Typedef Struct
{
    DPSDK_CHAR SzChannelID[DPSDK_CHANNEL_ID_LEN]; // channel ID
    DPSDK_INT32  IStatus;                          // Channel state See DPSDK_DEV_STATUS
Definition
}DPSDK_CHANNEL_STATUS_NOTIFY;
```

Father theme:structural morphology

DPSDK_ORG_BASE_INFO

Organization of basic data

```
Typedef Struct
{
     DPSDK_CHAR SzOrgCode[DPSDK_ORG_CODE_LEN]; // organization Code
     DPSDK_CHAR SzOrgName[DPSDK_NAME_LEN];      // Organization name
     DPSDK_CHAR SzOrgSN[DPSDK_ORG_SN_LEN];      // organization SN code
     DPSDK_INT32 IOrgType;                      // Organization node type
     DPSDK_INT32 IOrgSort;                      // Organization sort
}DPSDK_ORG_BASE_INFO;
```

Father theme:structural morphology

DPSDK_MOVE_ORG_NOTIFY

Mobile organization notification

```
Typedef Struct
{
    DPSDK_CHAR SzOldOrgCode[DPSDK_ORG_CODE_LEN];        // Old organization node Code
    DPSDK_CHAR   SzOldParentOrgCode[DPSDK_ORG_CODE_LEN];   //   Old   organization father node Code
    DPSDK_CHAR SzNewOrgCode[DPSDK_ORG_CODE_LEN];           //  New  organization nodeCode
    DPSDK_CHAR SzNewParentOrgCode[DPSDK_ORG_CODE_LEN]     //  New  organization father node Code
}DPSDK_MOVE_ORG_NOTIFY;
```

Father theme:structural morphology

DPSDK_ADD_DEVICE_NOTIFY

Increase the device notification

```
Typedef Struct
{
    DPSDK_CHAR SzOrgCode[DPSDK_ORG_CODE_LEN];     // organization code
    DPSDK_DEV_ALL_INFO StruDevAllInfo;            // Device data
}DPSDK_ADD_DEVICE_NOTIFY;
```

Father theme:structural morphology

DPSDK_MODIFY_DEVICE_NOTIFY

Modification of device notification

```
Typedef Struct
{
    DPSDK_CHAR SzOldOrgCode[DPSDK_ORG_CODE_LEN]; // Old organization
    DPSDK_CHAR SzNewOrgCode[DPSDK_ORG_CODE_LEN]; // New organization
    DPSDK_DEV_ALL_INFO StruDevAllInfo;          // Device data
}DPSDK_MODIFY_DEVICE_NOTIFY;
```

Father theme:structural morphology

DPSDK_DELETE_DEVICE_NOTIFY

Delete device notification   Support batch

```
Typedef Struct
{
    DPSDK_CHAR SzOrgCode[DPSDK_ORG_CODE_LEN];    // organization code
    DPSDK_CHAR SzDeviceID[DPSDK_DEVICE_ID_LEN]; // equipment ID
}DPSDK_DELETE_DEVICE_NOTIFY;
```

Father theme:structural morphology

DPSDK_MOVE_DEVICE_NOTIFY

Mobile device notification   Support batch

```
Typedef Struct
{
    DPSDK_CHAR SzDeviceID[DPSDK_DEVICE_ID_LEN]; // equipment ID
    DPSDK_CHAR SzOldOrgCode[DPSDK_ORG_CODE_LEN]; // The original organization of
the equipment code
    DPSDK_CHAR SzNewOrgCode[DPSDK_ORG_CODE_LEN]; // New organization of
equipment code
}DPSDK_MOVE_DEVICE_NOTIFY;
```

Father theme:structural morphology

DPSDK_USERONLINESTATUS_NOTIFY

User Online Status Notice

```
typedef struct
{
    DPSDK_INT32 iUserId;            // User ID
    DPSDK_INT32 iOnlineStatus;      //Online status. For details, please refer to
DPSDK_USER_STATUS definition.
}DPSDK_USERONLINESTATUS_NOTIFY;
```

Father theme:structural morphology

DPSDK_USERADD_NOTIFY

New Notice to User

```
typedef struct
{
    DPSDK_INT32 iUserId;                                             // User ID
    DPSDK_CHAR  szUserName[DPSDK_NAME_LEN];                                    // Username
}DPSDK_USERADD_NOTIFY;
```

Father theme:structural morphology

DPSDK_USERDELETE_NOTIFY

User Deletion Notice

```
typedef struct
{
    DPSDK_INT32 iUserId;                                    // User ID
}DPSDK_USERDELETE_NOTIFY;
```

Father theme:structural morphology

DPSDK_VIEWINFO_CHANGED_NOTIFY

Visible Range Change Notice

```
typedef struct
{
    DPSDK_CHAR szChannelId[DPSDK_CHANNEL_ID_LEN];                    // Channel ID
    DPSDK_INT32 iAzimuth;                                            // Starting Azimuth of Visible Range
    DPSDK_INT32 iDistance;                                           // Distance (Radius) of Visible Range
    DPSDK_INT32 iAngle;                                              // Angle of Visible Range
}DPSDK_VIEWINFO_CHANGED_NOTIFY;
```

Father theme:structural morphology

DPSDK_DEVICELOCATION_NOTIFY

Notice of equipment modification on the map

```
Typedef Struct
{
     DPSDK_UINT32 UiDeviceCodeListSize;           // Device code list number
     DPSDK_DEVICECODE_INFO StruDeviceCodeList[1]; // Device codinglist
}DPSDK_DEVICELOCATION_NOTIFY;
```

Father theme:structural morphology

DPSDK_LOCKSTATUS_CHANGED_NOTIFY

Cloud platform lock state change notification

```
Typedef Struct
{
    DPSDK_INT32 ILockStatus;                          // Lock state 0=Unlock,1=locking
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN];        // channel ID
    DPSDK_CHAR SzLockUserName[DPSDK_NAME_LEN];            // User name for locking  the
cloud
    DPSDK_CHAR SzLockUserLevel[DPSDK_USER_LEVEL_LEN];    // Lock the user level of
the cloud
}DPSDK_LOCKSTATUS_CHANGED_NOTIFY;
```

Father theme:structural morphology

DPSDK_RADERFRAME_NOTIFY

Notification of radar information

```
Typedef Struct
{
    DPSDK_CHAR SzRaderId[DPSDK_DEVICE_ID_LEN];      // radar Id
    DPSDK_UINT32 UiRaderTargetInfoSize;              // The number of target information of radar
    DPSDK_RADER_TARGET_INFO StruRaderTargetInfo[1]; // Radar target information list
}DPSDK_RADERFRAME_NOTIFY;
```

Father theme:structural morphology

DPSDK_FACE_INFO_NOTIFY

Face capture information notification

```
Typedef Struct
{
     DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; // channel ID
     DPSDK_INT32 IFaceImageId;                    // Snap face ID
     DPSDK_CHAR SzFaceImageUrl[DPSDK_URL_LEN];      // Take a picture of a face
     DPSDK_BOOL BHited;                            // Is it a hit
     DPSDK_CHAR SzPictureUrl[DPSDK_URL_LEN];        // Scene graph
     DPSDK_INT32 IRecAge;                          // Identification age
     DPSDK_INT32 IRecExpress;                      // Distinguish Express
     DPSDK_INT32 IRecFringe;                       // Identify the Liu Hai 0-no 1-yes
     DPSDK_INT32 IRecSex;                          // Identification of sex 0-Unknown 1-male 2-female
     DPSDK_INT32 IRecGlasses;                      // Eyeglasses 0-no 1-glasses 2-Sunglasses
     DPSDK_INT32 IRecEmotion; // Recognition of emotions 0-Smile 1-anger 2-Sadness 3-Hate
4-Fear 5-surprised 6-normal 7-Laugh
     DPSDK_INT32 IAppearTimes; // Number of historical occurrences
     DPSDK_CHAR SzBeginTime[DPSDK_TIME_LEN];       // View time
     DPSDK_CHAR SzEndTime[DPSDK_TIME_LEN];         // Departure time
     DPSDK_UINT32 UiSimilarFaceListSize;           // Number of similar face information
     DPSDK_FACE_INFO StruSimilarFaceList[1];       // Similar face information list
}DPSDK_FACE_INFO_NOTIFY;
```

Father theme:structural morphology

DPSDK_PERSONTYPE_NOTIFY

Staff type change notification

```
Typedef Struct
{
    DPSDK_INT32 IOperateType; // Operation type:1-Add to,   2-To update,   3-delete
    DPSDK_INT32 IPersonTypeId; // Type of personnelId
    DPSDK_CHAR SzPersonTypeName[DPSDK_PERSONTYPE_NAME_LEN]; // Person type
name
}DPSDK_PERSONTYPE_NOTIFY;
```

Father theme:structural morphology

DPSDK_USERDEFINE_INFO

User Custom Data Change

```
typedef struct
{
    DPSDK_INT32 iAlertUserDefineDataType;                    // User custom data change type. For details, please refer to DPSDK_ALERT_USERDEFINEDATA_TYPE.
    DPSDK_CHAR szFileName[DPSDK_USERDEFINEDATA_FILENAME_LEN];        //Custom User File Name
    DPSDK_CHAR szFileData[1];                                    //Custom User File Info
}DPSDK_USERDEFINE_INFO;
```

Father theme:structural morphology

DPSDK_POS_DATA_NOTIFY

```
Typedef Struct
{
     DPSDK_CHAR Sz Device Code[DPSDK_DEVICE_ID_LEN];
     DPSDK_CHAR Sz POS ChnlId[DPSDK_CHANNEL_ID_LEN];
     DPSDK_INT32 IPOSDataLen;
     DPSDK_CHAR *pPOSData;
     DPSDK_TIMET LPostTime;
}DPSDK_POS_DATA_NOTIFY;
```

Father theme:structural morphology

DPSDK_ADD_RELATION_NOTIFY

```
Typedef Struct
{
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN];          //channel ID
    DPSDK_INT32 INum;                                //The number of associated video channels
    DPSDK_LINKED_CHANNEL LinkChannel[DPSDK_LINKED_CHANNEL_SIZE];
    DPSDK_INT32 ILocation;                           //Center video or device video
} DPSDK_ADD_RELATION_NOTIFY;
```

Father theme:structural morphology

DPSDK_MODIFY_RELATION_NOTIFY

```
Typedef Struct
{
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; //passagewayID
    DPSDK_INT32 INum;                             //The number of associated video channels
    DPSDK_LINKED_CHANNEL LinkChannel[DPSDK_LINKED_CHANNEL_SIZE];
    DPSDK_INT32 ILocation;                        //Center video or device video
} DPSDK_MODIFY_RELATION_NOTIFY;
```

Father theme:structural morphology

DPSDK_DELETE_RELATION_NOTIFY

```
Typedef Struct
{
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; //channel ID
} DPSDK_DELETE_RELATION_NOTIFY;
```

Father theme:structural morphology

DPSDK_BITMAP_INFO_NOTIFY

Map change notification message structure

```
Typedef Struct
{
    DPSDK_INT32                    IOperateType;                                    //Operation type:1=Increase,2=Modification,3=delete
    DPSDK_CHAR SzMapID[DPSDK_DEVICE_ID_LEN];          //MapID
    DPSDK_CHAR SzMapName[DPSDK_NAME_LEN];             //Map name
    DPSDK_CHAR SzMapPath[DPSDK_URL_LEN];              //Map path
    DPSDK_CHAR SzParentID[DPSDK_DEVICE_ID_LEN];       //Superior ID
    DPSDK_INT32 IStatus;                         //Enable state,0=Discontinuation,1=Enable
} DPSDK_BITMAP_INFO_NOTIFY;
```

Father theme:structural morphology

DPSDK_TVWALL_NOTIFY

Television wall notice <increase/modify>

```
Typedef Struct
{
    DPSDK_UINT32 UiTvWallId;                        // TV wall ID
    DPSDK_CHAR SzTvWallName[DPSDK_TVWALL_NAME_LEN];       // TV wall name
    DPSDK_CHAR SzOwnerCode[DPSDK_TVWALL_OWNERCODE_LEN];    // Coding of TV
wall
    DPSDK_INT32 IStates;                        // Television wall state 0=Not enabled 1=Enable
    DPSDK_CHAR SzTvWallXml[1];                        // TV wall XML
}DPSDK_TVWALL_NOTIFY;
```

Father theme:structural morphology

DPSDK_SMARTTRACKOBJECT_NOTIFY

Gun ball master tracking subscription notification

```
Typedef Struct
{
    DPSDK_CHAR SzDevId[DPSDK_DEVICE_ID_LEN];   // device ID
    DPSDK_INT32 IGroup;                        //Subscribe group
    DPSDK_INT32 ISubscribeID;                  // Server-side subscriptionID
    DPSDK_INT32 ISlaveID;                      // Trace group from the machine sequence number
    DPSDK_CHAR SzClass[MASTERSALVE_CLASS_LEN]; // Algorithm scheme type
    DPSDK_INT32 IObjectID;                     // Algorithm target ID
}DPSDK_SMARTTRACKOBJECT_NOTIFY;
```

Father theme:structural morphology

DPSDK_EVENT_PARAM

Event callback parameter structure

```
Typedef Struct
{
     DPSDK_INT32 ISessionID; // Conversation ID
     DPSDK_LPVOID PBuf;      // Message structure
     DPSDK_UINT32 UiBufLen; // Message structure  length
}DPSDK_EVENT_PARAM;
```

Father theme:structural morphology

MEDIA_ENCHANGE

```
typedef struct MEDIA_ENCHANGE
{
    int         iWidth;         // Image Width
    int         iHeight;        // Image Height
}MEDIA_ENCHANGE_T;
```

Father theme:structural morphology

MEDIA_DISPLAY

```
typedef struct MEDIA_DISPLAY
{
    long        lPort;      // Channel No.
    char*       pBuf;       // Return Image Data
    long        lSize;      // User Data
    long        lWidth;     // Image Width in Pixel
    long        lHeight;    // Image Height
    long        lStamp;     // Time Mark Info in Millisecond
    long        lType;      // Data Type, T_RGB32,T_UYVY
}MEDIA_DISPLAY_T;
```

Father theme:structural morphology

FileStoreInfo

```
typedef struct FileStoreInfo
{
    uint32_t        uiStoreLen;         // Recording Length
    time_t          tBeginTime;         // Start Time
    time_t          tEndTime;           // End Time
     std::string     strFile;           // Full Path of Recording File
    FileStoreInfo()
    {
        uiStoreLen = 0;
        tBeginTime = 0;
        tEndTime = 0;
    }
}FileStoreInfo_t;
```

Father theme:structural morphology

DPSDK_PICDATA_CALLBACK

Picture data callback function

```
Typedef DPSDK_INT32 (DPSDK_CALL *DPSDK_PICDATA_CALLBACK)  (DPSDK_INT32 ISession,
     DPSDK_CHAR* PData,
     DPSDK_INT32 IDataLen,
     DPSDK_LPVOID PUserParam,
     DPSDK_INT32 IPicEventType
);
```

Father theme:structural morphology

DPSDK_CAR_SURVEY_ALARM

Bayonet dispatched alarm

```
Typedef Struct
{
    DPSDK_CHAR SzChannelID[DPSDK_CHANNEL_ID_LEN];      // Alarm channel ID
    DPSDK_TIMET LAlarmTime;                           // Alarm time (unit: seconds))
    DPSDK_CHAR SzPlateNumber[DPSDK_PLATE_NUMBER_LEN]; // Number plate number
    DPSDK_INT32 IPlateColor; // License plate color coding 99-Unidentified,0-Blue,1-
Yellow,2-White,3-黑Color,100-Other colors
    DPSDK_INT32 ISurveyType; // Control  type encoding 1-Overspeed vehicles,2-Theft of
vehicles,3-Traffic accident vehicle,4-suspicion
Vehicles,5-Intercepting vehicles,6-Check and check,7-Observation and tracking,8-High risk
vehicles,9-White list,11-Special abnormal vehicle,12-Yellow label vehicle
    DPSDK_INT32 ICarColor; // Vehicle color coding 0-White,1-Black,2-Red,3-Yellow,4-Silver
gray，5-Blue,6-Green,7-Orange,8-Violet，9-Green,10-Pink,11-Brown,99-Unidentified,100-Other
    DPSDK_CHAR SzImgPath[DPSDK_URL_LEN]; // snapshot the image download path
}DPSDK_CAR_SURVEY_ALARM;
```

Father theme:structural morphology

DPSDK_LOG_LEVEL_TYPE

Log rank The higher the level is, the less the content of the output is

```
Typedef Enum
{
    LOG_LEVEL_DEBUG = 2,   // debugging Do not print normally for debugging and use
    LOG_LEVEL_INFO = 4,    // information
    LOG_LEVEL_WARN = 5,    // Notice
    LOG_LEVEL_ERR = 6,     // error
}DPSDK_LOG_LEVEL_TYPE;
```

Father theme:structural morphology

DPSDK_COMPRESS_TYPE

Compression method

```
Typedef Enum
{
    COMPRESS_DISABLE = 0, // No use of compression
    COMPRESS_DEFAULT = 1, // Using the default compression method
}DPSDK_COMPRESS_TYPE;
```

Father theme:structural morphology

DPSDK_LOGIN_PARAM

Log rank The higher the level is, the less the content of the output is

```
Typedef Struct
{
    DPSDK_BOOL BDomainUser;                    // Whether or not domain login
    DPSDK_CHAR SzUserName[DPSDK_NAME_LEN];     // User name
    DPSDK_CHAR SzPWD[DPSDK_PWD_LEN];           // The password is plaintext, the login
type is0(basic account), can not be empty
    DPSDK_IP StruIP;                           // logon serverIP
    DPSDK_UINT32 UiPort;                       // Login server port
    DPSDK_CHAR SzMACAddress[DPSDK_MACADDRESS_LEN]; // MACaddress
    DPSDK_CHAR  SzIMEI[DPSDK_IMEI_LEN];                  // Check code for mobile client
landing platform
    DPSDK_UINT32   UiClientType;                              // Client type: Reference
 DPSDK_CLIENT_TYPE
    DPSDK_CHAR SzReserve[32];                  // Reserved field
}DPSDK_LOGIN_PARAM;
```

Father theme:structural morphology

DPSDK_QUERY_ORG_INFO

Organization query conditions

```
Typedef Struct
{
    DPSDK_CHAR  SzOrgCode[DPSDK_ORG_CODE_LEN];      // Oorganization  code is the length which is the default query root organization
    DPSDK_INT32   ISubNodeType;                                        //   Query   subnode   type   See DPSDK_SUB_CODE_TYPEDefinition
    DPSDK_INT32 IContainDevice;                // Include device 0=Do not contain 1=Contain
    DPSDK_INT32 IChannelTypeList[1];            // Channel type set that needs to be querying  See DPSDK_DEV_UNIT_TYPEDefinition
}DPSDK_QUERY_ORG_INFO;
```

Father theme:structural morphology

DPSDK_DataCallback

Data synchronization callback for upper copy data

```
Typedef DPSDK_VOID (DPSDK_CALL * DPSDK_DataCallback) (
    DPSDK_INT32 IDataType,
    DPSDK_VOID* PDataBuf,
    DPSDK_UINT32 PDataBuf,
    DPSDK_VOID* PUserData
);
```

Father theme:structural morphology

DPSDK_DATA_TYPE

Data type definition of data synchronization callback function

```
Typedef Enum
{
    // Organization, equipment
    DPSDK_DATA_ORG_INFO = 1,            // Organizational data A detailed view of the structure DPSDK_ORG_INFO
    DPSDK_DATA_DEVICE_INFO = 2,             // Device data A detailed view of the structure DPSDK_DEV_ALL_INFO_LIST
    DPSDK_DATA_COLLECT_ORG_INFO = 3,      // Collection tree A detailed view of the structure DPSDK_COLLECTION_ORG_INFO
    DPSDK_DATA_DEVICE_LAYERED = 4, // Hierarchical acquisition of device tree A detailed view of the structure DPSDK_LAYERED_RESULT_LIST
    DPSDK_DATA_DEVICE_LIST_BY_ORG = 5, // Device data A detailed view of the structure DPSDK_DEV_ALL_INFO_LIST
    DPSDK_DATA_ALL_ORG_INFO = 6,         // All organizational data A detailed view of the structure DPSDK_ALL_ORG_INFO
    //TV wall
    DPSDK_DATA_TVWALL_TASK_INFO = 2000, // Get TV wall task information A detailed view of the structure DPSDK_TVWALL_TASK_INFO
    DPSDK_DATA_GET_TVWALL_PROJECT_FILE = 2001, // Access to the TV wall plan file structure DPSDK_TVWALL_TASK_INFO_LIST
} DPSDK_DATA_TYPE;
```

Father theme:structural morphology

DPSDK_ORG_INFO

Basic Organizational Data

```
typedef struct DPSDK_ORG_INFO_T
{
    DPSDK_ORG_BASE_INFO struOrgBaseInfo;                        // Organization Info
    DPSDK_INT32 iDevNum;                                        // Number of Child Device
    DPSDK_ORG_SUB_DEV_INFO* pDevList;                           // List of Child Device
    DPSDK_INT32 iChannelNum;                                    // Number of Sub-channel
    DPSDK_ORG_SUB_CHANNEL_INFO* pChannelList;                   // List of Sub-channel
    DPSDK_INT32 iOrgNum;                                        // Number of Sub-organization
    DPSDK_ORG_INFO_T* pOrgList;                                 // List of Sub-organization
}DPSDK_ORG_INFO;
```

Father theme:structural morphology

DPSDK_DEV_ALL_INFO_LIST

Device list

```
Typedef Struct
{
     DPSDK_INT32 IDevNum;                    // Number of devices
     DPSDK_DEV_ALL_INFO* PDevAllInfoList;   // Device list data
}DPSDK_DEV_ALL_INFO_LIST;
```

Father theme:structural morphology

DPSDK_ALL_ORG_INFO

All Organizational Info (Recursion Tree)

```
typedef struct DPSDK_ALL_ORG_INFO_T
{
    DPSDK_SINGLE_ORG_INFO struOrgBaseInfo;                    // Info about Organization at
this Level
    DPSDK_INT32 iOrgNum;                                      // Number of Sub-organizations
    DPSDK_ALL_ORG_INFO_T* pOrgList;                           // List of Sub-organizations
}DPSDK_ALL_ORG_INFO;
```

Father theme:structural morphology

DPSDK_GET_DEVICE_LAYERED_PARAM

Hierarchical acquisition of device tree request parameters

```
Typedef Struct
{
    DPSDK_CHAR SzID[DPSDK_ORG_CODE_LEN]; // node ID
    DPSDK_INT32   INodeType;                              //  DPSDK_NODE_TYPE  Definition
1:Organization,2:Equipment,3:passageway
    DPSDK_INT32 IOrgType;             // 1: Basic organization
    DPSDK_INT32 IShowDev;              // 0: No device nodes are needed,1: Need device node
    DPSDK_INT32    IDeep;                                                         //    2:
organization+Equipment,3Organization+equipment+passageway
    DPSDK_INT32 ICategoryNum;           // Device large class list length
    DPSDK_INT32* PCategoryList;         // Device large list
    DPSDK_INT32 IChannelTypeNum;         // Channel type list length
    DPSDK_INT32* PChannelTypeList;           // Channel type set that needs to be querying See
DPSDK_DEV_UNIT_TYPE Definition
    DPSDK_CHAR SzKeyWord[DPSDK_MEMO_LEN]; // Search keywords
}DPSDK_GET_DEVICE_LAYERED_PARAM;
```

Father theme:structural morphology

DPSDK_PAGE_INFO

Paging information

```
Typedef Struct
{
    DPSDK_UINT32 UiPage;     // The current paging, from 1start
    DPSDK_UINT32 UiPageSize; // pagesize
}DPSDK_PAGE_INFO;
```

Father theme:structural morphology

DPSDK_LAYERED_RESULT_LIST

Obtain Layered List of Device Tree Returned Result

```
typedef struct
{
    DPSDK_INT32 iResultNum;                          // List Length
    DPSDK_LAYERED_RESULT* pResultList;                        // Result List
}DPSDK_LAYERED_RESULT_LIST;
```

Father theme:structural morphology

DPSDK_REALPLAY_PARAM

Unicast video parameters

```
Typedef Struct
{
     DPSDK_MEDIA_BASE_PARAM StruMediaBaseParam; // Basic video parameters
     DPSDK_MEDIA_CALLBACK StruMediaCallBack;    // Video callback structure
     //Transcoding parameter
     DPSDK_INT32 IUsedVcs; // Whether the tag needs to pass throughVCSTranscoding.0It means that there is no need for transcoding;1It means that transcoding is required.
     DPSDK_CHAR  SzVideoCode[DPSDK_VIDEO_PARAM_LEN];    // Video coding format, reference video coding format to define strings
     DPSDK_CHAR SzResolution[DPSDK_VIDEO_PARAM_LEN]; // Code stream resolution, reference stream resolution definition string
     DPSDK_INT32 IFps;                              // Frame rate
     DPSDK_INT32 IBps;                              // Bit stream code stream
}DPSDK_REALPLAY_PARAM;
```

Father theme:structural morphology

DPSDK_MULITVIEW_REALPLAY_PARAM

Multi screen preview video parameters

```
Typedef Struct
{
    DPSDK_MEDIA_BASE_PARAM StruMediaBaseParam;   // Basic video parameters
    DPSDK_MEDIA_CALLBACK StruMediaCallBack;      // Video callback structure
    DPSDK_CHAR SzTrackId[DPSDK_VIDEO_PARAM_LEN]; // track ID
    //Multi picture preview
    DPSDK_INT32 IScreenNum;                      // Multi picture segmentation number
    DPSDK_INT32 IStartChnlIndex;                 // Starting channel
}DPSDK_MULITVIEW_REALPLAY_PARAM;
```

Father theme:structural morphology

DPSDK_MULITCAST_REALPLAY_PARAM

Multicast video parameters

```
Typedef Struct
{
    DPSDK_MEDIA_BASE_PARAM StruMediaBaseParam;    // Basic video parameters
    DPSDK_MEDIA_CALLBACK StruMediaCallBack;       // Video callback structure
    DPSDK_CHAR SzTrackId[DPSDK_VIDEO_PARAM_LEN]; // track ID
}DPSDK_MULITCAST_REALPLAY_PARAM;
```

Father theme:structural morphology

DPSDK_DECODE_TYPE

Decode type

```
Typedef Enum
{
    DPSDK_DECODE_SW = 0,       //CPUDecode
    DPSDK_DECODE_HW = 1,       //GPUDecode
    DPSDK_DECODE_HW_FAST = 2, //GPUFast decoding
}DPSDK_DECODE_TYPE;
```

Father theme:structural morphology

DPSDK_STREAM_MODE

Playback mode

```
Typedef Enum
{
    DPSDK_STREAM_REAL_MODE = 0,   // Real time priority mode
    DPSDK_STREAM_SMOOTH_MODE = 1, // Fluency priority model
    DPSDK_STREAM_POISE_MODE = 2, // Equilibrium priority model
    DPSDK_STREAM_CUSTOM_MODE = 3, // Custom priority mode
}DPSDK_STREAM_MODE;
```

Father theme:structural morphology

DPSDK_VIDEO_LOCK_TYPE

```
typedef enum
{
    DPSDK_VIDEO_CMD_LOCK        = 0,            // Lock the current camera.
    DPSDK_VIDEO_CMD_UNLOCK_ONE = 1,             // Unlock the current camera.
}DPSDK_VIDEO_LOCK_TYPE;
```

Father theme:structural morphology

DPSDK_RECORD_STATUS_INFO

Channel video information

```
Typedef Struct
{
    DPSDK_CHAR SzChannelCode[DPSDK_CHANNEL_ID_LEN]; // Channel coding
    DPSDK_INT32 IChannelSeq;                 // Channel number
    DPSDK_INT32   IRecordStatus;                                     //   Video   status
SeeDPSDK_RECORD_STATUSDefinition
    DPSDK_INT32 IFlow;                       // Average flow rate (Kbps)
    DPSDK_INT32 IStreamType;                                  // Code stream type See
DPSDK_STREAM_TYPEDefinition
    DPSDK_INT32 IUsedCapacity;               // Used storage capacity
}DPSDK_RECORD_STATUS_INFO;
```

Father theme:structural morphology

DPSDK_QUERY_RECORD_PARAM

Inquire video information

```
Typedef Struct
{
    DPSDK_CHAR SzCameraId[DPSDK_CHANNEL_ID_LEN]; // channel ID
    DPSDK_STREAM_TYPE IStreamType;              // Code stream type
    DPSDK_SOURCE_TYPE ISourceType;              // Video source type
    DPSDK_RECORD_TYPE IRecordType;              // Video type
    DPSDK_TIMET TBeginTime;                     // Start time
    DPSDK_TIMET TEndTime;                       // End time
}DPSDK_QUERY_RECORD_PARAM;
```

Father theme:structural morphology

DPSDK_RECORD_INFO_LIST

Video information

```
Typedef Struct
{
    DPSDK_CHAR SzCameraId[DPSDK_CHANNEL_ID_LEN];  //  channel  ID
    DPSDK_UINT32 IRetCount;                            // Return the number of records, that is record
number of recorded video records
    DPSDK_SINGLE_RECORD_INFOO StruSingleRecord[1]; // Video recording information
}DPSDK_RECORD_INFO_LIST;
```

Father theme:structural morphology

DPSDK_QUERY_RECORD_DATE_PARAM

Inquire the date of video

```
Typedef Struct
{
    DPSDK_CHAR SzCameraId[DPSDK_CHANNEL_ID_LEN]; // channel ID
    DPSDK_SOURCE_TYPE  ISourceType;            // Video source type
    DPSDK_INT32 IYear;                         // year
    DPSDK_INT32 IMonth;                        // month
}DPSDK_QUERY_RECORD_DATE_PARAM;
```

Father theme:structural morphology

DPSDK_RECORD_DATE_INFO

```
Typedef Struct
{
    DPSDK_INT32 RecordDays[DPSDK_DAY_IN_MONTH]; // The record is video taped.
0start for the first day
}DPSDK_RECORD_DATE_INFO;
```

Father theme:structural morphology

DPSDK_LOCK_RECORD_FILE_PARAM

Lock the video file

```
Typedef Struct
{
    DPSDK_CHAR SzCameraId[DPSDK_CHANNEL_ID_LEN];        // Camera ID
    DPSDK_CHAR SzFilename[DPSDK_RECORD_FILE_NAME_LEN]; // The name of the video (different manufacturers are different in the identification of the documents)
}DPSDK_LOCK_RECORD_FILE_PARAM;
```

Father theme:structural morphology

DPSDK_LOCK_RECORD_FILE_RESULT

Locking or unlocking the results of video files

```
Typedef Struct
{
    DPSDK_INT32 ILockNum; // Lock number
}DPSDK_LOCK_RECORD_FILE_RESULT;
```

Father theme:structural morphology

DPSDK_UNLOCK_RECORD_FILE_PARAM

Unlocking video files

```
Typedef Struct
{
    DPSDK_CHAR SzCameraId[DPSDK_CHANNEL_ID_LEN];        // Camera ID
    DPSDK_CHAR SzFilename[DPSDK_RECORD_FILE_NAME_LEN]; // The name of the video (different manufacturers are different in the identification of the documents)
    DPSDK_BOOL BForce;                                  // Whether or not compulsory
}DPSDK_UNLOCK_RECORD_FILE_PARAM;
```

Father theme:structural morphology

DPSDK_PLAYBACK_BY_TIME_PARAM

Playback parameter

```
Typedef Struct
{
    DPSDK_MEDIA_CALLBACK StruMediaCallBack; // Video callback function
    HCWND PHWnd;                        // Window handle
    DPSDK_INT32   IDirection;                                    //   Playback   direction   See
DPSDK_PLAY_DIRECTIONDefinition
    DPSDK_CHAR SzCodeId[DPSDK_CHANNEL_ID_LEN]; // passageway ID Or equipment
ID
    DPSDK_TIMET TBeginTime;                 // start time(time stamp)
    DPSDK_TIMET TPlayTime;                  // Start playing time
    DPSDK_TIMET TEndTime;                   // End time(time stamp)
    DPSDK_INT32 IRecordSource;              // Video source, see see DPSDK_SOURCE_TYPE
    DPSDK_INT32   IStreamType;                              // Code stream type, see  see
DPSDK_STREAM_TYPE
    DPSDK_INT32 IRecordType;                // Video type, see see DPSDK_RECORD_TYPE
}DPSDK_PLAYBACK_BY_TIME_PARAM;
```

Father theme:structural morphology

DPSDK_PLAYBACK_BY_FILE_PARAM

Playback parameter

```
Typedef Struct
{
    DPSDK_MEDIA_CALLBACK StruMediaCallBack; // Video callback function
    HCWND PHWnd;                            // Window handle
    DPSDK_INT32   IDirection;                                    //   Playback   direction   See
DPSDK_PLAY_DIRECTIONDefinition
    DPSDK_CHAR SzCodeId[DPSDK_CHANNEL_ID_LEN]; //passageway ID
    DPSDK_TIMET TBeginTime;                 //start time
    DPSDK_TIMET TEndTime;                   //End time
    DPSDK_INT32 IRecordSource;              //Video source, see see DPSDK_SOURCE_TYPE
    DPSDK_UINT64 USSId;                     //Storage service (IDReturn to the query)
    DPSDK_UINT64 UFileHandle;               //File handle(Return to the  query)
    DPSDK_CHAR SzDiskId[DPSDK_DISDK_ID_LEN];   //disk (IDReturn to the query)
    DPSDK_CHAR   SzFilename[DPSDK_RECORD_FILE_NAME_LEN];   //The   name   of the
video (different manufacturers are different in the identification of the documents)
}DPSDK_PLAYBACK_BY_FILE_PARAM;
```

Father theme:structural morphology

DPSDK_PLAYBACK_SEEK_PARAM

Jump playback parameters

```
Typedef Struct
{
    DPSDK_TIMET TBeginTime;     //start time
    DPSDK_TIMET TEndTime;       //End time
    DPSDK_PLAYBACK_SPEED ISpeed; //Playback speed
    DPSDK_INT32       IDirection;                              //Playback       direction       See
DPSDK_PLAY_DIRECTIONDefinition
}DPSDK_PLAYBACK_SEEK_PARAM;
```

Father theme:structural morphology

DPSDK_PLAYBACK_SPEED

Video playback speed

```
Typedef Enum
{
    DPSDK_PB_NORMAL = 1024,
    DPSDK_PB_NORMAL_FAST2 = DPSDK_PB_NORMAL * 2,
    DPSDK_PB_NORMAL_FAST4 = DPSDK_PB_NORMAL * 4,
    DPSDK_PB_NORMAL_FAST8 = DPSDK_PB_NORMAL * 8,
    DPSDK_PB_NORMAL_FAST16 = DPSDK_PB_NORMAL * 16,
    DPSDK_PB_NORMAL_SLOW2 = DPSDK_PB_NORMAL / 2,
    DPSDK_PB_NORMAL_SLOW4 = DPSDK_PB_NORMAL / 4,
    DPSDK_PB_NORMAL_SLOW8 = DPSDK_PB_NORMAL / 8,
    DPSDK_PB_NORMAL_SLOW16 = DPSDK_PB_NORMAL / 16,
}DPSDK_PLAYBACK_SPEED;
```

Father theme:structural morphology

DPSDK_PTZOPERATE_STARTREMOTERECORD_PARAM

Open the manual video parameters

```
Typedef Struct
{
     DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; // channel ID
     DPSDK_INT32  IStreamType;        // Code stream type (code stream type:1-Main stream, 2-
Auxiliary code stream
     DPSDK_INT32 IRecordDuration; // Video length(default 12*3600秒)
}DPSDK_PTZOPERATE_STARTREMOTERECORD_PARAM;
```

Father theme:structural morphology

DPSDK_PTZOPERATE_REMOTERECORD_RESULT

open/Stop the result of manual video

```
Typedef Struct
{
    DPSDK_INT32 IPlanId;                // plan ID
    DPSDK_CHAR SzNow[DPSDK_PTZ_TIME_LEN]; // current time(time stamp)
}DPSDK_PTZOPERATE_REMOTERECORD_RESULT;
```

Father theme:structural morphology

DPSDK_PTZOPERATE_STOPREMOTERECORD_PARAM

Turn off the manual video parameters

```
Typedef Struct
{
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; // channel ID
    DPSDK_INT32 IStreamType; // Code stream type (code stream type:1-Main stream, 2-
Auxiliary code stream
}DPSDK_PTZOPERATE_STOPREMOTERECORD_PARAM;
```

Father theme:structural morphology

DPSDK_DOWNLOAD_BY_TIME_PARAM

Download parameters by time

```
Typedef Struct
{
    DPSDK_EVENT_DOWNLOAD_CALLBACK FEventCallBack;     // event callbacks
    DPSDK_LPVOID PEventUserData;                      // Event callback user data
    DPSDK_CHAR SzChannelID[DPSDK_CHANNEL_ID_LEN];     //passageway ID
    DPSDK_SOURCE_TYPE ISourceType;                    //Video source
    DPSDK_STREAM_TYPE IStreamType;                    //Code stream type
    DPSDK_RECORD_TYPE IRecordType;                    //Video type
    DPSDK_TIMET TBeginTime;                           //start time
    DPSDK_TIMET TEndTime;                             //End time
    DPSDK_CHAR SzChannelName[DPSDK_CHANNEL_NAME_LEN]; //Channel name
    DPSDK_CHAR SzDownloadPath[DPSDK_FILE_PATH_LEN];   //Downloading path
    DPSDK_RECORD_FILE_NAME_RULE INameRule;            //Download file naming rules
    DPSDK_CHAR SzDownloadFileName[DPSDK_FILE_PATH_LEN]; //Download the name of
the file, if it is empty, use INameRuleThe defined rules are generated, not empty,Neglecting
INameRule, szDownloadPath, szChannelNamefield
    DPSDK_INT32  ISplitFileSize;                      //Division of file size, unit MB,   0Non
segmentation
    DPSDK_DOWNLOAD_RECORD_FILE_FORMAT IFileFormat;    //Download file format
}DPSDK_DOWNLOAD_BY_TIME_PARAM;
```

Father theme:structural morphology

DPSDK_DOWNLOAD_BY_FILE_PARAM

Download parameters by file

```
Typedef Struct
{
    DPSDK_EVENT_DOWNLOAD_CALLBACK_CALLBACK    FEventCallBack;    //    event callbacks
    DPSDK_LPVOID PEventUserData;                // Event callback user data
    DPSDK_CHAR SzChannelID[DPSDK_CHANNEL_ID_LEN]; //channel ID
    DPSDK_SOURCE_TYPE ISourceType;              //Video source
    DPSDK_TIMET TBeginTime;                     //start time
    DPSDK_TIMET TEndTime;                       //End time
    DPSDK_UINT64 USSId;                         //Storage service (IDReturn to the query)
    DPSDK_UINT64 UFileHandle;                   //File handle(Return to the query)
    DPSDK_CHAR SzDiskId[DPSDK_DISDK_ID_LEN];     //disk (IDReturn to the query)
    DPSDK_CHAR   SzFilename[DPSDK_RECORD_FILE_NAME_LEN];  //The   name   of the video (different manufacturers are different in the identification of the documents)
    DPSDK_CHAR SzChannelName[DPSDK_CHANNEL_NAME_LEN];   //Channel name
    DPSDK_CHAR SzDownloadPath[DPSDK_FILE_PATH_LEN];    //Downloading path
    DPSDK_RECORD_FILE_NAME_RULE INameRule;            //Download file naming rules
    DPSDK_CHAR SzDownloadFileName[DPSDK_FILE_PATH_LEN]; //Download the name of the file, if it is empty, use INameRuleThe defined rules are generated, not empty,Neglecting INameRule, szDownloadPath, szChannelNamefield
    DPSDK_INT32  ISplitFileSize;                        //Division of file size, unit MB,   0Non segmentation
    DPSDK_DOWNLOAD_RECORD_FILE_FORMAT IFileFormat;      //Download file format
}DPSDK_DOWNLOAD_BY_FILE_PARAM;
```

Father theme:structural morphology

DPSDK_DOWNLOAD_RECORD_INFO

Video download information parameters

```
Typedef Struct
{
     DPSDK_INT32 IDownloadID;
     DPSDK_INT32 IFileID;
     DPSDK_INT32 IDownloadMode;
     DPSDK_INT32 IRecordSource;
     DPSDK_INT32 IRecordType;
     DPSDK_INT32 IStreamType;
     DPSDK_UINT64 UiCurFileSize;
     DPSDK_UINT64 UiPrevFileSize;
     DPSDK_TIMET TBeginTime;
     DPSDK_TIMET TEndTime;
     DPSDK_INT32 IDownloadState;
     DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN];
     DPSDK_UINT64 UiFileHandle;
     DPSDK_CHAR SzDiskId[DPSDK_DISDK_ID_LEN];
     DPSDK_INT32 IDownloadStatus;
     DPSDK_INT32 IFileCount;
     DPSDK_CHAR SzDownloadFileName[1][DPSDK_FILE_PATH_LEN];
}DPSDK_DOWNLOAD_RECORD_INFO;
```

Father theme:structural morphology

DPSDK_FILE_STORE_LIST

Video file list

```
Typedef Struct
{
    DPSDK_SIZET UlTotal;                  //Total number of video files
    DPSDK_FILE_STORE_INFO StruFileList[1];   //Video file list
}DPSDK_FILE_STORE_LIST;
```

Father theme:structural morphology

DPSDK_PIC_FORMAT

Capture image format

```
Typedef Enum
{
    DPSDK_PIC_FORMAT_BMP = 0, // BMP type
    DPSDK_PIC_FORMAT_JPEG = 1, // JPEG type
}DPSDK_PIC_FORMAT;
```

Father theme:structural morphology

DPSDK_CONVERT_BMP

Picture turn BMPformat

```
{
    DPSDK_CHAR*   PBuf;                          // Image data pointer
    DPSDK_LONG    LSize;                         // Image data size
    DPSDK_LONG LWidth;                           // Image width
    DPSDK_LONG LHeight;                          // Image height
    DPSDK_LONG LType;                            // image type
    DPSDK_CHAR SzFileName[DPSDK_FILE_PATH_LEN]; // File name to be saved.It is best
to BMPAs a file extension
}DPSDK_CONVERT_BMP;
```

Father theme:structural morphology

DPSDK_MHFPTZ_INIT_PARAM

Parameters of fish ball linkage initialization channel

```
Typedef Struct
{
    DPSDK_INT32 IHimgWidth;     // From the camera image width
    DPSDK_INT32 IHimgHeight;    // From the camera image
    DPSDK_INT32 *arriZoomList; // Ball multiplier table
    DPSDK_INT32 IZoomListSize; // The number of multiple tables of the ball machine
}DPSDK_MHFPTZ_INIT_PARAM;
```

Father theme:structural morphology

DPSDK_FISH_TYPE

Fish eye opening type

```
Typedef Enum
{
    DPSDK_FISH_CORRECT = 0,          // Fish eye correction
    DPSDK_FISH_CORRECT_AND_LINK = 1, // Fish ball linkage and correction
    DPSDK_FISH_LINK = 2,             // Fish ball linkage
}DPSDK_FISH_TYPE;
```

Father theme:structural morphology

DPSDK_FISH_OPTPARAM

Fish eye parameters

```
Typedef Struct
{
    DPSDK_FISH_SIZE StruMainStreamSize; // The original width of the main stream is high, and when the incoming resolution is different from it, it is shown that the auxiliary code stream is the scaling of this resolution.
    DPSDK_INT32 IOriginX; // The center abscissa of the fish eye circle in the input image, Normalization to 0-8192coordinate system
    DPSDK_INT32 IOriginY; // The center longitudinal coordinates of the fish eye circle in the input image, Normalization to 0-8192coordinate system
    DPSDK_INT32 IRadius; // The radius of the fish eye circle in the input image, Normalization to 0-8192coordinate system
    DPSDK_INT32 ILensDirection;        // Angle of rotation, Q7format, Range 0-360*128, Generally matched 0
    DPSDK_UINT32 UiMainMountMode;        // Master installation mode,See DPSDK_FISH_MOUNTMODE
    DPSDK_UINT32 UiMainCalibrateMode; // Image master correction model,See DPSDK_FISH_SHOWMODES
    DPSDK_FISH_MODEINITPARAM StruModeInitParam; // The external afferent mode initializes each picture information, which is suitable for the mode switch to restore to the last state.,
    DPSDK_FISH_OUTPUTFORMAT* POutputFormat; // Output image information
    DPSDK_MHFPTZ_CONFIGPARAM* PConfigParam; // Fish ball linkage configuration parameters
    DPSDK_INT32 IEnableAutoContrast;        //IN OUT/*Open automatic contrast, 0Close, 1open, This function increases the time consuming of the algorithm, It needs good performance The machine is recommended to open* /
    DPSDK_INT32 IAlphaHistogram; //IN OUT *histogram IIRStrength 0-255, default 128, The bigger the current frame of reference* /
    DPSDK_INT32 IAlphaGray; //IN OUT/*Grayscale tensile strength 0-255, default 245, The bigger the weak contrast* /
    DPSDK_FISH_SIZE StruCaptureSize; //OUT/*Corresponding to the current mode of screen resolution* /
    DPSDK_INT32 IMhfptzIndex;        //IN *Serial number of fish ball linkage ball machine 0,1,2.*/... / /At present, the client supports only one fish and one ball, so this parameter Default filling 0There will be an abnormality
    DPSDK_INT32 ArriReserved[1]; // Reserved bytes
}DPSDK_FISH_OPTPARAM;
```

Father theme:structural morphology

DPSDK_FISH_UPDATE_PARAM

```
Typedef Struct
{
    DPSDK_INT32 ICircleX;     // The center abscissa of the fish eye circle in the input image
    DPSDK_INT32 ICircleY;     // The center longitudinal coordinates of the fish eye circle in the input image
    DPSDK_INT32 IRadius;      // The radius of the fish eye circle in the input image
    DPSDK_LONG LWidthRatio;   // The original width of the main stream
    DPSDK_LONG LHeightRatio;  // The original height of the main stream
}DPSDK_FISH_UPDATE_PARAM;
```

Father theme:structural morphology

DPSDK_FISH_EPTZPARAM

EPTZparameter

```
Typedef Struct
{
     DPSDK_UINT32  UiPtzCmd;                    // The operation of the cloud platform shows that 
DPSDK_FISH_EPTZCMD Definition
     DPSDK_INT32  IWinId;                 // To carry onEptzWindow number, upper left  cornerWinId
为0Increase from left to right
     DPSDK_INT32 IArg1;
     DPSDK_INT32  IArg2;
     DPSDK_INT32  IArg3;
     DPSDK_INT32  IArg4;
     DPSDK_INT32  IArg5;
     DPSDK_INT32 IArg6;
     DPSDK_INT32 ArriReserved0[6]; // Reserved bytes
     DPSDK_LPVOID PParam;            // Fish ball linkage
     DPSDK_LPVOID PResult;
     DPSDK_LPVOID PArg;
     DPSDK_INT32 ArriReserved1[7]; // Reserved bytes
}DPSDK_FISH_EPTZPARAM;
```

Father theme:structural morphology

DPSDK_FISH_PARAMS

Fish eye parameters

```
Typedef Struct
{
    DPSDK_INT32 IFitmode;
    DPSDK_INT32 IDisplaymode;
    DPSDK_INT32 IWidth;
    DPSDK_INT32 IHeight;
    DPSDK_INT32 IOriginX;
    DPSDK_INT32 IOriginY;
    DPSDK_INT32 IRadius;
    DPSDK_INT32 IWidthRatio;
    DPSDK_INT32 IHeightRatio;
    DPSDK_SUBORDINATE_CAMCONFIGPARAM StruSubCamConfigParam;
}DPSDK_FISH_PARAMS;
```

Father theme:structural morphology

DPSDK_RECT

Increase the device notification

```
Typedef Struct
{
     DPSDK_LONG Left;
     DPSDK_LONG Top;
     DPSDK_LONG Right;
     DPSDK_LONG Bottom;
}DPSDK_RECT;
```

Father theme:structural morphology

DPSDK_IVSE_INFO

Input parameters of video enhancement algorithm

```
Typedef Struct
{
     DPSDK_UINT32 UiFuncType; // Functional options See DPSDK_IVSE_FUNC_TYPE
     DPSDK_IVSE_ROI StruRoi; // ROI To configure
     DPSDK_INT32 IMode;        // 0Representing the picture pattern, 1For video mode, please use
the video when you use it 1Video mode
     DPSDK_INT32 IParam[2];   // Processing parameters, range[1,5]
}DPSDK_IVSE_INFO;
```

Father theme:structural morphology

DPSDK_VAX_BUF_TYPE

Buffer Type

```
typedef enum
{
    DPSDK_VAX_BUF_VIDEO_SRC      = 1,            // Video Source Buffer
    DPSDK_VAX_BUF_AUDIO_SRC      = 2,            // Audio Source Buffer
    DPSDK_VAX_BUF_VIDEO_RENDER   =3,             // Decoded Video Data Buffer
    DPSDK_VAX_BUF_AUDIO_RENDER  = 4,             // Decoded Audio Data Buffer
}DPSDK_VAX_BUF_TYPE;
```

Father theme:structural morphology

DPSDK_IVS_VISIBLE

Intelligence

```
Typedef Enum
{
    DPSDK_IVS_RULE_VISIBLE = 1,    // rule
    DPSDK_IVS_OBJ_VISIBLE = 2,     // Target box
    DPSDK_IVS_LOCUS_VISIBLE = 3,   // trajectory = = The interface is no longer displayed?
}DPSDK_IVS_VISIBLE;
```

Father theme:structural morphology

DPSDK_SPLIT_TRECE_TYPE

4KSplit screen mode

```
Typedef Enum
{
    DPSDK_SPLIT_ORG = 0,     // Basic pattern
    //Four. 4K
    DPSDK_SPLIT_1P_3 = 1,   // 1P+3Pattern
    DPSDK_SPLIT_1P_5 = 2,   // 1P+5Pattern
    //Three.
    DPSDK_SPLIT_3_3_1P = 4, // 1P+3Pattern
    DPSDK_SPLIT_3_6_1P = 5, // 1P+6Pattern
}DPSDK_SPLIT_TRECE_TYPE;
```

Father theme:structural morphology

DPSDK_DISPLAY_RECT

Matrix coordinate of stitching algorithm

```
Typedef Struct
{
    DPSDK_INT32 ILeft;
    DPSDK_INT32 ITop;
    DPSDK_INT32 IRight;
    DPSDK_INT32 IBottom;
    DPSDK_INT32 IX; // Central point coordinates, used for scaling, recording positions at
normal scale and narrowing to the edge.
    DPSDK_INT32 IY; // Central point coordinates, used for scaling, recording positions at
normal scale and narrowing to the edge.
    DPSDK_INT32 IPicWidth; // Picture resolution-width
    DPSDK_INT32 IPicHeight; // Picture resolution-height
}DPSDK_DISPLAY_RECT;
```

Father theme:structural morphology

DPSDK_PICTURE_MONITOR

Port picture monitoring parameter structure

```
Typedef Struct
{
    DPSDK_CHAR SzCodeId[DPSDK_DEVICE_ID_LEN]; // channel ID Or equipment ID
    DPSDK_UINT32 UiDataType;                // Subscription type:1=Vehicle information, 2=Vehicle information+Picture information
    DPSDK_UINT32 UiStreamType;              // Stream type:1=Main stream, 2=Auxiliary code stream
    DPSDK_PICDATA_CALLBACK FBayPicCallBack;      // Card port picture monitoring callback
    DPSDK_LPVOID PUserData;                 // user data
}DPSDK_PICTURE_MONITOR;
```

Father theme:structural morphology

DPSDK_BAYONET_DICTIONARY_TYPE

Card type monitoring dictionary type

```
Typedef Enum
{
    DPSDK_LANE_NUMBER = 19,                //Lane number
    DPSDK_TRAFFIC_DICTIONARY = 20,         //Driving direction
    DPSDK_VEHICLE_PLATE_COLOR = 22,        //License plate color
    DPSDK_VEHICLE_COLOR = 2001,            //Vehicle color
    DPSDK_VEHICLE_TYPE = 2002,             //Vehicle type
    DPSDK_PROVINCE = 2003,                 //Province
    DPSDK_VIOLATION_TYPE = 2006,           //Violation type
    DPSDK_CONTROL_TYPE = 2007,             //Control type
    DPSDK_CONTROL_STATUS = 2008,           //Control state
    DPSDK_CONTROL_LEVEL = 2009,            //Control level
    DPSDK_MONITOR_SPEED = 2010,            //Interval velocity measurement of violation type
    DPSDK_VEHICLE_TRADEMARK = 2016,        //Vehicle trademark
    DPSDK_VEHICLE_PLATE_TYPE = 2017,       //License plate type
    DPSDK_VEHICLE_TYPE_STATUS = 2018,      //Vehicle type state
    DPSDK_VEHICLE_USE_STATUS = 2019,       //Vehicle status
    DPSDK_CREDENTIALS_TYPE = 2027,         //Document type
    DPSDK_PROVINCE_TYPE = 2028,            //Province type
    DPSDK_CITY_TYPE = 2029,                //Urban type
}DPSDK_BAYONET_DICTIONARY_TYPE;
```

Father theme:structural morphology

DPSDK_CONFIRMALARM_PARAM

Alarm confirmation parameter

```
Typedef Struct
{
    DPSDK_CHAR SzAlarmCode[DPSDK_ALARM_ALARMCODE_LEN];          // Alarm code
    DPSDK_CHAR SzHandleUser[DPSDK_ALARM_HANDLERUSER_LEN];          // Handling
human username
    DPSDK_CHAR   SzHandleMessage[DPSDK_ALARM_HANDLEMESSAGE_LEN];          //
Handling opinions
    DPSDK_UINT32 UiEmailRevceiverNumber;                    // Alarm processing mailbox
number
    DPSDK_INT32 IHandleStatus;                    // Processing state (Reference resources
AlarmDealWith_e)
    DPSDK_EMAILADDRESS StruEmailReceiverList[1];                    // Alarm processing
notification mailbox list
}DPSDK_CONFIRMALARM_PARAM;
```

Father theme:structural morphology

DPSDK_QUERYALARM_PARAM

Alarm query parameters

```
Typedef Struct
{
    DPSDK_CHAR  SzBeginTime[DPSDK_ALARM_TIME_LEN];          // Start time of alarm
Yyyy mm dd hh mmss
    DPSDK_CHAR  SzEndTime[DPSDK_ALARM_TIME_LEN];             // The end time of the
alarm Yyyy mm dd hh mm ss
    DPSDK_CHAR SzHandleBeginTime[DPSDK_ALARM_TIME_LEN]; // Alarm processing
start time Yyyy mm dd hh mm ss
    DPSDK_CHAR  SzHandleEndTime[DPSDK_ALARM_TIME_LEN];     // Alarm processing
end time Yyyy mm dd hh mmss
    DPSDK_CHAR SzDeviceId[DPSDK_ALARM_DEVICEID_LEN];    // device ID
    DPSDK_CHAR SzChannelId[DPSDK_ALARM_CHANNELID_LEN]; // channel ID
    DPSDK_CHAR SzOrgId[DPSDK_ALARM_ORGID_LEN];          // Organization node ID
    DPSDK_CHAR SzAlarmId[DPSDK_ALARM_ALARMID_LEN];      // Call the police ID
    DPSDK_CHAR  SzAlarmCode[DPSDK_ALARM_ALARMCODE_LEN];     //  Alarm code
(specifying this condition to ignore other conditions)
    DPSDK_CHAR     SzHandleUser[DPSDK_ALARM_HANDLERUSER_LEN];     //     Alarm
processing person
    DPSDK_INT32 IPageSize;                              // Number of alerts per page
    DPSDK_INT32 IPageNo;                                // Query page number (from 1Start)
    DPSDK_INT32   ISortType;   //   Sort   fields   (1=Alarm   time,2=Alarm   type,3=Alarm
level,4=report Police officer,5=Processing state)
    DPSDK_INT32  ISortOrder;                                          // Sort direction (0=Ascending
order,1=Descending order)
    DPSDK_INT32* PAlarmType;                                   // Alarm type (Reference resources
Alarm_type_e)
    DPSDK_UINT32 UiAlarmTypeNumber;                  // Number of alarm types
    DPSDK_INT32* PAlarmGrade;                                   // Alarm level (Reference resources
AlarmLevel_e)
    DPSDK_UINT32 UiAlarmGradeNumber;                  // The number of alarm levels
    DPSDK_INT32* PAlarmStatus;                                   // Alarm state (Reference resources
AlarmState_e)
    DPSDK_UINT32 UiAlarmStatusNumber;                 // The number of alarm states
    DPSDK_INT32* PHandleStatus;                                // Alarm processing state (Reference
resources AlarmDealWith_e)
    DPSDK_UINT32 UiHandleStatusNumber;                       // The number of state of the alarm
processing
}DPSDK_QUERYALARM_PARAM;
```

Father theme:structural morphology

DPSDK_ALARM_DETAILINFO_LIST

Alarm record list

```
Typedef Struct
{
    DPSDK_UINT32 UiTotal;                    // Total number of alarm records
    DPSDK_ALARM_DETAILINFO StruAlarmInfoList[1]; // Alarm record
}DPSDK_ALARM_DETAILINFO_LIST;
```

Father theme:structural morphology

DPSDK_QUERYALARMCOUNT_PARAM

Alarm total query parameters

```
Typedef Struct
{
    DPSDK_CHAR SzBeginTime[DPSDK_ALARM_TIME_LEN];              // Start time of alarm
Yyyy mm dd hh mmss
    DPSDK_CHAR SzEndTime[DPSDK_ALARM_TIME_LEN];                 // The end time of the
alarm Yyyy mm dd hh mm ss
    DPSDK_CHAR  SzHandleBeginTime[DPSDK_ALARM_TIME_LEN];                    // Alarm
processing start time Yyyy mm dd hh mm ss
    DPSDK_CHAR  SzHandleEndTime[DPSDK_ALARM_TIME_LEN];                      // Alarm
processing end time Yyyy mm dd hh mmss
    DPSDK_CHAR SzDeviceId[DPSDK_ALARM_DEVICEID_LEN];            // dvice ID
    DPSDK_CHAR SzChannelId[DPSDK_ALARM_CHANNELID_LEN];          // channel ID
    DPSDK_CHAR SzOrgId[DPSDK_ALARM_ORGID_LEN];                  // Organization node
ID
    DPSDK_CHAR SzAlarmId[DPSDK_ALARM_ALARMID_LEN];               // Call the police
ID
    DPSDK_CHAR SzAlarmCode[DPSDK_ALARM_ALARMCODE_LEN];          // Alarm code
(specifying this condition to ignore other conditions)
    DPSDK_CHAR  SzHandleUser[DPSDK_ALARM_HANDLERUSER_LEN];             // Alarm
processing person
    DPSDK_INT32* PAlarmType;                                // Alarm type (Reference resources
Alarm_type_e)
    DPSDK_UINT32 UiAlarmTypeNumber;                         // Number of alarm types
    DPSDK_INT32* PAlarmGrade;                               // Alarm level (Reference resources
AlarmLevel_e)
    DPSDK_UINT32 UiAlarmGradeNumber;                         // The number of alarm levels
    DPSDK_INT32* PAlarmStatus;                              // Alarm state (Reference resources
AlarmState_e)
    DPSDK_UINT32 UiAlarmStatusNumber;                       // The number of alarm states
    DPSDK_INT32* PHandleStatus;                            // Alarm processing state (Reference
resources AlarmDealWith_e)
    DPSDK_UINT32 UiHandleStatusNumber;                       // The number of state of the alarm
processing
}DPSDK_QUERYALARMCOUNT_PARAM;
```

Father theme:structural morphology

DPSDK_ALARMPROCESS_DETAILINFO_LIST

Alarm processing record list

```
Typedef Struct
{
    DPSDK_UINT32 UiTotal;                                   // Total number of alarm  processing
information
    DPSDK_ALARMPROCESS_DETAILINFO   StruAlarmProcessInfoList[1];           // Alarm
processing information
}DPSDK_ALARMPROCESS_DETAILINFO_LIST;
```

Father theme:structural morphology

DPSDK_BLOCKALARM_PARAM

Shielded alarm parameter

```
Typedef Struct
{
    DPSDK_CHAR      SzAlarmCodeSource[DPSDK_ALARM_ALARMSOURCE_LEN];      // Shielded alarm source (device alarm for device code, channel alarm as channel Code, system alarm for service code
    DPSDK_INT32 IAlarmType; // Shielding alarm type (Reference resources Alarm_type_e)
    DPSDK_INT32 IDuration; // The length of the shielding time (unit: Second)
}DPSDK_BLOCKALARM_PARAM;
```

Father theme:structural morphology

DPSDK_ALARMEXPORT_PARAM

Alarm export parameters

```
Typedef Struct
{
    DPSDK_INT32 ISortType; // Sort fields (1=Alarm time,2=Alarm type,3=Alarm level,4=AlarmTake care of people,5=Processing state)
    DPSDK_INT32 ISortOrder; // Sort direction (0=Ascending order,1=Descending order)
    DPSDK_CHAR SzAlarmId[DPSDK_ALARM_ALARMID_LEN];          // Call the police ID
    DPSDK_CHAR SzAlarmCode[DPSDK_ALARM_ALARMCODE_LEN];      // Alarm code
    DPSDK_CHAR SzOrgId[DPSDK_ALARM_ORGID_LEN];              // Organization node ID
    DPSDK_CHAR SzDeviceId[DPSDK_ALARM_DEVICEID_LEN];        // equipment ID
    DPSDK_CHAR SzChannelId[DPSDK_ALARM_CHANNELID_LEN];      // channel ID
    DPSDK_CHAR SzBeginTime[DPSDK_ALARM_TIME_LEN];           // Start time of alarm Yyyymmddhhmmss
    DPSDK_CHAR SzEndTime[DPSDK_ALARM_TIME_LEN];             // The end time of the alarm Yyyymmddhhmmss
    DPSDK_CHAR SzHandleBeginTime[DPSDK_ALARM_TIME_LEN];     // Alarm processing start time Yyyymmddhhmmss
    DPSDK_CHAR SzHandleEndTime[DPSDK_ALARM_TIME_LEN];       // Alarm processing end time Yyyymmddhhmmss
    DPSDK_CHAR SzHandleUser[DPSDK_ALARM_HANDLERUSER_LEN];   // Alarm processing person
    DPSDK_CHAR SzLanguage[DPSDK_ALARM_LANGUAGE_LEN];        // language
    DPSDK_INT32* PAlarmType;                                // Alarm type(Reference resources Alarm_type_e)
    DPSDK_UINT32 UiAlarmTypeNumber;                         // Number of alarm types
    DPSDK_INT32* PAlarmGrade;                               // Alarm level(Reference resources AlarmLevel_e)
    DPSDK_UINT32 UiAlarmGradeNumber;                        // The number of alarm levels
    DPSDK_INT32* PAlarmStatus;                              // Alarm state(Reference resources AlarmState_e)
    DPSDK_UINT32 UiAlarmStatusNumber;                       // The number of alarm states
    DPSDK_INT32* PHandleStatus;                             // Alarm processing state(Reference resources AlarmDealWith_e)
    DPSDK_UINT32 UiHandleStatusNumber;                      // The number of state of the alarm processing
}DPSDK_ALARMEXPORT_PARAM;
```

Father theme:structural morphology

DPSDK_PTZOPERATE_FUNCTION_PARAM

Operating parameters of cloud platform

```
Typedef Struct
{
    Typedef Enum
    {
        PtzOF_Show_PtzMenu = 0,              // display "Cloud platform menu"
        PtzOF_Move_PtzMenu = 1,              // control "Menu direction of the cloud platform"
        PtzOF_Confirm_PtzMenuItem = 2,          // Determine "Cloud platform menu item"
        PtzOF_Set_LineScannBorder = 3,          // Set up "Line scavenging boundary"
        PtzOF_Switch_LineScanBorder = 4,        // switch "Line scan"
        PtzOF_Switch_AutoRotate = 5,            // switch "Horizontal rotation"
        PtzOF_Switch_Light = 6,                // switch "lighting"
        PtzOF_Switch_RainBrush = 7,             // switch "Wiper"
        PtzOF_Switch_InfraredLight = 8,         // switch "infrared light"
        PtzOF_Switch_AssisentPoint = 9,         // switch "Auxiliary point"
        PtzOF_Switch_Cruise = 10,              // switch "Cruise function"
        PtzOF_Switch_Track = 11,               // switch "Cruising"
        PtzOF_Switch_SetTrack = 12,             // switch "Track setting"
    }PtzOperateFunction_e;
        DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; // channel ID
        PtzOperateFunction_e IPtzOFType;              // Operating function type of cloud platform
        DPSDK_INT32  ICruiseId;     // (This variable is only in Switch_CruiseOperation when
effective) Cruise ID
        DPSDK_INT32     ITrackId;                      // (This variable is only in Switch_Track
、Switch_SetTrackEffective operation during operation ID
        DPSDK_INT32 ISwitchMode; // (This variable is only in SwitchEffective operation)0-
Close,  1-open
        DPSDK_INT32 IBorderType; // (This variable is only in Set_LineScannBorderEffective
operation)16-Left boundary17-.Right boundary
         DPSDK_INT32 IAssisentType; // (This variable is only in Switch_AssisentPointEffective
operation)23-Backlight compensation,24-Number doubled,27- Color turn black,35-Shutter time,41-
Brightness,42-Image flip,43-The name of the preset point is hidden,80-Restore factory settings
        DPSDK_INT32  IMoveType;  //  (This  variable  is  only  in  Move_PtzMenuEffective
operation)25-Upward movement,26-Move down,27-Left shift,28-Right
        DPSDK_INT32 ISwitchPtzMenu; // (This variable is only in Show_PtzMenuEffective
operation)22=Open the platform menu,23=Close the cloud table menu
}DPSDK_PTZOPERATE_FUNCTION_PARAM;
```

Father theme:structural morphology

DPSDK_PTZOPERATE_RESULT

Function operation result of cloud platform

```
Typedef Struct
{
    DPSDK_PTZ_LOCKUSER StruLockUser;
    DPSDK_INT32 IResult; // Operation results:0-failure,1-success
}DPSDK_PTZOPERATE_RESULT;
```

Father theme:structural morphology

DPSDK_PTZOPERATE_CAMERA_PARAM

Operating cloud platform camera parameters

```
Typedef Struct
{
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; // channel ID
    DPSDK_INT32 IDirect;                          // Direction:1-increase,2-decrease
    DPSDK_INT32 ICommand;                         // Order:0-stop it，  1-open
    DPSDK_INT32 IStep;                            // step
    DPSDK_INT32 IOperateType;                     // Operation type:1-variable,2-zoom,3-aperture
    DPSDK_CHAR SzExtend[DPSDK_PTZ_EXTEND_LEN];    // Extended data
}DPSDK_PTZOPERATE_CAMERA_PARAM;
```

Father theme:structural morphology

DPSDK_PTZOPERATE_DIRECT_PARAM

Cloud platform direction control parameters

```
Typedef Struct
{
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; // channel ID
    DPSDK_INT32 IStepY;                           // Vertical direction step
    DPSDK_INT32 IStepX;                           // Horizontal direction step
    DPSDK_INT32  IDirect;           // Direction:1-On,2-Under the,3-Left,4-Right,5-Upper left,6-
Lower left,7-On the right,8-lower right
    DPSDK_INT32 ICommand;                         // Order:0-Stop it, 1-open
    DPSDK_CHAR SzExtend[DPSDK_PTZ_EXTEND_LEN];    // Extended data
}DPSDK_PTZOPERATE_DIRECT_PARAM;
```

Father theme:structural morphology

DPSDK_PTZOPERATE_FOCUS_PARAM

Electric focusing control parameters

```
Typedef Struct
{
     DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; // channel ID
     DPSDK_FLOAT Focus;                            // focal length
     DPSDK_FLOAT FZoom;                             // Multiple
     DPSDK_INT32  IOperateType;                                // Operation type:0-Reset,1-Continuous
focusing,2-Autofocus
}DPSDK_PTZOPERATE_FOCUS_PARAM;
```

Father theme:structural morphology

DPSDK_PTZOPERATE_PRESETPOINT_PARAM

Control preset point parameters

```
Typedef Struct
{
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN];       // channel ID
    DPSDK_CHAR SzPointCode[DPSDK_PRESETPOINT_CODE_LEN]; // Preset point coding
    DPSDK_CHAR SzPointName[DPSDK_PRESETPOINT_NAME_LEN]; // Preset point name
    DPSDK_INT32 IOperateType;                           // Operation type:1-Location, 2-Set up, 3-
delete, 4-Update working time
    DPSDK_CHAR SzStartTime[DPSDK_PTZ_TIME_LEN];         // start time(time stamp)
    DPSDK_CHAR SzEndTime[DPSDK_PTZ_TIME_LEN];           // end time(time stamp)
}DPSDK_PTZOPERATE_PRESETPOINT_PARAM;
```

Father theme:structural morphology

DPSDK_PTZOPERATE_SITPOSITION_PARAM

Three dimensional positioning parameters

```
Typedef Struct
{
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; // channel ID
    DPSDK_INT32 IPointX; // Horizontal coordinates:-8192 ~ ~ Eight thousand one hundred and
ninety-two
    DPSDK_INT32 IPointY; // Vertical coordinates:-8192 ~ ~ Eight thousand one hundred and
ninety-two
    DPSDK_INT32 IPointZ; // Variable number:-4 ~ ~ 4
    DPSDK_CHAR SzExtend[DPSDK_PTZ_EXTEND_LEN]; // Extended data
}DPSDK_PTZOPERATE_SITPOSITION_PARAM;
```

Father theme:structural morphology

DPSDK_PTZOPERATE_ARRANGEPTZ_PARAM

Lock the unlocking parameters

```
Typedef Struct
{
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; // channel ID
    DPSDK_UINT32 UiLockTime; // Lock time, unit second,texpression has been locked until the release or Bei Qiang Wins.
    DPSDK_INT32 IOperateType; // Operation type:0-Unknown, 1-Lock the current camera, 2-Unlock the current camera, 3-Unlock all the cameras locked by the user, 4-Lock all the cameras, 5-Query lock state
    DPSDK_CHAR SzExtend[DPSDK_PTZ_EXTEND_LEN];   // Extended data
}DPSDK_PTZOPERATE_ARRANGEPTZ_PARAM;
```

Father theme:structural morphology

DPSDK_PTZOPERATE_ALARMOUT_PARAM

Alarm output control parameters

```
Typedef Struct
{
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; // channel  ID
    DPSDK_INT32 IOperateType;                    // Control type:1-Status control,2-Pattern control
    DPSDK_INT32 ICommand; // Control commands: state control,1-Open,0-Shut down; mode
control:0-Close,  1-Automatically,2-Manual
}DPSDK_PTZOPERATE_ALARMOUT_PARAM;
```

Father theme:structural morphology

DPSDK_PTZ_PRESETPOINT_LIST

Get a list of preset points

```
Typedef Struct
{
    DPSDK_UINT32 UiTotal;                          // Total
    DPSDK_PTZ_PRESETPOINT_INFO StruPresetPointInfo[1]; // Preset point list
}DPSDK_PTZ_PRESETPOINT_LIST;
```

Father theme:structural morphology

DPSDK_TVWALL_LIST

TV wall list

```
Typedef Struct
{
     DPSDK_UINT32 UiTotal;                    // The number of information of the TV wall
     DPSDK_TVWALL_BASE_INFO StruTVWallBaseInfo[1]; // TV wall information list
}DPSDK_TVWALL_LIST;
```

Father theme:structural morphology

DPSDK_TVWALL_INFO

Television wall information

```
Typedef Struct
{
    DPSDK_TVWALL_BASE_INFO StruBaseInfo;              // The basic information of the TV wall
    DPSDK_SCREEN_DECODER_LIST StruScreenDecoderList; // A list of decoded channels for screen binding
}DPSDK_TVWALL_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_TASK_LIST

TV wall task list

```
Typedef Struct
{
    DPSDK_UINT32 UiTotal;                    // TV wall task number
    DPSDK_TVWALL_TASK_BASE_INFO StruTVWallTask[1]; // TV wall task list
}DPSDK_TVWALL_TASK_LIST;
```

Father theme:structural morphology

DPSDK_TVWALL_TASK_INFO

TV Wall Task Info

```
typedef struct
{
    DPSDK_TVWALL_TASK_BASE_INFO struBaseInfo;                    // TV Wall Task Info
    DPSDK_TVWALL_TASK_SCREEN_OPER_LIST struScreenOperList;         // Operating Info List of TV Wall Task Screen
     DPSDK_TVWALL_TASK_CHANNEL_EXT_LIST  struChannelExtList;              // Decoding device refers to device info list needed by video source under direct decoding mode
}DPSDK_TVWALL_TASK_INFO;
```

Father theme:structural morphology

DPSDK_CURRENT_TVWALL_TASK_LIST

The current execution of the TV wall task list

```
Typedef Struct
{
    DPSDK_UINT32 UiTotal;                    // TV wall task number
    DPSDK_CURRENT_TVWALL_TASK_INFO StruTVWallTask[1]; // TV wall task list
}DPSDK_CURRENT_TVWALL_TASK_LIST;
```

Father theme:structural morphology

DPSDK_TVWALL_OPEN_WINDOW

Window

```
Typedef Struct
{
    DPSDK_INT32 ITvWallId; // TV wall ID
    DPSDK_INT32 IScreenId; // screen ID
    DPSDK_INT32  ITvIndex;    // Physical channel number, cubeless screen can be used with screen IDidentical
    DPSDK_CHAR  SzMatrixId[DPSDK_DEVICE_ID_LEN];        // Decoder, matrix and other decoding devices ID
    DPSDK_INT32 ITvType;                          // 0 Non cubeless screen, 1 Cubeless screen
    DPSDK_TVWALL_WINDOW_INFO StruWndInfo[1];      // Window list
}DPSDK_TVWALL_OPEN_WINDOW;
```

Father theme:structural morphology

DPSDK_TVWALL_CONTROL_INFO

Upper wall control operation

```
Typedef Struct
{
    DPSDK_INT32   IControlType;                                        //   control   command   See
DPSDK_TVWALL_CONTROL_TYPEDefinition
    DPSDK_CHAR   SzMatrixId[DPSDK_DEVICE_ID_LEN];     // Decoder, matrix and other
decoding devices ID
    DPSDK_INT32 ITvIndex; //The screen number, the ordinary screen is the decode channel
number, the fusion screen is the screen ID
    DPSDK_INT32 ISubTvIndex;                    // Subscreen number (fusion screen is valid)
    DPSDK_INT32 ITvType;                        // Screen type:0-Non fusion screen,1-Fusion screen
    DPSDK_INT32 ISplitNum;                      // Screen partition or window number
    DPSDK_TVWALL_SCREEN_POS StruScreenPos;       // Screen position
    DPSDK_INT32 ITvWallDBId;                    // TV wall configuration scheme DBId
    DPSDK_INT32 ITvWallVersion;                 // TV wall version number
    DPSDK_INT32 IScreenId;                      // Screen ID for instant mode operation on single
screen, task mode is invalid
    DPSDK_INT32 IZoder;                         // ZOrder,-1 Bottom,0Top
    DPSDK_INT32 ISubWindNo;                     //Split screen number selected in window window
}DPSDK_TVWALL_CONTROL_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_TASK_INFO_LIST

TV Wall Task Info

```
typedef struct
{
    DPSDK_INT32 iTVWallTaskNum;                          // Number of TV Wall Task
    DPSDK_TVWALL_TASK_INFO* pTVWallTaskList;             // List of TV Wall Task
    DPSDK_TVWALL_PROJECT_LIST struProjectList;           // List of TV Wall Plan
}DPSDK_TVWALL_TASK_INFO_LIST;
```

Father theme:structural morphology

AlarmLevel_e

Alarm level

```
Typedef Enum
{
    ALARM_LEVEL_ZERO = 0,
    ALARM_LEVEL_ONE = 1,
    ALARM_LEVEL_TWO = 2,
    ALARM_LEVEL_TFREE = 3,
    ALARM_LEVEL_FOUR = 4,
    ALARM_LEVEL_FIVE = 5,
}AlarmLevel_e;
```

Father theme:structural morphology

AlarmState_e

Alarm type

```
Typedef Enum
{
    ALARM_EVENT_OCCUR = 1,     // Alarm production
    ALARM_EVENT_DISAPPEAR,     // Disappearance of alarm
    ALARM_ENENT_PULSE,         // Pulse state
}AlarmState_e;
```

Father theme:structural morphology

AlarmObject_e

Alarm object type

```
Typedef Enum
{
    ALARM_EVENT_DEVICE = 1, // Equipment alarm
    ALARM_EVENT_CHANNEL,     // Channel alarm
    ALARM_ENENT_SYSTEM,      // System alarm
}AlarmObject_e;
```

Father theme:structural morphology

Alarm_type_e

Alarm type

```
typedef enum
{
    ALARM_TYPE_Unknown                    = 0, // Unknown(when search alarm, use this to search
all alarms, as not filter alarm type)
    ALARM_TYPE_VIDEO_LOST                 = 1,    // video loss
    ALARM_TYPE_EXTERNAL_ALARM                    = 2,    // external alarm
    ALARM_TYPE_MOTION_DETECT                     = 3,    // motion detect
    ALARM_TYPE_VIDEO_SHELTER                     = 4,    // video tamper
    ALARM_TYPE_DISK_FULL                         = 5, // disk full
   ALARM_TYPE_DISK_FAULT                         = 6,    // disk failure
   ALARM_TYPE_FIBER                          = 7,    // fiber alarm
   ALARM_TYPE_GPS                            = 8,    // GPS info
   ALARM_TYPE_3G                            = 9,    // 3G
   //device patrol
   ALARM_TYPE_STATUS_RECORD                      = 10,   // device record status
   ALARM_TYPE_STATUS_DEVNAME                     = 11,   // device name
   ALARM_TYPE_STATUS_DISKINFO                    = 12,   // disk info
   ALARM_TYPE_IPC_OFF                        = 13,   // front IPC offline
   ALARM_TYPE_DEV_DISCONNECT                     = 16,   //device offline alarm
   //Jingde Huarun Gas Project
   ALARM_POWER_INTERRUPT                         = 17,   // City grid outage alarm
   ALARM_POWER_ENABLED                           = 18,   // city grid ON alarm
   ALARM_INFRARED_DETECT                         = 19,   // IR detection alarm
   ALARM_GAS_OVER_SECTION                        = 20,   // gas concentration over limit
   ALARM_FLOW_OVER_SECTION                       = 21,    // instant flow over limit
   ALARM_TEMPERATURE_OVER_SECTION                = 22,    // temperature over limit
   ALARM_TEMPERATURE_UNDER_SECTION               = 23,    // temperature below limit
   ALARM_PRESSURE_OVER_SECTION                   = 24,    // pressure over limit
   ALARM_PRESSURE_UNDER_SECTION                  = 25,    // pressure below limit
   ALARM_STATIC_DETECTION                        = 26,    // status detection
   ALARM_REMOTE_EXTERNAL_ALARM                   = 28,    // remote external alarm
   ALARM_BUTTON_EXTERNAL_ALARM                   = 29,    // alarm button external alarm
    ALARM_POWER_INTERRUPT_EXTERNAL_ALARM             =  30,       // outage signal
external alarm
  ALARM_RECORD_LOSS                          = 31,    // record loss event, caused by error and
 etc. when disk is normal
   ALARM_VIDEO_FRAME_LOSS                        = 32,    // video loss event, such as caused by
poor or poor encode capacity
   ALARM_RECORD_VOLUME_FAILURE                   = 33,    // error caused by disk volume,
so record is abnormal
```

```
    //门禁
    ALARM_DOOR_BEGIN                              = 40,     // A&C device alarm start
    ALARM_FORCE_CARD_OPENDOOR                     = 41,     // duress card unlock
    ALARM_VALID_PASSWORD_OPENDOOR                 = 42,     // valid password unlock
    ALARM_INVALID_PASSWORD_OPENDOOR               = 43,     // invalid password unlock
    ALARM_FORCE_PASSWORD_OPENDOOR                 = 44,     // duress password unlock
    ALARM_VALID_FINGERPRINT_OPENDOOR              = 45,     // valid fingerprint unlock
    ALARM_INVALID_FINGERPRINT_OPENDOOR            = 46,     // invalid fingerprint unlock
    ALARM_FORCE_FINGERPRINT_OPENDOOR              = 47,     // duress fingerprint unlock
    ALARM_REMOTE_METHOD_OPENDOOR                  = 48,     // remote unlock: VTH
unlock/platform remote unlock
    ALARM_BUTTON_METHOD_OPENDOOR                  = 49,     // key unlock
    ALARM_LOCKKEY_METHOD_OPENDOOR                 = 50,     // key unlock
    ALARM_TYPE_VALID_CARD_READ                    = 51,     // valid card/unlock
    ALARM_TYPE_INVALID_CARD_READ                  = 52,     // invalid card/unlock
    ALARM_TYPE_DOOR_MAGNETIC_ERROR                = 53,     // door sensor
    ALARM_TYPE_DOOR_BREAK                         = 54,     // unlock error
    ALARM_TYPE_DOOR_ABNORMAL_CLOSED               = 55,     // lock error
    ALARM_TYPE_DOOR_NORMAL_CLOSED                 = 56,     // normal lock
    ALARM_TYPE_DOOR_OPEN                          = 57,     // normal unlock
    ALARM_TALK_REQUEST                            = 59,     //A&C talk request alarm
    ALARM_DOOR_OPEN_TIME_OUT_BEG                  = 60,
    ALARM_DOOR_OPEN_TIME_OUT_END                  = 70,
    // alarm 主机
    ALARM_TYPE_ALARMHOST_BEGIN                    = 80,
    ALARM_TYPE_ALARM_CONTROL_ALERT                = 81,     // alarm controller alarm
    ALARM_TYPE_FIRE_ALARM                         = 82,     // fire alarm
    ALARM_TYPE_ZONE_DISABLED                      = 83,     // invalid zone
    ALARM_TYPE_BATTERY_EMPTY                      = 84,     // no battery-device alarm
    ALARM_TYPE_AC_OFF                             = 85,     // city grid outage-device alarm
    ALARM_TYPE_ALARMHOST_END                      = 90,
    ALARM_FILESYSTEM                              = 100,    // file system
    ALARM_RAID_FAULT                              = 101,    // raid failure
     ALARM_RECORDCHANNELFUNCTION_ABNORMAL         = 102,    // record channel
function error
    SVR_HARDDISK_STATUS                           = 103,    // HDD status
    ALARM_RECORD_REPAIR                           = 104,    // record repair -P3.0
    //begingrid  alarm 类型
    ELECTRIC_WIRE_SHOCK                           = 109,    // grid shock
    ELECTRIC_WIRE_INTERRUPT                       = 110,    // grid outage
    ELECTRIC_WIRE_SHORT_CIRCUIT                   = 111,    // grid short circuit
    ELECTRIC_WIRE_BREAKDOWN                       = 112,    // grid failure
    ELECTRIC_WIRE_VOLTAGE_LOW                     = 113,    // grid low voltage
```

```
   //end
   ALARM_TYPE_RECORD_WRITE_FAIL              = 114,      // failed to write in record
   //grid  new  alarm  type  begin  add  by  hu_wenjuan
   ELECTRIC_ALARM_BEGIN_EX                 = 115,
   ELECTRIC_BREAK_CIRCUIT                 =  115,  //  grid  open  circuit
   ELECTRIC_SENSE_ALARM                   =  116,  //  grid  sensor  alarm
   ELECTRIC_ALARM_END_EX                  = 150,
   //grid new alarm type end
   ALARM_VTT_URGENCY                      = 160,       // VTT device emergency button alarm
   //-M related alarm is added here
   ALARM_MOTOR_BEGIN                      = 200,
   ALARM_OVERSPEED_OCCURE                    = 201,               // over speed alarm occur
    ALARM_OVERSPEED_DISAPPEAR                   = 202,                     // Over speed  alarm
cancel
   ALARM_DRIVEROUT_DRIVERALLOW              = 203,                // exit zone
   ALARM_DRIVERIN_DRIVERALLOW              = 204,                // enter zone
   ALARM_DRIVEROUT_FORBIDDRIVE             = 205,                // exit forbidden zone
   ALARM_DRIVERIN_FORBIDDRIVE              = 206,                // enter forbidden zone
   ALARM_DRIVEROUT_LOADGOODS               = 207,                // exit loading zone
   ALARM_DRIVERIN_LOADGOODS                = 208,                 // enter loading zone
   ALARM_DRIVEROUT_UNLOADGOODS              = 209,                // exit unloading zone
   ALARM_DRIVERIN_UNLOADGOODS               = 210,                // enter unloading zone
   ALARM_CAR_OVER_LOAD                     = 211,                // over load
   ALARM_SPEED_SOON_ZERO                   = 212,                //  brake
   ALARM_3GFLOW                         = 213,                // 3G flow
   ALARM_AAC_POWEROFF                     = 214,                // ACC outage alarm
    ALARM_SPEEDLIMIT_LOWERSPEED                 = 215,                     // speed limit  alarm
LowerSpeed
    ALARM_SPEEDLIMIT_UPPERSPEED                 = 216,                     // speed limit alarm
UpperSpeed
   ALARM_VEHICLEINFOUPLOAD_CHECKIN           = 217,                 // mobile custom info
uplopad CheckIn
    ALARM_VEHICLEINFOUPLOAD_CHECKOUT           = 218,                     // mobile custom
info uplopad CheckOut
   ALARM_CAR_OPEN_DOOR                     = 219,                // unlock alarm
   ALARM_URGENCY                         = 220,                // emergency alarm
   ALARM_VEHICLE_LARGE_ANGLE                  = 224,                    // mobile camera large
angle twist event
   ALARM_BATTERYLOWPOWER                    = 225,                // low battery alarm
   ALARM_TEMPERATURE                      = 226,              // high temperature alarm
   ALARM_DEV_VOICE_EX                     = 229,              // device audio request alarm
   ALARM_POWER_OFF_EX                     = 230,              // outage alarm
   ALARM_ROUTE_OFFSET_EX                   = 231,              // route shift alarm
```

```
    ALARM_TYRE_PRESSURE_EX                    = 232,                    // tire pressure detection
alarm
   ALARM_FATIGUE_DRIVING                    = 233,                    // fatigue driving alarm
   ALARM_DRIVER_CHECKIN                     = 234,                    // driver sign in
   ALARM_DRIVER_CHECHOUT                    = 235,                    // driver sign out
   ALARM_GAS_LOWLEVEL                       = 236,                    // low level alarm
   ALARM_GAS_INFO                           = 237,                    // low level info
   ALARM_GETIN_STATION                      = 238,                    // enter alarm
   ALARM_GETOUT_STATION                     = 239,                    // exit alarm
   ALARM_STATION_BEGIN_IN                   = 240,                    // departure station enter alarm
   ALARM_STATION_BEGIN_OUT                  = 241,                    // departure station exit alarm
   ALARM_STATION_END_IN                     = 242,                    // destination enter alarm
   ALARM_STATION_END_OUT                    = 243,                    // destination exit alarm
  <in/out station type alarm are placed together>
   ALARM_STAY_STATION_OVERTIME              = 244,                    // parking timeout alarm
   ALARM_RECOVER_RUNNING                    = 245,                    // running recover alarm
   ALARM_MEAL                               = 246,                    // meal alarm
   ALARM_BLOCK                              = 247,                    // block alarm
   ALARM_CALL                               = 248,                    // call alarm
   ALARM_CAR_BREAKDOWN                      = 249,                    // vehicle breakdown alarm
   ALARM_STOP_RUNNING                       = 250,                    // stop running alarm
   ALARM_ROBING                             = 251,                    // robbery alarm
   ALARM_DISPUTE                            = 252,                    // dispute alarm
   ALARM_ACCIDENT                           = 253,                    // event alarm
   ALARM_OVER_SPEED                         = 254,                    // over speed   alarm
   ALARM_RENTAL                             = 255,                    // rental alarm
   ALARM_MAINTENANCE                        = 256,                    // maintenance alarm
   ALARM_CLOSURE                            = 257,                    // closure alarm
   ALARM_OPEN_OR_CLOSE_DOOR                 = 258,                    // open/close door alarm
   ALARM_ILLEGALIN_OVERSPEED                = 259,                    // invalid speed limit alarm
   ALARM_ILLEGALOUT_OVERSPEED               = 260,                    // invalid exit speed limit
alarm
    ALARM_ILLEGALIN_DRIVERALLOW             = 261,                    // invalid enter drive
zone alarm
    ALARM_ILLEGALOUT_DRIVERALLOW            = 262,                    // invalid exit drive
zone alarm
   ALARM_ILLEGALIN_FORBIDDRIVE              = 263,                    // invalid enter forbidden
zone alarm
   ALARM_ILLEGALOUT_FORBIDDRIVE             = 264,                    // invalid exit forbidden
zone alarm
    ALARM_ILLEGALIN_LOADGOODS               = 265,                    // invalid enter loading
zone alarm
    ALARM_ILLEGALOUT_LOADGOODS              = 266,                    // invalid exit loading
```

```
zone alarm
    ALARM_ILLEGALIN_UNLOADGOODS                = 267,                    // invalid enter
unloading zone alarm
    ALARM_ILLEGALOUT_UNLOADGOODS               = 268,                    // invalid exit
unloading zone alarm
   ALARM_ILLEGALIN_GETIN_STATION           = 269,                // invalid enter alarm
   ALARM_ILLEGALIN_GETOUT_STATION          = 270,                // invalid exit alarm
   ALARM_DETAINED                       = 272,              // vehicle detained alarm
    ALARM_DELAY                          = 273,                // trusteeship alarm， vehicle shift
delayed
   ALARM_MOTOR_END                       = 300,
   //智能 alarm
ALARM_IVS_ALARM_BEGIN                           = 300,                   // IVS device alarm on
 dhnetsdk.h base, add in type +300 server（DMS）
   ALARM_IVS_ALARM                        = 0x00000001 + 300,    // IVS device alarm
   ALARM_CROSSLINEDETECTION               = 0x00000002 + 300,    // warning line event
   ALARM_CROSSREGIONDETECTION          = 0x00000003 + 300,   // warning zone
event event
   ALARM_PASTEDETECTION                   = 0x00000004 + 300,    // stripe event
   ALARM_LEFTDETECTION                    = 0x00000005 + 300,    // abandoned object event
   ALARM_STAYDETECTION                    = 0x00000006 + 300,    // stay event
   ALARM_WANDERDETECTION                  = 0x00000007 + 300,    // wandering event
   ALARM_PRESERVATION                     = 0x00000008 + 300,    // object protection event
   ALARM_MOVEDETECTION                    = 0x00000009 + 300,    // move event
   ALARM_TAILDETECTION                    = 0x0000000A + 300,    // tail event
   ALARM_RIOTERDETECTION                  = 0x0000000B + 300,    // crowd event
   ALARM_FIREDETECTION                    = 0x0000000C + 300,    // fire event
   ALARM_SMOKEDETECTION                   = 0x0000000D + 300,    // smoke alarm event
   ALARM_FIGHTDETECTION                   = 0x0000000E + 300,   // fight event
   ALARM_FLOWSTAT                         = 0x0000000F + 300,    // flow statistics event
   ALARM_NUMBERSTAT                       = 0x00000010 + 300,    // quantity statistics event
   ALARM_CAMERACOVERDDETECTION              = 0x00000011 + 300,    // camera cover
event
   ALARM_CAMERAMOVEDDETECTION               = 0x00000012 + 300,    // camera move
event
    ALARM_VIDEOABNORMALDETECTION                  =  0x00000013 + 300,      // video
abnormal event
   ALARM_VIDEOBADDETECTION                = 0x00000014 + 300,    // video loss event
   ALARM_TRAFFICCONTROL                   = 0x00000015 + 300,   // traffic control event
   ALARM_TRAFFICACCIDENT                  = 0x00000016 + 300,   // traffic event
   ALARM_TRAFFICJUNCTION                  = 0x00000017 + 300,   // intersection event
   ALARM_TRAFFICGATE                      = 0x00000018 + 300,   // ANPR event
   ALARM_TRAFFICSNAPSHOT                  = 0x00000019 + 300,   // traffic snapshot event
```

```
    ALARM_FACEDETECT                      = 0x0000001A + 300, // face detection event—
normal face detection.mark: abnormal face detection（901）define below, type difference at
DMS,2013.12.10,18842)
   ALARM_TRAFFICJAM                  = 0x0000001B + 300,   // traffic jam event
   ALARM_TRAFFIC_RUNREDLIGHT              = 0x00000100 + 300,   // traffic violation-run
the red light event
   ALARM_TRAFFIC_OVERLINE                 = 0x00000101 + 300, // traffic violation-cross
lane line event
    ALARM_TRAFFIC_RETROGRADE                = 0x00000102 + 300, // traffic violation-
retrogradation event
    ALARM_TRAFFIC_TURNLEFT                   = 0x00000103 + 300, // traffic violation-left
turn
   ALARM_TRAFFIC_TURNRIGHT                 = 0x00000104 + 300, // traffic violation-right
turn
   ALARM_TRAFFIC_UTURN                  = 0x00000105 + 300,   // traffic violation-u turn
   ALARM_TRAFFIC_OVERSPEED                 = 0x00000106 + 300,   // traffic violation-over
speed
   ALARM_TRAFFIC_UNDERSPEED                = 0x00000107 + 300, // traffic violation-low
speed
   ALARM_TRAFFIC_PARKING                   = 0x00000108 + 300, // traffic violation-illegal
parking
    ALARM_TRAFFIC_WRONGROUTE                = 0x00000109 + 300, // traffic violation-
wrong lane drive
    ALARM_TRAFFIC_CROSSLANE                  = 0x0000010A + 300, // traffic violation-
illegal lane change
   ALARM_TRAFFIC_OVERYELLOWLINE            = 0x0000010B + 300, // traffic violation-
cross yellow line
     ALARM_TRAFFIC_DRIVINGONSHOULDER              =  0x0000010C + 300,    // traffic
violation-road shoulder drice event
     ALARM_TRAFFIC_YELLOWPLATEINLANE              =  0x0000010E + 300,    // traffic
violation-yellow-plate vehicle in lane event
   ALARM_CROSSFENCEDETECTION             = 0x0000011F + 300,   // cross fence event
    ALARM_ELECTROSPARKDETECTION               = 0X00000110 + 300,   // electric spark
event
    ALARM_TRAFFIC_NOPASSING                 = 0x00000111 + 300, // traffic violation-no
entry event
    ALARM_ABNORMALRUNDETECTION                = 0x00000112 + 300,   // abnormal run
event
      ALARM_RETROGRADEDETECTION                    = 0x00000113 + 300,    // user
retrogradation event
   ALARM_INREGIONDETECTION              = 0x00000114 + 300,   // in-zone check event
    ALARM_TAKENAWAYDETECTION                  = 0x00000115 + 300,   // missing object
event
```

```
   ALARM_PARKINGDETECTION                = 0x00000116 + 300,   // invalid parking event
   ALARM_FACERECOGNITION                 = 0x00000117 + 300,   // face recognition event
   ALARM_TRAFFIC_MANUALSNAP                        = 0x00000118 + 300,    // traffic manual
snapshot event
    ALARM_TRAFFIC_FLOWSTATE                    = 0x00000119 + 300,   // traffic flow statistics
event
   ALARM_TRAFFIC_STAY                    = 0x0000011A + 300,   // traffic stay event
    ALARM_TRAFFIC_VEHICLEINROUTE                 = 0x0000011B + 300,    // vehicle in lane
event
    ALARM_MOTIONDETECT                            = 0x0000011C + 300,    // video move detection
event
   ALARM_LOCALALARM                        = 0x0000011D + 300,   // external alarm event
    ALARM_PRISONERRISEDETECTION                  = 0X0000011E + 300,    // rise detection
event
    ALARM_IVS_DENSITYDETECTION                   = 0X00000121 + 300,   // crowd detection
event
     ALARM_AUDIO_ABNORMALDETECTION                 = 0x00000126 + 300,    // abnormal
sound detection
   ALARM_IVS_B_ALARM_END,                                     // The above alarms are for IVS_B
server, cooperate with SDK
    ALARM_VIDEODIAGNOSIS                         = 0X00000120 + 300,   // video diagnosis result
event
   ALARM_IVS_V_ALARM                     = ALARM_VIDEODIAGNOSIS,
     ALARM_IVS_AUDIO_ABNORMALDETECTION              =  0x00000126 + 300,    //  sound
abnormal detection
   //福安看守所
   ALARM_CLIMB_UP                      = 0x00000128 + 300,   // climb detection
   ALARM_LEAVE_POST                      = 0x00000129 + 300,   // leave duty detection
   ALARM_VEHICLEACC                        = 0x00000130 + 300,   // mobile ACC outage alarm
 event
   ALARM_VEHICLE_TURNOVER                     = 0x00000131 + 300,   // vehicle side turn over
alarm  event
   ALARM_VEHICLE_COLLISION                    = 0x00000132 + 300,   // vehicle collision alarm
event
   ALARM_FIGHT                       = 0x00000133 + 300,   //fight
   ALARM_VIDEO_ABNORMAL                       = 0x00000136 + 300,   //video abnormal
   // new violation alarm type
   ALARM_VEHICLE_INBUSROUTE                    = 700,               // in bus lane event 41
   ALARM_BACKING                         = 701,               // illegal back event     42
   ALARM_RUN_YELLOWLIGHT                      = 702,               // run the yellow light event
43
   ALARM_PARKINGSPACE_PARKING                 = 703,               // occupied parking event
44
```

```
   ALARM_PARKINGSPACE_NOPARKING            = 704,              // free parking event     45
   ALARM_COVERINGPLATE                     = 705,
   ALARM_PARKINGONYELLOWBOX                = 706,
   ALARM_THROW                             = 707,              // traffic spilled material event    71
   ALARM_PEDESTRAIN                        = 708,              // traffic pedestrian event     7
   ALARM_COMPARE_PLATE                     = 715,              // plate contrast 79
   //以下为_m3.0新增
   ALARM_IVS_M_BEGIN                       = 800,              // _M3.0 special IVS alarm start
   ALARM_IVS_ALARM_CAPTURPIC               = 897,              // alarm snapshot
   ALARM_IVS_TIMING_CAPTURPIC              = 898,              // scheduled snapshot
   ALARM_IVS_CLIENT_CAPTURPIC              = 899,              // client snapshot
   ALARM_IVS_M_END                         = 900,              // _M3.0 special IVS alarm end
   ALARM_IVS_ABNORMAL_FACEDETECT           = 901,              // face detection event—
abnormal face detection
    ALARM_IVS_SIMILAR_FACEDETECT           = 902,              // face detection event—
adjacent face detection
   // ---ALARM_VIDEOABNORMALDETECTION alarm sub type start
   ALARM_IVS_VIDEOABNORMAL_SUBBEGIN    = 950,
                        ALARM_IVS_VIDEOABNORMAL_LOST                                      =
ALARM_IVS_VIDEOABNORMAL_SUBBEGIN,         // video abnormal event: video loss
                           ALARM_IVS_VIDEOABNORMAL_FREEZE                                 =
ALARM_IVS_VIDEOABNORMAL_SUBBEGIN + 1,     // video abnormal event: video frozen
                             ALARM_IVS_VIDEOABNORMAL_SHELTER                              =
ALARM_IVS_VIDEOABNORMAL_SUBBEGIN  +  2,         // video abnormal event: tampered
camera
                          ALARM_IVS_VIDEOABNORMAL_MOTION                                  =
ALARM_IVS_VIDEOABNORMAL_SUBBEGIN + 3,     // video abnormal event:move camera
                              ALARM_IVS_VIDEOABNORMAL_HIGHDARK                            =
ALARM_IVS_VIDEOABNORMAL_SUBBEGIN + 4,     // video abnormal event:too dark
                                     ALARM_IVS_VIDEOABNORMAL_HIGHBRIGHT                   =
ALARM_IVS_VIDEOABNORMAL_SUBBEGIN + 5,     // video abnormal event:too bright
                                 ALARM_IVS_VIDEOABNORMAL_COLORCAST                        =
ALARM_IVS_VIDEOABNORMAL_SUBBEGIN + 6,     // video abnormal event:color shift
                         ALARM_IVS_VIDEOABNORMAL_NOISE                                    =
ALARM_IVS_VIDEOABNORMAL_SUBBEGIN + 7,     // video abnormal event:noise
                                        ALARM_IVS_VIDEOABNORMAL_SCENE_CHANGE              =
ALARM_IVS_VIDEOABNORMAL_SUBBEGIN + 8,    // video abnormal event:scene change
   ALARM_IVS_VIDEOABNORMAL_SUBEND      = 960,
   // ---ALARM_VIDEOABNORMALDETECTION alarm sub type stop
    ALARM_IVS_ALARM_END                    = 1000,              // IVS device alarm type range  is
300-1000
   ALARM_OSD,                                                   // osd info
   ALARM_CROSS_INFO,                                                  // cross
   ALARM_CLIENT_PLATFORM_BEGIN          = 1100,               // client platform alarm start
    ALARM_DERELICTION                          = 1101,                  // abandoned object[traffic  event-
spilled material]
   ALARM_RETROGRADATION               = 1102,                // retrogradation [traffic event]
  ALARM_OVERSPEED                          = 1103,               // over speed    [traffic event]
  ALARM_LACK_ALARM                      = 1104,               // under speed [traffic event]
  ALARM_FLUX_COUNT                      = 1105,               // flow statistics[traffic event]
  ALARM_PARKING                           = 1106,               // parking detection[traffic event]
  ALARM_PASSERBY                         = 1107,               // pedestrian detection[traffic event]
  ALARM_JAM                                   = 1108,               // block detection[traffic event]
  ALARM_AREA_INBREAK                  = 1109,               // special area intrusion
  ALARM_OVERSPEED_MANUAL                  = 1123,                    // ANPR over speed alarm
 ,   PCS report to client,   client trigger manual alarm to ADS
    //face detection event detailed event (face detection type is detailed in DMS, so these two
definitions are useless)
    ALARM_TYPE_ALARM_FACEDETECT_NORMAL  = 1151,                      // face detection
event－normal face
    ALARM_TYPE_ALARM_FACEDETECT_UNNORMAL= 1152,                    // face detection
event－abnormal face
   ALARM_CLIENT_PLATFORM_END          = 1200,               // client platform alarm end
   ALARM_SYSTEM_BEGIN                    = 1200,               // alarm from system
   ALARM_HOST_TEMPRATURER             = 1201,                // host temperature too high
   ALARM_RAID_LOAD                        = 1202,               // raid degrade
   ALARM_SERVER_AUTO_MIGRATE          = 1203,               // server self-migrate
   ALARM_SERVER_MANUAL_MIGRATE         = 1204,               // server manual migrate
   ALARM_SERVER_STATUS_CHANGE         = 1205,               // server status change
   ALARM_MASTER_TO_BACKUP              = 1206,               // dual hot spare host switch  to
spare
   ALARM_BACKUP_TO_MASTER             = 1207,               // dual hot spare spare switch to
host
   ALARM_BACKUP_ABNORMAL             = 1208,               // dual spare spare failure
    ALARM_BACKUP_NORMAL                   = 1209,                // dual hot spare spare  failure
recover
   ALARM_SYSTEM_POWER_OFF             = 1214,               // system outage alarm 【city grid
outage】
    ALARM_SYSTEM_POWER_ON               = 1215,                // system power recover  alarm
【city grid recover】
   ALARM_SYSTEM_END                     = 1300,
   // -E video quality diagnosis new 12 types of alarm
    ALARM_VQDS_VIDEO_LOST               = 1500,               // video quality diagnosis-video
loss
   ALARM_VQDS_HIGHBRIGHT              = 1501,              // high brightness alarm
   ALARM_VQDS_HIGHBRIGHT_RED          = 1502,              // high brightness red alarm
```

```
    ALARM_VQDS_LOWBRIGHT                = 1503,             // low brightness alarm
    ALARM_VQDS_LOWBRIGHT_RED            = 1504,             // low brightness alarm
    ALARM_VQDS_CONTRAST                 = 1505,             // contrast alarm
    ALARM_VQDS_CONTRAST_RED             = 1506,             // contrast red alarm
    ALARM_VQDS_CLARITY                  = 1507,             // definition alarm
    ALARM_VQDS_CLARITY_RED              = 1508,             // definition red alarm
    ALARM_VQDS_COLOR_OFFSET             = 1509,             // color shift alarm
    ALARM_VQDS_COLOR_OFFSET_RED         = 1510,             // color shift red alarm
    ALARM_VQDS_DIAGNOSE_FAIL            = 1511,             // video quality diagnosis failed
    ALARM_ALARMHOST_MEDICAL             = 1604,             // medical alarm
    ALARM_ALARMHOST_URGENCY                 = 1605,             //   alarm host emergency
alarm
    ALARM_ALARMHOST_CATCH               = 1606,             // duress alarm
    ALARM_ALARMHOST_MENACE_SLIENCE      = 1607,             // mute menace
    ALARM_ALARMHOST_PERIMETER           = 1608,             // perimeter alarm
    ALARM_ALARMHOST_DEFENCEAREA_24H     = 1609,             // 24 hour zone alarm
    ALARM_ALARMHOST_DEFENCEAREA_DELAY   = 1610,             // delay zone alarm
    ALARM_ALARMHOST_DEFENCEAREA_INITIME  = 1611,             // intime zone alarm
    ALARM_ALARMHOST_BREAK               = 1612,             // vandal-proof
    ALARM_ALARMHOST_AUX_OVERLOAD        = 1613,             // AUX overload
    ALARM_ALARMHOST_AC_POWDOWN          = 1614,             // AC down
    ALARM_ALARMHOST_BAT_DOWN            = 1615,             // low battery
    ALARM_ALARMHOST_SYS_RESET           = 1616,             // system reset
    ALARM_ALARMHOST_PROGRAM_CHG         = 1617,             // battery down
    ALARM_ALARMHOST_BELL_CUT            = 1618,             // siren is cut or short circuit
    ALARM_ALARMHOST_PHONE_ILL           = 1619,             // phone cut or invalid
    ALARM_ALARMHOST_MESS_FAIL           = 1620,             // communication failed
    ALARM_ALARMHOST_WIRELESS_PWDOWN       = 1621,                 // wireless  sensor
under voltage
    ALARM_ALARMHOST_SIGNIN_FAIL         = 1622,             // failed to log in
    ALARM_ALARMHOST_ERR_CODE            = 1623,             // wrong password login
    ALARM_ALARMHOST_MANAUL_TEST         = 1624,             // manual test
    ALARM_ALARMHOST_CYCLE_TEST          = 1625,             // scheduled test
    ALARM_ALARMHOST_SVR_REQ             = 1626,             // server request
    ALARM_ALARMHOST_BUF_RST             = 1627,             // alarm buffer reset
    ALARM_ALARMHOST_CLR_LOG             = 1628,             // clear log
    ALARM_ALARMHOST_TIME_RST            = 1629,             // date time reset
    ALARM_ALARMHOST_NET_FAIL            = 1630,             // network error
    ALARM_ALARMHOST_IP_CONFLICT         = 1631,             // IP conflict
    ALARM_ALARMHOST_KB_BREAK            = 1632,             // keyboard vandal-proof
    ALARM_ALARMHOST_KB_ILL              = 1633,             // keyboard problem
    ALARM_ALARMHOST_SENSOR_O            = 1634,             // sensor open circuit
    ALARM_ALARMHOST_SENSOR_C            = 1635,             // sensor short circuit
    ALARM_ALARMHOST_SENSOR_BREAK          = 1636,                // sensor vandal-proof
    ALARM_FIRE_ALARM                      = 1637,                // alarm controller fire alarm
    ALARM_CALL_ALARM_HOST                 = 1652,                // phone alarm controller device
alarm
     ALARM_CALL_ALARM_HOST_CHN            = 1653,                // phone alarm controller
channel alarm
    //PE(PE) alarm -(SCS_ALARM_SWITCH_START name is from SCS PE file)
    //system project PE add alarmtype ALARM_SCS_BEGIN
    //ON/OFF, not controllable
    ALARM_SCS_SWITCH_START                = 1800,
    ALARM_SCS_INFRARED,                                          // IR alarm
    ALARM_SCS_SMOKE,                                             // smoke alarm
    ALARM_SCS_WATER,                                             // flood alarm
    ALARM_SCS_COMPRESSOR,                                        // compressor failure alarm
    ALARM_SCS_OVERLOAD,                                          // overload alarm
    ALARM_SCS_BUS_ANOMALY,                                       // bus abnormal
    ALARM_SCS_LIFE,                                              // life alarm
    ALARM_SCS_SOUND,                                             // sound alarm
    ALARM_SCS_TIME,                                              // clock alarm
    ALARM_SCS_FLOW_LOSS,                                         // flow loss alarm
    ALARM_SCS_FUSING,                                            // fuse alarm
    ALARM_SCS_BROWN_OUT,                                         // down alarm
    ALARM_SCS_LEAKING,                                           // leak alarm
    ALARM_SCS_JAM_UP,                                            // block alarm
    ALARM_SCS_TIME_OUT,                                          // timeout alarm
    ALARM_SCS_REVERSE_ORDER,                                     // reverse order alarm
    ALARM_SCS_NETWROK_FAILURE,                                   // group network failed alarm
    ALARM_SCS_UNIT_CODE_LOSE,                                    // unit code loss alarm
    ALARM_SCS_UNIT_CODE_DISMATCH,                                // unit code not match alarm
    ALARM_SCS_FAULT,                                             // failure alarm
    ALARM_SCS_UNKNOWN,                                           // unknown alarm
    ALARM_SCS_CUSTOM,                                            // custom alarm
    ALARM_SCS_NOPERMISSION,                                      // no right alarm
    ALARM_SCS_INFRARED_DOUBLE,                                   // IR dual alarm
    ALARM_SCS_ELECTRONIC_FENCE,                                  // e-fence alarm
    ALARM_SCS_UPS_MAINS,                                         // city grid normal/abnormal
    ALARM_SCS_UPS_BATTERY,                                       // battery normal/abnormal
     ALARM_SCS_UPS_POWER_SUPPLY,                                 // UPS normal outputbypass
power supply
    ALARM_SCS_UPS_RUN_STATE,                                     // UPS normal UPS failure
     ALARM_SCS_UPS_LINE_STYLE,                                   // UPS type id online UPS type as
backup
    ALARM_SCS_XC,                                                // small vehicle
```

```
   ALARM_SCS_DRQ,                                        // breaker
   ALARM_SCS_GLDZ,                                       // isolator
   ALARM_SCS_JDDZ,                                       // ground
   ALARM_SCS_IN_END,                                       // this value is not a cut off；here only mark
“ON/OFF, not controllable” end；
              //because   the   following“ON/OFF,   controllable”   not   marked   such   as
ALARM_SCS_DOOR_START
   //ON/OFF， controllable， be careful that the following ALARM_SCS_DOOR_SWITCH cannot
be BEGIN
   ALARM_SCS_DOOR_SWITCH          = 1850,            // access controller switch alarm
   ALARM_SCS_UPS_SWITCH,                                  // UPS switch alarm，
   ALARM_SCS_DBCB_SWITCH,                                 // cabinet switch alarm
   ALARM_SCS_ACDT_SWITCH,                                 // air condition alarm
   ALARM_SCS_DTPW_SWITCH,                                 // current power switch alarm
   ALARM_SCS_LIGHT_SWITCH,                                // light control switch alarm
   ALARM_SCS_FAN_SWITCH,                                  // fan controller switch alarm
   ALARM_SCS_PUMP_SWITCH,                                 // pump switch alarm
   ALARM_SCS_BREAKER_SWITCH,                                // breaker switch alarm
   ALARM_SCS_RELAY_SWITCH,                                // relay switch alarm
   ALARM_SCS_METER_SWITCH,                                // meter switch alarm
   ALARM_SCS_TRANSFORMER_SWITCH,                              // transformer switch alarm
   ALARM_SCS_SENSOR_SWITCH,                               // sensor switch alarm
   ALARM_SCS_RECTIFIER_SWITCH,                              // rectifier alarm
   ALARM_SCS_INVERTER_SWITCH,                               // inverter alarm
   ALARM_SCS_PRESSURE_SWITCH,                               // pressure switch alarm
   ALARM_SCS_SHUTDOWN_SWITCH,                                // shut down alarm
   ALARM_SCS_WHISTLE_SWITCH,                              // whistle alarm
   ALARM_SCS_SWITCH_END,
   //analog
   ALARM_SCS_ANALOG_START          = 1880,
   ALARM_SCS_TEMPERATURE,                                 // temperature alarm
   ALARM_SCS_HUMIDITY,                                  // humidity alarm
   ALARM_SCS_CONCENTRATION,                                // concentration alarm
   ALARM_SCS_WIND,                                       // fan speed alarm
   ALARM_SCS_VOLUME,                                    // capacity alarm
   ALARM_SCS_VOLTAGE,                                   // voltage alarm
   ALARM_SCS_ELECTRICITY,                                 // current alarm
   ALARM_SCS_CAPACITANCE,                                 // capacitance alarm
   ALARM_SCS_RESISTANCE,                                 // resistance alarm
   ALARM_SCS_CONDUCTANCE,                                 // conductance alarm
   ALARM_SCS_INDUCTANCE,                                 // inductance alarm
   ALARM_SCS_CHARGE,                                    // change alarm
   ALARM_SCS_FREQUENCY,                                  // frequency alarm
```

```
    ALARM_SCS_LIGHT_INTENSITY,                                // light intensity alarm (candela)
    ALARM_SCS_PRESS,                                      // press alarm （newton）
    ALARM_SCS_PRESSURE,                                       // pressure alarm （pa）
    ALARM_SCS_HEAT_TRANSFER,                                          // heat transfer alarm （Watts per
square meter）
    ALARM_SCS_THERMAL_CONDUCTIVITY,                               // thermal conductivity alarm
（kcal/(m*h*℃)）
    ALARM_SCS_VOLUME_HEAT,                                   // volume heat（kcal/(kg*℃)）
    ALARM_SCS_HOT_WORK,                                     // heat work alarm （joule）
    ALARM_SCS_POWER,                                      // power alarm （watt）
    ALARM_SCS_PERMEABILITY,                                   // permeability alarm （darcy）
    ALARM_SCS_PROPERTION,                                         // such as（Including voltage current
ratio, power factor, load unit %.）
    ALARM_SCS_ENERGY,                                     // energy（unit is J）
    ALARM_SCS_ANALOG_END,
    //ALARM_SCS_END,
    ALARM_IP_DEV_TALK                    = 1907,            // IP device intercom alarm
      ALARM_TYPE_UNIFY_BEGIN                      = 1908,                      //   alarm type unified
management，  no need to add EnumCenterRecType增加
    ALARM_VOICE_EXCEPTION                = 1909,               // audio abnormal alarm
    ALARM_RECORD_EXCEPTION               = 1910,               // record abnormal alarm
    ALARM_VOICE_LOSE                     = 1911,               // audio loss alarm
    ALARM_WIFITERM_FIND                  = 1912,               //WIFI end found alarm
    ALARM_WIFITERM_SURVEY                = 1913,               //WIFI end arm alarm
    ALARM_PTZ_DIAGNOSES                  = 1914,               // PTZ diagnosis info
    ALARM_SNAP_ALARM                     = 1915,               // general snapshot alarm
    ALARM_NO_DISK                        = 1916,               // no HDD alarm
    ALARM_DOUBLE_DEV_VERSION_ABNORMAL   = 1917,                 // dual control device
motherboard and backboard version do not match event
    ALARM_DCSSWITCH                      = 1918,               // host/spare switch event/group switch
alarm
    ALARM_DEV_RAID_FAILED                = 1919,               // device RAID error alarm
    ALARM_DEV_RAID_DEGRADED              = 1920,               // device RAID 降 级 alarm
    ALARM_BUF_DROP_FRAME                 = 1921,               // record buffer zone loss frame
alarm
    ALARM_VIDEO_UNFOCUS                  = 1997,               // video loss focus alarm
    ALARM_DEV_AUDIO_MUTATION             = 1998,               //audio mutation alarm
    ALARM_HEATIMG_TEMPER                 = 1999,               // thermal temperature test point
temperature abnormal alarm  event
    //AE_ALARM_TYPE_BEGIN                = 2000,
     ALARM_TYPE_RFID_BEGIN               = 2000,               //in protocol, RFID alarm type is
betweenAE_ALARM_TYPE_BEGIN and AE_ALARM_TYPE_END definition，  but RFID alarm
type is RFIDdevice  alarm，  not PEdevice alarm，  so here separate RFID alarm type
```

```
    ALARM_TYPE_RFID_BATTERY_EMPTY       = 2010,                 //radio frequency device
low battery alarm
    ALARM_TYPE_RFID_BUTTON              = 2011,                 //radio frequency device button
alarm
    ALARM_TYPE_RFID_DATA_EXCEPTION      = 2012,                 //radio frequency device
data error alarm
    ALARM_TYPE_RFID_ENTER_RECEIVER      = 2013,                 //radio frequency device
receiver sensor to cuff alarm
   ALARM_RFID_ILLEGAL_ENTER          = 2014,                //invalid enter
   ALARM_RFID_ILLEGAL_LEAVE          = 2015,                //invalid exit
   ALARM_RFID_ILLEGAL_GATHER          = 2016,               //invalid crowd
   ALARM_RFID_WITHOUT_TUTELAGE        = 2017,               //no care alarm
   ALARM_RFID_STAY                    = 2018,               //stay alarm
   ALARM_RFID_EXCEPTION               = 2019,               //abnormal alarm
   ALARM_RFID_CUTOFF_LABEL            = 2021,               //user tag cut
   ALARM_RFID_GPS                     = 2022,               //radio frequency device GPS report
   ALARM_RFID_APPROACH                = 2024,               //approach perimeter manager
   ALARM_RFID_LEAVEAWAY               = 2025,               //leave perimeter manager
   ALARM_TYPE_RFID_END,
   AE_ALARM_TYPE_BEGIN,
   ALARM_DOOR_MAGNETISM               = 2200,               // door sensor
   ALARM_PASSIVE_INFRARED             = 2201,               // passive IR
   ALARM_GAS                          = 2202,               // gas sensor
   ALARM_INITIATIVE_INFRARED          = 2203,               // active IR
   ALARM_GLASS_CRASH                  = 2204,               // glass broken
   ALARM_EXIGENCY_SWITCH              = 2205,               // emergency switch
   ALARM_SHAKE                        = 2206,               // vibration
   ALARM_BOTH_JUDGE                   = 2207,               // dual (IR+microwave)
   ALARM_THREE_TECHNIC                = 2208,               // three technologies
   ALARM_CALL_BUTTON                  = 2209,               // call button
   ALARM_SENSE_OTHER                  = 2210,               // other
   AE_ALARM_TYPE_END                  = 2400,
   //begin vibration fiber alarm typw
   ALARM_TYPE_VIBRATIONFIBER_BEGIN     = 2601,              // vibration fiber 1
   ALARM_VIBRATIONFIBER_SNLALARM,                           // ON/OFF alarm
   ALARM_VIBRATIONFIBER_BOXALARM,                           //  switch box alarm
   ALARM_VIBRATIONFIBER_INVALIDZONE,                        // zone invalid 1106
   ALARM_VIBRATIONFIBER_SIGNAL_OFF,                         // fiber signal source stop
   ALARM_VIBRATIONFIBER_FIBRE_BREAK,                        // fiber break
   ALARM_TYPE_VIBRATIONFIBER_END       = 2700,              // vibration fiber 5
   //end
   //patrol alarm
   ALARM_PATROL_BEGIN                 = 2701,
   ALARM_PATROL_EXCEPTION              = 2702,             // patrol abnormal alarm
   ALARM_PATROL_END                    = 2800,
   // -F reserve alarm type,   customalarm
   ALARM_TYPE_USERDEFINE_BEGIN         = 3101,
   ALARM_TYPE_USERDEFINE_END           = 3130,
   // alarm platform,   expend custom alarm type
   ALARM_TYPE_USERDEFINEEX_BEGIN       = 3201,
   ALARM_TYPE_USERDEFINEEX_END         = 4200,
   ALARM_NODE_ACTIVE                   = 4201,           // master slave switch alarm
   ALARM_ISCSI_STATUS                  = 4202,           // ISCSI storage status change alarm
   ALARM_OUTDOOR_STATIC                = 4203,
   ALARM_FALLING                       = 4204,           // fall event alarm
    ALARM_ITC_OUTSIDE_CARNUM            = 4205,             // entrance/exit external vehicle
alarm
   ALARM_POS_TRANING_MODE              = 4206,           //POS training mode alarm
   ALARM_REFUND_OVER_QUOTA             = 4207,           //return quota alarm
   ALARM_SWING_CARD_FREQUENTLY             = 4208,                 //member card frequent
appearance alarm
   ALARM_SIGNLE_COST_OVER_QUOTA        = 4209,           //sales over quota alarm
   // VDP device VTH new sensor alarm type
   ALARM_SENSE_BEGIN                   = 4299,
   ALARM_SENSE_DOOR                    = 4300,           //door sensor
   ALARM_SENSE_PASSIVEINFRA            = 4301,           //passive IR
   ALARM_SENSE_GAS                     = 4302,             //gas
   ALARM_SENSE_SMOKING                 = 4303,               //smoke
   ALARM_SENSE_WATER                   = 4304,               //water
   ALARM_SENSE_ACTIVEFRA               = 4305,               //active IR
   ALARM_SENSE_GLASS                   = 4306,               //glass broken
   ALARM_SENSE_EMERGENCYSWITCH         = 4307,                //emergency switch
   ALARM_SENSE_SHOCK                   = 4308,               //vibration
   ALARM_SENSE_DOUBLEMETHOD            = 4309,                //dual(IR+microwave)
   ALARM_SENSE_THREEMETHOD             = 4310,                //three technologies
   ALARM_SENSE_TEMP                    = 4311,               //temperature
   ALARM_SENSE_HUMIDITY                = 4312,               //humidity
   ALARM_SENSE_WIND                    = 4313,               //fan speed
   ALARM_SENSE_CALLBUTTON              = 4314,                //call button
   ALARM_SENSE_GASPRESSURE             = 4315,                //gas pressure
   ALARM_SENSE_GASCONCENTRATION        = 4316,                //gas concentration
   ALARM_SENSE_GASFLOW                 = 4317,               //gas flow
    ALARM_SENSE_OIL                    = 4319,                 //gas detection, gas, fuel and other oil
check
   ALARM_SENSE_MILEAGE                 = 4320,               //mileage detection
   ALARM_SENSE_URGENCYBUTTON= 4321,                          //emergency check
```

```
   ALARM_SENSE_STEAL                    = 4322,                //steal
   ALARM_SENSE_PERIMETER                = 4323,                //perimeter
   ALARM_SENSE_PREVENTREMOVE            = 4324,                //vandal-proof
   ALARM_SENSE_DOORBELL                 = 4325,                //door bell
   ALARM_SENSE_LOCK_LOCKKEY             = 4326,                //lock key alarm
   ALARM_SENSE_LOCK_LOWPOWER            = 4327,                //lock low voltage alarm
   ALARM_SENSE_LOCK_PREVENTREMOVE       = 4328,                //lock vandal proof
   ALARM_SENSE_LOCK_FORCE               = 4329,                //lock duress alarm
   ALARM_SENSE_END                      = 4399,
   ALARM_STORAGE_BEGIN                  = 4400,
   ALARM_IO_QUEUE_FULL                  = 4401,                // disk read write high load
   ALARM_DISK_DESTROY                   = 4402,                // disk error
   ALARM_IPSAN_OFF_LINE                 = 4403,                // IPSan offline
   ALARM_NO_DISK_STORAGE                = 4404,                // no disk
   ALARM_GET_STREAM_ERROR               = 4405,                // get stream error
   ALARM_STORAGE_END                    = 4499,
   //Entrance/exit ANPR blacklist new alarm type
   ALARM_TRAFFIC_SUSPICIOUSCAR          = 4501,
   //Entrance /exit controller alarm type
   ALARM_SLUICE_BEGIN                   = 4502,
    ALARM_SLUICE_IC_CARD_STATUS_LOWCARD                = 4503,      //card box missing
card alarm
    ALARM_SLUICE_IC_CARD_STATUS_NOCARD                 = 4504,      //card box no card
alarm
   ALARM_SLUICE_IC_CARD_STATUS_FULLCARDS          = 4505,    //card box full alarm
    ALARM_SLUICE_CAR_DETECTOR_STATE_OFFLINE          = 4506,      //vehicle detector
offline alarm
     ALARM_SLUICE_CAR_DETECTOR_STATE_LOOPOFFLINE       = 4507,       //GND coil
offline alarm
   ALARM_SLUICE_LED_DEV_STATE_OFFLINE             = 4508,     //LED offline alarm
     ALARM_SLUICE_SWIPING_CARD_DEV_STATE_OFFLINE       = 4509,       //Panel card
board offline alarm
    ALARM_SLUICE_DELIVE_CARD_DEV_OFFLINE               = 4510,      //issue card board
offline alarm
   ALARM_SLUICE_SPEAK_DEV_STATUS                  = 4511,     //talk event alarm
   ALARM_SLUICE_END                               = 4550,
   //Self-cashier device alarm type
   ALARM_SELFPAY_BEGIN                            = 4551,
   ALARM_SELFPAY_NOPAPER                          = 4552,//missing paper
   ALARM_SELFPAY_NOCASH50                         = 4553,
   ALARM_SELFPAY_NOCASH20                         = 4554,
   ALARM_SELFPAY_NOCASH10                         = 4555,
   ALARM_SELFPAY_NOCASH1                          = 4556,
   ALARM_SELFPAY_NOCOIN                        = 4557,
   ALARM_SELFPAY_LOCKMONEY                     = 4558,    //card money
   ALARM_SELFPAY_DISMANTLE                     = 4559,    //vandal proof
   ALARM_SELFPAY_UNPACK                        = 4560,    //open box
   ALARM_SELFPAY_UNKONWN                       = 4561,    //unknown money
   ALARM_SELFPAY_END                           = 4580,
   //client IP talk alarm
   ALARM_IP_DEV_BEGIN                          = 4700,
   ALARM_IP_DEV_CALLIN                         = 4701,     //extension call
   ALARM_IP_DEV_CALLOUT                        = 4702,     //dial
   ALARM_IP_DEV_END                            = 4800,
   //mobile phone APP alarm type
   ALARM_MOBILEAPP_BEGIN                       = 4900,
   ALARM_MOBILEAPP_GPS                         = 4901,     //mobile phone APP upload GPS
   ALARM_MOBILEAPP_ONE_CLICK                   = 4902,     //mobile phone APP one-click
alarm
   ALARM_MOBILEAPP_MANUAL_ADD                  = 4903,     //mobile phone APP manual
add alarm
   ALARM_MOBILEAPP_END                         = 5000,
   //scene alarm start
   ALARM_SCENE_BEGIN                           = 5001,
   ALARM_PEOPLE_UPPER_LIMIT                    = 5002,     //user limit
   ALARM_PEOPLE_LOWER_LIMIT                    = 5003,     //lower limit
   ALARM_INFLUX_UPPER_LIMIT                    = 5004,     //user flow over limit (in)
   ALARM_OUTFLUX_UPPER_LIMIT                   = 5005,     //user flow over limit (out)
   ALARM_DENSITY_UPPER_LIMIT                   = 5006,     //density alarm
   ALARM_SCENE_EXCEPTION                       = 5007,     //scene error alarm
   ALARM_SCENE_END                             = 5100,
   //scene alarm end
   // thermal image alarm
    ALARM_RADIOMETRY_HEATIMG_TEMPER                    = 5120,          //thermal image
temperature abnormal alarm
    ALARM_RADIOMETRY_FIRE_WARNING                      = 5121,          //thermal image fire
alarm
   ALARM_RADIOMETRY_FIREWARNING_INFO                   = 5122,          //thermal image fire
condition alarm
    ALARM_RADIOMETRY_HOTSPOT_WARNING                   = 5123,          //thermal image
hotspot abnormal alarm
    ALARM_RADIOMETRY_COLDSPOT_WARNING                  = 5124,          //thermal image
cold spot abnormal alarm
    ALARM_RADIOMETRY_DIFFERENCEG_TEMPER                = 5125,          //thermal image
temperature test rule difference alarm
   //client stb device custom alarm
```

```
   ALARM_STB_BEGIN                                = 5200,
   ALARM_STB_FIRE                                 = 5201,         //fire alarm
   ALARM_STB_CRIME                                = 5202,         //theft alarm
   ALARM_STB_EMERGENCY                            = 5203,         //emergency center
   ALARM_STB_OTHER                                = 5204,         //other alarm
   ALARM_STB_END                                  = 5250,
   //-C/-P new alarm reserve
   ALARM_DSSC_BEGIN                               = 5300,
  ALARM_PATIENTDETECTION_TYPE_CROSS_REGION          = ALARM_DSSC_BEGIN +
  1, // alert zone alarm ,   maybe patient leave or someone approach patient
    ALARM_PATIENTDETECTION_TYPE_LIGHT_OFF           = ALARM_DSSC_BEGIN + 2,
// ward light off
    ALARM_PATIENTDETECTION_TYPE_STOP_DETECTION      = ALARM_DSSC_BEGIN
+ 3, // disarm,   do not monitor patient
    ALARM_PATIENTDETECTION_TYPE_START_DETECTION     = ALARM_DSSC_BEGIN
+ 4, // start arm
   ALARM_DSSC_END                                 = 5400,
   //VTA alarm tower alarm
   ALARM_U700_BEGIN                               = 5401,
    ALARM_VTA_INSPECTION                          = ALARM_U700_BEGIN + 1, // VTA alarm
tower patrol alarm
    ALARM_VTA_OVERSPEED                           = ALARM_U700_BEGIN + 2, // VTA alarm
tower over speed alarm
     ALARM_VTA_INSPECTION_SWING_CARD                  = ALARM_U700_BEGIN + 3,
//VTA petrol card
    ALARM_VTA_PATROL_SWING_CARD                   = ALARM_U700_BEGIN + 4, //VTA
patrol card
   ALARM_U700_END                                 = 5500,
   //MCS alarm
    ALARM_MCS_CAPACITY_LOW                        = 11600,            // micro cloud general
capacity alarm config
   ALARM_MCS_DATANODE_OFFLINE                     = 11601,            // micro cloud storage
node offline
   ALARM_MCS_DISK_OFFLINE                         = 11602,            // micro cloud disk offline
alarm config
   ALARM_MCS_DISK_SLOW                            = 11603,            // disk slow alarm config
    ALARM_MCS_DISK_BROKEN                         = 11604,            // disk damage alarm
config
    ALARM_MCS_DISK_UNKNOWERROR                    = 11605,            // disk unknown
error alarm config
    ALARM_MCS_METADATA_SERVER_ABNORMAL            = 11606,            // metadata
server error alarm config
    ALARM_MCS_CATALOG_SERVER_ABNORMAL             = 11607,            // directory
```

```
server error alarm config
    ALARM_MCS_GENERAL_CAPACITY_RESUME              = 11608,              // micro cloud
general capacity recover event
    ALARM_MCS_DATA_NODE_ONLINE                     = 11609,              // micro cloud storage
node online event
    ALARM_MCS_DISK_ONLINE                          = 11610,              // micro cloud disk online
event
    ALARM_MCS_METADATA_SLAVE_ONLINE                = 11611,              // micro cloud
metadata spare online event
    ALARM_MCS_CATALOG_SERVER_ONLINE                = 11612,              // micro cloud
directory server online event
} Alarm_type_e;
```

Father title : structure

AlarmCategory_e

Type of alarm

```
Typedef Enum
{
    COMMON_ALARM     = 1,     // Common alarm
    HOST_ALARM       = 2,     // Alarm host alarm
    DOOR_ALARM       = 3,     // Door guard alarm
    GPS_ALARM        = 4,     // GPSCall the police
    IVSF_ALARM       = 5,     // Face alarm
    SYSTEM_ALARM     = 6,     // System event alarm
    PE_ALARM_TYPE    = 7,     //Ring alarm type
}AlarmCategory_e;
```

Father theme:structural morphology

DPSDK_ALARMLINKVEDIO_INFO

Alarm linkage video information

```
Typedef Struct
{
    DPSDK_CHAR  SzLinkVedioId[DPSDK_ALARM_CHANNELID_LEN];    // Linkage video channel ID
    DPSDK_INT32 IStreamType;                                  // Code stream type
}DPSDK_ALARMLINKVEDIO_INFO;
```

Father theme:structural morphology

Structures

DPSDK_QUERY_ORG_INFO
Condition for Organization Query

DPSDK_REALPLAY_PARAM
Unicast Video Parameter

DPSDK_LOG_LEVEL_TYPE
Log Level Type

DPSDK_DEV_ALL_INFO_LIST
Device List

DPSDK_REALPLAY_PARAM
Record File List

DPSDK_STREAM_MODE
Play Mode

DPSDK_VIDEO_LOCK_TYPE

DPSDK_PIC_FORMAT
Screenshot Picture Format

DPSDK_CONVERT_BMP
Convert Picture to BMP Format

DPSDK_FISH_TYPE
Fisheye Type

DPSDK_MHFPTZ_INIT_PARAM
SmartTrack Initialized Channel Parameter

DPSDK_FISH_OPTPARAM
Fisheye Parameter

DPSDK_FISH_UPDATE_PARAM

DPSDK_FISH_EPTZPARAM
EPTZ Parameter

DPSDK_FISH_PARAMS

Fisheye Parameter

DPSDK_IVSE_INFO
Parameter Input by Video Enhancing Algorithm

DPSDK_DISPLAY_RECT
Split Algorithm Rect Display

DPSDK_PICTURE_MONITOR
Parameter Structure of Bayonet Picture Monitor

DPSDK_DECODE_TYPE
Decoding Type

DPSDK_SPLIT_TRECE_TYPE
4K Split Type

DPSDK_IVS_VISIBLE
Intelligence

DPSDK_BAYONET_DICTIONARY_TYPE
Dictionary Type of Bayonet Monitor

DPSDK_VAX_BUF_TYPE
Buffer Type

DPSDK_SUB_CODE_TYPE
Node Type of Organization Query

DPSDK_CONFIRMALARM_PARAM
Alarm Confirmation Parameter

DPSDK_QUERYALARM_PARAM
Alarm Query Parameter

DPSDK_ALARM_DETAILINFO_LIST
Alarm Information List

DPSDK_QUERYALARMCOUNT_PARAM
Parameter of Alarm Total Query

DPSDK_ALARMPROCESS_DETAILINFO_LIST
Alarm Processing Information List

DPSDK_BLOCKALARM_PARAM
Block Alarm Parameter

DPSDK_ALARMEXPORT_PARAM
Alarm Export Parameter

DPSDK_PTZOPERATE_FUNCTION_PARAM
PTZ Function Operation Parameter

DPSDK_PTZOPERATE_RESULT
PTZ Function Operation Result

DPSDK_PTZOPERATE_CAMERA_PARAM
Parameter to Operate PTZ Camera

DPSDK_PTZOPERATE_DIRECT_PARAM
Parameter to Control PTZ Direction

DPSDK_PTZOPERATE_FOCUS_PARAM
Motorized Focusing Control Parameter

DPSDK_PTZOPERATE_PRESETPOINT_PARAM
Preset Point Control Parameter

DPSDK_PTZOPERATE_SITPOSITION_PARAM
Three-Dimensional Positioning Parameter

DPSDK_PTZOPERATE_ARRANGEPTZ_PARAM
Unlock Parameter

DPSDK_DEV_UNIT_TYPE
Unit Type

MEDIA_ENCHANGE

MEDIA_DISPLAY

DPSDK_ALARMEVENT_NOTIFY
Alarm Event (Notification)

DPSDK_ALARMCONFIRM_NOTIFY
Alarm Confirmation (Notification)

DPSDK_ALARM_DETAILINFO_NOTIFY
Alarm Information (Notification)

DPSDK_ALARMEXPORT_RESULT_NOTIFY
Alarm Export Result (Notification)

DPSDK_DEV_STATUS_NOTIFY
Device Status Change Notification

DPSDK_CHANNEL_STATUS_NOTIFY
Channel Status Change Notification

DPSDK_ORG_BASE_INFO
Organization Base Data

DPSDK_MOVE_ORG_NOTIFY
Moving Organization Notification

DPSDK_ADD_DEVICE_NOTIFY
Adding Device Notification

DPSDK_MODIFY_DEVICE_NOTIFY
Modifying Device Notification

DPSDK_DELETE_DEVICE_NOTIFY
Device Notification Supports Batch Removal

DPSDK_MOVE_DEVICE_NOTIFY
Device Notification Supports Batch Move

DPSDK_USERONLINESTATUS_NOTIFY
User Online Status Notification

DPSDK_USERADD_NOTIFY
User Adding Notification

DPSDK_USERDELETE_NOTIFY
User Deletion Notification

DPSDK_VIEWINFO_CHANGED_NOTIFY
Visible Range Change Notification

DPSDK_DEVICELOCATION_NOTIFY
Notification of Device Adding / Change on Bitmap

DPSDK_LOCKSTATUS_CHANGED_NOTIFY
PTZ Lock Status Change Notification

DPSDK_RADERFRAME_NOTIFY
Rader Information Report Notification

DPSDK_FACE_INFO_NOTIFY
Face Snap Information Notification

DPSDK_PERSONTYPE_NOTIFY
Person Type Change Notification

DPSDK_USERDEFINE_INFO
User Defined Data Change

DPSDK_POS_DATA_NOTIFY

DPSDK_BITMAP_INFO_NOTIFY
Structure of Bitmap Change Notification

DPSDK_SMARTTRACKOBJECT_NOTIFY
SmartTrack Object Notification

DPSDK_CAR_SURVEY_ALARM
Bayonet Survey Alarm

DPSDK_TVWALL_NOTIFY
TV Wall Notification (Add / Modify)

DPSDK_EVENT_PARAM
Event Callback Parameter Structure

DPSDK_COMPRESS_TYPE
Compressing Type

Alarm_type_e
Alarm Type

AlarmLevel_e
Alarm Level

AlarmState_e
Type of Alarm Occurrence

AlarmDealWith_e
Alarm Processing Status

DPSDK_FISH_MOUNTMODE
Fisheye Mounting Mode

DPSDK_FISH_SHOWMODES
Fisheye Image Display Mode

DPSDK_FISH_EPTZCMD
EPTZ Move Options

AlarmObject_e
Alarm Object Type

AlarmCategory_e
Alarm Category

DPSDK_USER_STATUS
User Status

DPSDK_DEV_STATUS
Front End Status

DPSDK_IVSE_FUNC_TYPE
IVSE Function Type Enum

DPSDK_ALARMLINKVEDIO_INFO
Alarm Linked Video Information

DPSDK_EMAILADDRESS
Email Address

DPSDK_DEV_ALL_INFO
Device Data

DPSDK_DEV_INFO
Device Basic Information

DPSDK_ENC_CHANNEL_INFO
Encoding Channel Information

DPSDK_DEC_CHANNEL_INFO
Decoding Channel Information

DPSDK_ALARMIN_CHANNEL_INFO
Alarm in Channel Information

DPSDK_ALARMOUT_CHANNEL_INFO
Alarm out Channel Information

DPSDK_TVWALLIN_CHANNEL_INFO
TV Wall Input Channel Information

DPSDK_TVWALLOUT_CHANNEL_INFO
TV Wall Output Channel Information

DPSDK_DOOR_CHANNEL_INFO
Door Access Control Channel Information

DPSDK_VOICE_CHANNEL_INFO
Voice Channel Information

DPSDK_ROADGATE_CHANNEL_INFO
Road Gate Channel Information

DPSDK_LED_CHANNEL_INFO
LED Channel Information

DPSDK_DISPATCHER_CHANNEL_INFO
Dispatcher Channel Information

DPSDK_POS_CHANNEL_INFO
POS Channel Information

DPSDK_VIRTUAL_CHANNEL_INFO
Virtual Channel Information

DPSDK_DEV_TYPE
Device Type (Shall Be Consistent with Web)

DPSDK_BASE_CHANNEL_INFO

Channel Base Information

DPSDK_CAMERA_TYPE
Camera Type

DPSDK_CHANNEL_REMOTE_TYPE
M60 M30 M70 Remote Channel Type

DPSDK_DECODE_MODE
Video Source Mode of Decoder

DPSDK_CHANNEL_TYPE
Channel Type

DPSDK_RADER_TARGET_INFO
Rader Target Information

DPSDK_FACE_INFO
Similar Face Information

DPSDK_ALERT_USERDEFINEDATA_TYPE
Change Type of User Defined Data

DPSDK_MEDIA_BASE_PARAM
Video Base Parameter

DPSDK_MEDIA_CALLBACK
Video Callback Structure

DPSDK_FILE_STORE_INFO
Record File Path

DPSDK_LEN_TYPE
Lens Type

DPSDK_CAM_TYPE
Fixed Camera Type

DPSDK_SUBORDINATE_CAMCONFIGPARAM
Externally Configured Subordinate Camera Parameter (SmartTrack)

DPSDK_IVSE_ROI
ROI Data Type Definition

DPSDK_ALARM_DETAILINFO
Alarm Information

DPSDK_ALARMPROCESS_DETAILINFO
Alarm Processing Record

DPSDK_DEVICECODE_INFO
Device Code Information

DPSDK_DEMUXDEC_CALLBACK
Data Callback after Source Data Analysis

DPSDK_EVENT_CALLBACK
Event Callback Function

DPSDK_DataCallback
Data Synchronized Callback. For Upper Level Data Copy

DPSDK_TVWALL_PLAYBACK_CALLBACK
TV Wall Playback Callback Function

DPSDK_ORG_INFO
Organization Base Data

DPSDK_DATA_TYPE
Data Type Definition of Synchronized Callback Function

DPSDK_ORG_SUB_DEV_INFO
Device Information of the Organization Tree

DPSDK_ORG_SUB_CHANNEL_INFO
Channel Information of the Organization Tree

DPSDK_COLLECTION_ORG_INFO
Organization Information Collection

DPSDK_LAYERED_RESULT_LIST
Layered Result List of the Organization Tree

DPSDK_LAYERED_RESULT
Layered Result of the Device Tree

DPSDK_NODE_TYPE
Node Type

DPSDK_ALL_ORG_INFO
All Organization Information (Recursive Tree)

DPSDK_SINGLE_ORG_INFO
Single Organization Information

DPSDK_GET_DEVICE_LAYERED_PARAM
Get by Layer the Request Parameter of the Organization Tree

DPSDK_MULITCAST_REALPLAY_PARAM
Multicast Video Parameter

DPSDK_MULITVIEW_REALPLAY_PARAM
Multiscreen Preview Video Parameter

DPSDK_RECORD_STATUS_INFO
Channel Record Information

DPSDK_RECORD_STATUS
Record Status

DPSDK_QUERY_RECORD_PARAM
Query Record Information

DPSDK_RECORD_INFO_LIST
Record Information

DPSDK_RECORD_TYPE
Record Type

DPSDK_SOURCE_TYPE
Record Source

DPSDK_SINGLE_RECORD_INFO
Single Record Information

DPSDK_QUERY_RECORD_DATE_PARAM
Query Record Date

DPSDK_RECORD_DATE_INFO

DPSDK_LOCK_RECORD_FILE_PARAM
Lock Record File

DPSDK_LOCK_RECORD_FILE_RESULT
Lock or Unlock Record File Result

DPSDK_UNLOCK_RECORD_FILE_PARAM
Unlock Record File

DPSDK_PLAYBACK_BY_FILE_PARAM
Parameter Playback by Record File

DPSDK_PLAY_DIRECTION
Play Direction

DPSDK_PLAYBACK_BY_TIME_PARAM
Parameter Playback by Time

DPSDK_PTZOPERATE_STARTREMOTERECORD_PARAM
Parameter to Open Manual Recording

DPSDK_PTZOPERATE_REMOTERECORD_RESULT
Open / Stop Manual Recording Result

DPSDK_PTZOPERATE_STOPREMOTERECORD_PARAM
Parameter to Close Manual Recording

DPSDK_DOWNLOAD_BY_TIME_PARAM
Download Parameter by Time

DPSDK_EVENT_DOWNLOAD_CALLBACK
Record Event Callback Function

DPSDK_DOWNLOAD_BY_FILE_PARAM
Download Parameter by File

DPSDK_DOWNLOAD_RECORD_INFO
Record Download Information Parameter

DPSDK_PTZOPERATE_ALARMOUT_PARAM
Alarm Output Control Parameter

DPSDK_PTZ_PRESETPOINT_LIST
Get Preset Point List

DPSDK_PTZ_PRESETPOINT_INFO
Get Preset Point Information

DPSDK_PAGE_INFO
Page Information

DPSDK_TVWALL_LIST
TV Wall List

DPSDK_TVWALL_BASE_INFO
TV Wall Base Information

DPSDK_TVWALL_INFO
TV Wall Information

DPSDK_SCREEN_DECODER_LIST
Screen Decoder Channel List

DPSDK_SCREEN_DECODER_INFO
Screen Decoder Channel Information

DPSDK_TVWALL_SCREEN_POS
TV Wall Screen Position

DPSDK_COMBINED_SCREEN_INFO
General Screen Information under the Combined Screen

DPSDK_TVWALL_TASK_LIST
TV Wall Task List

DPSDK_TVWALL_TASK_BASE_INFO
TV Wall Base Task Information

DPSDK_TVWALL_TASK_INFO
TV Wall Task Information

DPSDK_TVWALL_TASK_SCREEN_OPER_LIST
TV Wall Task Screen Operation Information List

DPSDK_TVWALL_TASK_CHANNEL_EXT_LIST
Device Information List Needed by the Video Source (Directly Connected Decoder)

DPSDK_TVWALL_TASK_CHANNEL_EXT_INFO
Device Information Needed by the Video Source (Decoder is directly connected)

DPSDK_TVWALL_TASK_SCREEN_OPER_INFO
Operation Information on the Task Screen of TV Wall

DPSDK_TVWALL_WND_INFO
TV Wall Window Information

DPSDK_TVWALL_VIDEO_SOURCE_INFO
Video Source Information

DPSDK_TVWALL_SUBWND_INFO
TV Wall Sub-Window Information

DPSDK_TVWALL_TASK_INFO_LIST
TV Wall Task Information

DPSDK_TVWALL_PROJECT_LIST
TV Wall Project List

DPSDK_TVWALL_PROJECT_INFO
TV Wall Project Information

DPSDK_TVWALL_PROJECT_TASK_INFO
Task Information

DPSDK_CURRENT_TVWALL_TASK_LIST
Current TV Wall Task List

DPSDK_CURRENT_TVWALL_TASK_INFO
Current TV Wall Task Information

DPSDK_TVWALL_WINDOW_INFO
Window Information

DPSDK_TVWALL_OPEN_WINDOW
Open Window

DPSDK_TVWALL_CONTROL_INFO

TV Wall Control

DPSDK_TVWALL_CONTROL_TYPE
TV Wall Control Command

DPSDK_ADD_RELATION_NOTIFY

DPSDK_MODIFY_RELATION_NOTIFY

DPSDK_DELETE_RELATION_NOTIFY

DPSDK_LOGIN_PARAM

DPSDK_CLIENT_TYPE

DPSDK_STREAM_TYPE

DPSDK_LINKED_CHANNEL

DPSDK_FISH_MODEINITPARAM

DPSDK_FISH_SIZE

DPSDK_FISH_OUTPUTFORMAT

DPSDK_MHFPTZ_CONFIGPARAM

DPSDK_FISH_REGIONPARAM

DPSDK_FISH_SUBMODE

DPSDK_FISH_POINT2D

DPSDK_FISHEYE_CALLBACK

DPSDK_DEV_STATUS

Front-end state

```
Typedef Enum
{
    DEV_STATUS_UNDEFINE = 0,     // Unknown
    DEV_STATUS_ONLINE = 1,       // On-line
    DEV_STATUS_OFFLINE,          // Off-line
    DEV_STATUS_FORBID,           // Disable
}DPSDK_DEV_STATUS;
```

Father theme:structural morphology

DPSDK_DEV_ALL_INFO

Device data

```
Typedef Struct
{
    DPSDK_DEV_INFO StruDevInfo;                    // Device data
    // Coding channel
    DPSDK_INT32 IEncChnlNum;                      // Number of coded channels
    DPSDK_ENC_CHANNEL_INFO * PEncChnlInfoList;            // Code channel list
    // Decoding channel
    DPSDK_INT32 IDecChnlNum;                      // Decode channel number
    DPSDK_DEC_CHANNEL_INFO  * PDecChnlInfoList;            // Decode channel list
    // Alarm input channel
    DPSDK_INT32 IAlarmInChnlNum;                    // Number of alarm input
    DPSDK_ALARMIN_CHANNEL_INFO * PAlarmInChnlInfoList;      // Alarm input list
    // Alarm output channel
    DPSDK_INT32 IAlarmOutChnlNum;                   // Alarm output number
    DPSDK_ALARMOUT_CHANNEL_INFO * PAlarmOutChnlInfoList;    // Alarm output list
    // Large screen input channel
    DPSDK_INT32 ITvWallInChnlNum;                   // Large screen input channel number
    DPSDK_TVWALLIN_CHANNEL_INFO * PTvWallInChnlInfoList;      // Large screen input
channel list
    // Large screen output channel
    DPSDK_INT32 ITvWallOutChnlNum;                   // Large screen output channel number
    DPSDK_TVWALLOUT_CHANNEL_INFO * PTvWallOutChnlInfoList;     // Large screen
output channel list
    // Entrance guard channel
    DPSDK_INT32 IDoorChnlNum;                      // Number of access channels
    DPSDK_DOOR_CHANNEL_INFO * PDoorChnlInfoList;           // List of access channels
    // Voice channel
    DPSDK_INT32 IVoiceChnlNum;                     // Voice channel number
    DPSDK_VOICE_CHANNEL_INFO * PVoiceChnlInfoList;          // Voice channel list
    // Channel gate
    DPSDK_INT32 IRoadGateChnlNum;                   // Number of channel gates
    DPSDK_ROADGATE_CHANNEL_INFO * PRoadGateChnlInfoList;    // List of channel gates
    // LEDpassageway
    DPSDK_INT32 ILEDChnlNum;                      // LED Number of channels
    DPSDK_LED_CHANNEL_INFO * PLEDChnlInfoList;            // LED Channel list
    // Dispatcher channel
    DPSDK_INT32  IDispatcherChnlNum;                        // Number of channels for the
dispatcher
    DPSDK_DISPATCHER_CHANNEL_INFO * PDispatcherChnlInfoList; // Scheduler list
    // POSpassageway
    DPSDK_INT32 IPosChnlNum;                          // POS Number of channels
    DPSDK_POS_CHANNEL_INFO * PPosChnlInfoList;              // POS Channel list
    // virtual channel
    DPSDK_INT32 IVirtualChnlNum;                      // Number of virtual channels
    DPSDK_VIRTUAL_CHANNEL_INFO * PVirtualChnlInfoList;      // Virtual channel list
} DPSDK_DEV_ALL_INFO;
```

Father theme:structural morphology

DPSDK_USER_STATUS

User Status

```
typedef enum
{
    USER_OFFLINE,            // Offline
    USER_ONLINE,             // Online
}DPSDK_USER_STATUS;
```

Father theme:structural morphology

DPSDK_DEVICECODE_INFO

Device coding information

```
Typedef Struct
{
    DPSDK_CHAR SzDeviceCode[DPSDK_DEVICE_CODE_LEN]; // Device coding
}DPSDK_DEVICECODE_INFO;
```

Father theme:structural morphology

DPSDK_RADER_TARGET_INFO

Radar target information

```
Typedef Struct
{
    DPSDK_INT32 IId;        // information Id
    DPSDK_INT32 ITargetId; // target ID
    DPSDK_FLOAT FTargetLen; // Target length
    DPSDK_FLOAT FSpeed_X;   // XDirection velocity
    DPSDK_FLOAT FSpeed_Y;   // YDirection velocity
    DPSDK_FLOAT FCod_X;     // Xcoordinate
    DPSDK_FLOAT FCod_Y;     // Ycoordinate
    DPSDK_FLOAT FDistance; // distance
    DPSDK_FLOAT FAzimuth;   // Azimuth
    DPSDK_FLOAT FSNR;       // Target signal-to-noise ratio
    DPSDK_FLOAT FEN;        // Target peak energy
}DPSDK_RADER_TARGET_INFO;
```

Father theme:structural morphology

DPSDK_FACE_INFO

Similar face information

```
Typedef Struct
{
    DPSDK_CHAR SzFaceImageUrl[DPSDK_URL_LEN];    // Face map
    DPSDK_CHAR SzName[DPSDK_NAME_LEN];           // Full name
    DPSDK_INT32 IGender;                         // Gender 0-Unknown 1-male 2-female
    DPSDK_CHAR SzBirthday[DPSDK_BIRTHDAY_LEN];   // Birthday
    DPSDK_INT32 IPersonType;                     // Type of personnel
    DPSDK_CHAR SzPersonId[DPSDK_PERSON_ID_LEN]; // personnel Id
    DPSDK_FLOAT FSimilarity;                     // Similarity degree
    DPSDK_BOOL BSurveillance;                    // Whether dispatched
}DPSDK_FACE_INFO;
```

Father theme:structural morphology

DPSDK_ALERT_USERDEFINEDATA_TYPE

User defined data change type

```
Typedef Enum
{
    AddUserDefineData,          //New user custom data
    ModifyUserDefineData,       //Modifying user custom data
    DeleteUserDefineData,       //Delete user custom data
}DPSDK_ALERT_USERDEFINEDATA_TYPE;
```

Father theme:structural morphology

DPSDK_LINKED_CHANNEL

```
Typedef Struct
{
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; // channel ID
} DPSDK_LINKED_CHANNEL;
```

Father theme:structural morphology

DPSDK_CLIENT_TYPE

```
Typedef Enum
{
    CLIENT_PC = 1,       // PC Client
    CLIENT_MAC = 2,      // MAC Client
    CLIENT_ANDROID = 3, // Android client
    CLIENT_IPHONE = 4,   // Iphone Client
    CLIENT_PAD = 5,      // PAD Client
    CLIENT_WEB = 6,      // WEB Client
    CLIENT_OTHER = 7,    // Other
}DPSDK_CLIENT_TYPE;
```

Father theme:structural morphology

DPSDK_SUB_CODE_TYPE

Organization Query Node Type

```
typedef enum
{
    SUB_SIGNAL_CODE,                                              // First-level Child Node
    SUB_ALL_CODE,                                                 // All Child Nodes
}DPSDK_SUB_CODE_TYPE;
```

Father theme:structural morphology

DPSDK_DEV_UNIT_TYPE

Unit Type

```
typedef enum
{
   DEV_UNIT_UNKOWN,                                    // Unknown
   DEV_UNIT_ENC,                                       // Encoding
   DEV_UNIT_DEC,                                       // Decoding
   DEV_UNIT_ALARMIN,                                   // Alarm Input
   DEV_UNIT_ALARMOUT,                                  // Alarm Output
   DEV_UNIT_TVWALLIN,                                  // TV Wall Input
   DEV_UNIT_TVWALLOUT,                                 // TV Wall Output
   DEV_UNIT_DOORCTRL,                                  // Access Control
   DEV_UNIT_VOICE,                                     // Video Intercom
   DEV_UNIT_PE                 = 10,                   // Power Environment (PE)
   DEV_UNIT_POS                = 11,                   // POS
   DEV_UNIT_VIRTUAL            = 12,                   // Virtual unit, which belongs to general
intelligent server
   DEV_UNIT_ROADGATE           = 14,                   // Barrier
   DEV_UNIT_LED                = 15,                   // LED
   DEV_UNIT_DISPATCHER         = 33,                   // Dispatcher
}DPSDK_DEV_UNIT_TYPE;
```

Father theme:structural morphology

DPSDK_ORG_INFO

Organization of basic data

```
Typedef Struct DPSDK_ORG_INFO_T
{
    DPSDK_ORG_BASE_INFO StruOrgBaseInfo; // Organizational information
    DPSDK_INT32 IDevNum;                  // Number of sub devices
    DPSDK_ORG_SUB_DEV_INFO* PDevList;     // Child device list
    DPSDK_INT32 IChannelNum;              // Subchannel number
    DPSDK_ORG_SUB_CHANNEL_INFO* PChannelList; // Subchannel list
    DPSDK_INT32 IOrgNum;                  // Suborganization number
    DPSDK_ORG_INFO_T* POrgList;           // Suborganization list
}DPSDK_ORG_INFO;
```

Father theme:structural morphology

DPSDK_DEV_ALL_INFO_LIST

Device List

```
typedef struct
{
    DPSDK_INT32 iDevNum;                                    // Number of Device
    DPSDK_DEV_ALL_INFO* pDevAllInfoList;                    // Device List Data
}DPSDK_DEV_ALL_INFO_LIST;
```

Father theme:structural morphology

DPSDK_COLLECTION_ORG_INFO

Collectors' information

```
Typedef Struct DPSDK_COLLECTION_ORG_INFO_T
{
    DPSDK_CHAR SzOrgCode[DPSDK_ORG_CODE_LEN]; // organization ID
    DPSDK_CHAR SzOrgName[DPSDK_ORG_NAME_LEN]; // Organization name
    DPSDK_INT32 ISort;                    // Sort value
    DPSDK_BOOL BHasData;                        // Whether there is direct data (non - organized
nodes, can be channels)
    DPSDK_CHAR SzParentCode[DPSDK_ORG_CODE_LEN]; // Parent tissue ID
    DPSDK_INT32 ISubOrgNum;                   // Sub organization number
    DPSDK_COLLECTION_ORG_INFO_T* PSuvOrgList;    // Sub organization list
}DPSDK_COLLECTION_ORG_INFO;
```

Father theme:structural morphology

DPSDK_LAYERED_RESULT_LIST

Gradation gets the device tree to return the result list

```
Typedef Struct
{
     DPSDK_INT32 IResultNum;              // List length
     DPSDK_LAYERED_RESULT* PResultList; // Result list
}DPSDK_LAYERED_RESULT_LIST;
```

Father theme:structural morphology

DPSDK_ALL_ORG_INFO

All organization information (recursion tree)

```
Typedef Struct DPSDK_ALL_ORG_INFO_T
{
    DPSDK_SINGLE_ORG_INFO StruOrgBaseInfo;     // Information at the level of the organization
    DPSDK_INT32 IOrgNum;                       // Sub organization number
    DPSDK_ALL_ORG_INFO_T* POrgList;            // Sub organization list
}DPSDK_ALL_ORG_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_TASK_INFO

TV wall task information

```
Typedef Struct
{
    DPSDK_TVWALL_TASK_BASE_INFO StruBaseInfo;            // TV wall task information
    DPSDK_TVWALL_TASK_SCREEN_OPER_LIST StruScreenOperList; // TV wall task screen operation information list
    DPSDK_TVWALL_TASK_CHANNEL_EXT_LIST StruChannelExtList; // A list of device information required by a video source in the decoding mode of a direct connection.
}DPSDK_TVWALL_TASK_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_TASK_INFO_LIST

TV Wall Task Info

```
typedef struct
{
    DPSDK_INT32 iTVWallTaskNum;                              // Number of TV Wall Task
    DPSDK_TVWALL_TASK_INFO* pTVWallTaskList;                 // List of TV Wall Task
    DPSDK_TVWALL_PROJECT_LIST struProjectList;               // List of TV Wall Plan
}DPSDK_TVWALL_TASK_INFO_LIST;
```

Father theme:structural morphology

DPSDK_ORG_BASE_INFO

Basic Organizational Data

```
typedef struct
{
   DPSDK_CHAR     szOrgCode[DPSDK_ORG_CODE_LEN];          // Organization Code
   DPSDK_CHAR     szOrgName[DPSDK_NAME_LEN];              // Organization Name
   DPSDK_CHAR     szOrgSN[DPSDK_ORG_SN_LEN];              // Organization SN
   DPSDK_INT32 iOrgType;                                  // Organization Node Type
   DPSDK_INT32 iOrgSort;                                  // Organization Sorting
}DPSDK_ORG_BASE_INFO;
```

Father theme:structural morphology

DPSDK_ORG_SUB_DEV_INFO

Device data for organizing trees

```
Typedef Struct
{
    DPSDK_CHAR SzDeviceId[DPSDK_DEVICE_ID_LEN]; // equipment ID
    DPSDK_INT32 ISort; // sort
}DPSDK_ORG_SUB_DEV_INFO;
```

Father theme:structural morphology

DPSDK_ORG_SUB_CHANNEL_INFO

The channel data of the organization tree

```
Typedef Struct
{
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; // passageway ID
    DPSDK_INT32 ISort;                            // sort
}DPSDK_ORG_SUB_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_ORG_INFO_T

Basic Organizational Data

```
typedef struct DPSDK_ORG_INFO_T
{
    DPSDK_ORG_BASE_INFO struOrgBaseInfo;                    // Organization Info
    DPSDK_INT32 iDevNum;                                    // Number of Child Device
    DPSDK_ORG_SUB_DEV_INFO* pDevList;                       // List of Child Device
    DPSDK_INT32 iChannelNum;                                // Number of Sub-channel
    DPSDK_ORG_SUB_CHANNEL_INFO* pChannelList;               // List of Sub-channel
    DPSDK_INT32 iOrgNum;                                    // Number of Sub-organization
    DPSDK_ORG_INFO_T* pOrgList;                             // List of Sub-organization
}DPSDK_ORG_INFO;
```

Father theme:structural morphology

DPSDK_SINGLE_ORG_INFO

Single organization data

```
Typedef Struct DPSDK_SINGLE_ORG_INFO_T
{
     DPSDK_CHAR SzOrgCode[DPSDK_ORG_CODE_LEN];    // organization code
     DPSDK_CHAR SzOrgName[DPSDK_ORG_NAME_LEN];    // Organization name
     DPSDK_CHAR SzOrgSN[DPSDK_SN_LEN];            // organization SN code
     DPSDK_BOOL BHasData;                         // Whether there is direct data
     DPSDK_INT32 IOrgSort;                        // Organization sort
     DPSDK_CHAR SzParentCode[DPSDK_ORG_CODE_LEN]; // Organization parent node ID
}DPSDK_SINGLE_ORG_INFO;
```

Father theme:structural morphology

DPSDK_ALL_ORG_INFO_T

All Organizational Info (Recursion Tree)

```
typedef struct DPSDK_ALL_ORG_INFO_T
{
    DPSDK_SINGLE_ORG_INFO struOrgBaseInfo;                    // Info about Organization at this Level
    DPSDK_INT32 iOrgNum;                                      // Number of Sub-organizations
    DPSDK_ALL_ORG_INFO_T* pOrgList;                           // List of Sub-organizations
}DPSDK_ALL_ORG_INFO;
```

Father theme:structural morphology

DPSDK_NODE_TYPE

Node type

```
Typedef Enum
{
    NODE_TYPE_ORG = 1,      // organization
    NODE_TYPE_DEV = 2,      // equipment
    NODE_TYPE_CHANNEL = 3, // channel
}DPSDK_NODE_TYPE;
```

Father theme:structural morphology

DPSDK_LAYERED_RESULT

Gradation gets the result of the device tree

```
Typedef Struct
{
    DPSDK_INT32  INodeType;                           // See DPSDK_NODE_TYPE Definition 
1:Organization,2:Equipment,3:passageway
    DPSDK_CHAR SzID[DPSDK_ORG_CODE_LEN];   // node ID
    DPSDK_CHAR SzName[DPSDK_ORG_NAME_LEN]; // Node name
    DPSDK_BOOL IsParent;                  // Whether it is a parent node
    DPSDK_CHAR SzParentID[DPSDK_ORG_CODE_LEN]; // Parent node ID
    DPSDK_INT32 ISort;                          // Sort value
    DPSDK_INT32 IStatus;                        // Channel state See DPSDK_DEV_STATUS Definition
    DPSDK_INT32  IType1;  //  INodeTypeFor  equipment,  it  represents  a  large  class  of 
equipment.INodeTypeUnit type for channel time
    DPSDK_INT32 IType2; // INodeTypeA small class of devices is represented when the device 
is used.INodeTypeExpress channel type for channel
    DPSDK_CHAR SzSN[DPSDK_SN_LEN]; // SNCode
    //Equipment information
    DPSDK_CHAR SzIP[DPSDK_IP_LEN]; // equipment IP
    //Channel information
    DPSDK_INT32 IChannelSeq;       // Channel number
    DPSDK_INT32 IDomainID;         // domain ID
}DPSDK_LAYERED_RESULT;
```

Father theme:structural morphology

DPSDK_MEDIA_BASE_PARAM

Basic video parameters

```
Typedef Struct
{
    HCWND PHWnd;                              // Window handle
    //Basic video parameters
    DPSDK_CHAR SzCodeId[DPSDK_DEVICE_ID_LEN]; // passageway ID Or equipment ID
    DPSDK_INT32 IStreamType;                       // Code stream type 1=Main stream, 2=Auxiliary
code stream
    DPSDK_INT32 IDataType;                        // Video type:1=video, 2=audio frequency, 3=Audio
and video
    DPSDK_INT32 IDecodeType;                           // Decode type See DPSDK_DECODE_TYPE
Definition
    DPSDK_INT32 IStreamMode;                      // Playback mode See DPSDK_STREAM_MODE
Definition
    DPSDK_UINT32 UiDelayTime; // Play delay time, when IstreamMode is
 DPSDK_STREAM_CUSTOM_MODETime, it is effective Company MS
}DPSDK_MEDIA_BASE_PARAM;
```

Father theme:structural morphology

DPSDK_MEDIA_CALLBACK

Video callback structure

```
Typedef Struct
{
    DPSDK_REALDATA_CALLBACK FRealDataCallBack; // Bitstream callback
    DPSDK_LPVOID PRealUserData;                // Code stream callback user data
    DPSDK_FISHEYE_CALLBACK FFishEyeCallBack;   // Fish eye data callback
    DPSDK_LPVOID PFishEyeUserData;             // Fish eye data callback user data
    DPSDK_DRAW_CALLBACK FDrawCallBack;         // Video plotting callback
    DPSDK_LPVOID PDrawUserData;                // Video plotting callback user data
    DPSDK_DEMUXDEC_CALLBACK FDemuxDecCallBack; // Data callback for the analysis
of source data
    DPSDK_LPVOID PDemuxDecUserData;            // Data callback to user data analyzed by
source data
    DPSDK_EVENT_CALLBACK FEventCallBack;       // event callbacks
    DPSDK_LPVOID PEventUserData;               // Event callback user data
    DPSDK_TVWALL_PLAYBACK_CALLBACK   FTVWallPlaybackCallBack;   //Replay   the
back wall callback
    DPSDK_LPVOID PTVWallPlaybackUserData;      // Playback the upper wall callback
user data
}DPSDK_MEDIA_CALLBACK;
```

Father theme:structural morphology

DPSDK_RECORD_STATUS

Video status

```
Typedef Enum
{
    RECORD_STATUS_IDLE = 0,       // Video does not make it possible
    RECORD_STATUS_NORMAL = 1,     // In the ordinary video
    RECORD_STATUS_EXCEPTION = 2, // abnormal
    RECORD_STATUS_MANUAL = 3,     // Manual video is being triggered
}DPSDK_RECORD_STATUS;
```

Father theme:structural morphology

DPSDK_STREAM_TYPE

```
Typedef Enum
{
    STREAM_UNKNOW_STREAM = 0,   // Unknown
    STREAM_MAIN_STREAM = 1,         // Main stream
    STREAM_SUB_STREAM = 2,          // Auxiliary code stream
    STREAM_THIRD_STREAM = 3,        // Three bit stream
    STREAM_LOCAL_SIGNAL_STREAM = 5, // Local signal
}DPSDK_STREAM_TYPE;
```

Father theme:structural morphology

DPSDK_SOURCE_TYPE

Video source

```
Typedef Enum
{
    DPSDK_SOURCE_TYPE_ALL = 1,    //All video, including platform video and device video
    DPSDK_SOURCE_TYPE_DEVICE = 2, //Device video
    DPSDK_SOURCE_TYPE_CENTER = 3, //Platform video
}DPSDK_SOURCE_TYPE;
```

Father theme:structural morphology

DPSDK_RECORD_TYPE

Video type

```
Typedef Enum
{
    DPSDK_RECORD_TYPE_ALL = 0,
    DPSDK_RECORD_TYPE_MANUAL = 1,        // Manual video
    DPSDK_RECORD_TYPE_ALARM = 2,         // Alarm video
    DPSDK_RECORD_TYPE_MOTION_DETECT = 3, // Dynamic detection
    DPSDK_RECORD_TYPE_VIDEO_LOST = 4,    // Video loss
    DPSDK_RECORD_TYPE_VIDEO_SHELTER = 5, // Video occlusion
    DPSDK_RECORD_TYPE_TIMER = 6,         // Timing video
    DPSDK_RECORD_TYPE_ALLDAY = 7,        // All-weather video
    DPSDK_RECORD_TYPE_FILE_RECORD = 8,  // File video conversion
    DPSDK_RECORD_TYPE_NORMAL = 9,        // Ordinary video
    DPSDK_RECORD_TYPE_CARD = 25,            // Card number video There is no this in the
protocol library for the time being
    DPSDK_RECORD_TYPE_ALARM_BEGIN = 10,    //Alarm start Definition in the match
protocol stack 10~300 -m -f -cSpecial alarm
    DPSDK_RECORD_TYPE_ALARM_END = 1000,   //End of intelligent alarm Definition in
the match protocol stack 300~1000Intelligent alarm
}DPSDK_RECORD_TYPE;
```

Father theme:structural morphology

DPSDK_SINGLE_RECORD_INFO

Single video recording information

```
Typedef Struct
{
    DPSDK_SOURCE_TYPE ISourceType; // Video source
    DPSDK_RECORD_TYPE IRecordType; // Video type. See RecordType_e
    DPSDK_TIMET IStartTime;        // Start time
    DPSDK_TIMET IEndTime;          // End time
    DPSDK_CHAR SzName[DPSDK_RECORD_FILE_NAME_LEN]; // The name of the video
(different manufacturers are different in the identification of the documents)
    DPSDK_INT64 ILength;           // File length, unit KB
    DPSDK_STREAM_TYPE IStreamType; // Code stream type
    // Here's the information needed for the center video
    DPSDK_INT64 IPlanId;           // Video program ID
    DPSDK_INT32 ISSId;             // Storage service ID
    DPSDK_CHAR SzDiskId[DPSDK_DISDK_ID_LEN];          // disk ID
    DPSDK_INT32 IFileHandle;                          // File handle
    DPSDK_CHAR SzChannelCode[DPSDK_CHANNEL_ID_LEN];       // Channel coding
    DPSDK_BOOL BRecordHidden;                         // Video hiding state True: concealment
; False So
    DPSDK_BOOL BForgotten;                            // Do you forget to forget the video
    // Information added to the alarm video
    DPSDK_CHAR    SzAlarmChannelId[DPSDK_ALARM_CHANNEL_ID_LEN];    //    Video
camera ID
    DPSDK_BOOL BLocked;                               // Whether or not to be locked,Device video
will not be locked
}DPSDK_SINGLE_RECORD_INFO;
```

Father theme:structural morphology

DPSDK_PLAY_DIRECTION

Playback direction

```
Typedef Enum
{
    DPSDK_FORWARD_DIRECTION = 0, // Following discharge
    DPSDK_BACK_DIRECTION = 1,    // Upside down
}DPSDK_PLAY_DIRECTION;
```

Father theme:structural morphology

DPSDK_PLAYBACK_SPEED

Video Playing Speed

```
typedef enum
{
    DPSDK_PB_NORMAL          = 1024,
    DPSDK_PB_NORMAL_FAST2    = DPSDK_PB_NORMAL * 2,
    DPSDK_PB_NORMAL_FAST4    = DPSDK_PB_NORMAL * 4,
    DPSDK_PB_NORMAL_FAST8    = DPSDK_PB_NORMAL * 8,
    DPSDK_PB_NORMAL_FAST16 = DPSDK_PB_NORMAL * 16,
    DPSDK_PB_NORMAL_SLOW2    = DPSDK_PB_NORMAL / 2,
    DPSDK_PB_NORMAL_SLOW4    = DPSDK_PB_NORMAL / 4,
    DPSDK_PB_NORMAL_SLOW8    = DPSDK_PB_NORMAL / 8,
    DPSDK_PB_NORMAL_SLOW16 = DPSDK_PB_NORMAL / 16,
}DPSDK_PLAYBACK_SPEED;
```

Father theme:structural morphology

DPSDK_EVENT_DOWNLOAD_CALLBACK

Video event callback function

```
Typedef DPSDK_VOID (DPSDK_CALL* DPSDK_EVENT_DOWNLOAD_CALLBACK) (
    DPSDK_INT32 IEventType,
    DPSDK_INT32 IMediaSessionID,
    DPSDK_VOID* PData,
    DPSDK_VOID* PUserParam
);
```

Father theme:structural morphology

DPSDK_RECORD_FILE_NAME_RULE

File naming rules

```
Typedef Enum
{
    DPSDK_NAME_RULE_TIME_CHANNELID = 0,
    DPSDK_NAME_RULE_TIME_CHANNELNAME = 1,
    DPSDK_NAME_RULE_CHANNELID_TIME = 2,
    DPSDK_NAME_RULE_CHANNELNAME_TIME = 3
}DPSDK_RECORD_FILE_NAME_RULE;
```

Father theme:structural morphology

DPSDK_DOWNLOAD_RECORD_FILE_FORMAT

file format

```
Typedef Enum
{
    DPSDK_FILE_FORMAT_NORMAL = 0,//Original stream
    DPSDK_FILE_FORMAT_AVI = 1,   //avi format
    DPSDK_FILE_FORMAT_MP4 = 2,   //mp4 format
    DPSDK_FILE_FORMAT_FLV = 3,   //flv format
    DPSDK_FILE_FORMAT_ASF = 4,   //asf format
}DPSDK_DOWNLOAD_RECORD_FILE_FORMAT;
```

Father theme:structural morphology

DPSDK_FILE_STORE_INFO

Video file path

```
Typedef Struct
{
    DPSDK_UINT32  UiStoreLen;                    // Video length
    DPSDK_TIMET LBeginTime;                       // Video start time
    DPSDK_TIMET LEndTime;                        // Video end time
    DPSDK_CHAR SzFile[DPSDK_FILE_PATH_LEN];   // Full path of video files
}DPSDK_FILE_STORE_INFO;
```

Father theme:structural morphology

DPSDK_FISH_SIZE

```
Typedef Struct
{
    DPSDK_INT32 IWidth;
    DPSDK_INT32 IHeight;
}DPSDK_FISH_SIZE;
```

Father theme:structural morphology

DPSDK_FISH_MOUNTMODE

Fish eye installation mode

```
Typedef Enum
{
    DPSDK_EMOUNT_MODE_INVALID = 0,
    DPSDK_EMOUNT_MODE_CEIL = 1, // Top loading
    DPSDK_EMOUNT_MODE_WALL = 2, // Wall mounted
    DPSDK_EMOUNT_MODE_FLOOR = 3, // .
    DPSDK_EMOUNT_MODE_NUM
}DPSDK_FISH_MOUNTMODE;
```

Father theme:structural morphology

DPSDK_FISH_SHOWMODES

Fish eye image display model

```
Typedef Enum
{
    DPSDK_SHOW_MODE_INVALID = 0,
    DPSDK_SHOW_MODE_OFF = 1,        // Shut off the fish eye algorithm library and shut off the
outside
    DPSDK_SHOW_MODE_ORIGINAL = 2, // Primitive pattern(Square),Zoom ratio
    DPSDK_SHOW_MODE_PANORAMA = 3, // 1P
    DPSDK_SHOW_MODE_PANORAMA_PLUS_ONE_EPTZ = 4,          // 1p+1
    DPSDK_SHOW_MODE_DOUBLE_PANORAMA = 5,                 // 2P
    DPSDK_SHOW_MODE_ORIGINAL_PLUS_DOUBLE_PANORAMA = 6,   // 1+2p
    DPSDK_SHOW_MODE_ORIGINAL_PLUS_THREE_EPTZ_REGION = 7, // 1+3
    DPSDK_SHOW_MODE_PANORAMA_PLUS_THREE_EPTZ_REGION = 8, // 1p+3
    DPSDK_SHOW_MODE_ORIGINAL_PLUS_TWO_EPTZ_REGION = 9,   // 1+2
    DPSDK_SHOW_MODE_ORIGINAL_PLUS_FOUR_EPTZ_REGION = 10, // 1+4
    DPSDK_SHOW_MODE_PANORAMA_PLUS_FOUR_EPTZ_REGION = 11, // 1p+4
    DPSDK_SHOW_MODE_PANORAMA_PLUS_SIX_EPTZ_REGION = 12, // 1p+6
    DPSDK_SHOW_MODE_ORIGINAL_PLUS_EIGHT_EPTZ_REGION = 13, // 1+8
    DPSDK_SHOW_MODE_PANORAMA_PLUS_EIGHT_EPTZ_REGION = 14, // 1p+8
    DPSDK_SHOW_MODE_TWO_EPTZ_REGION_WITH_ORIGINAL = 15,   // 1F+2
    DPSDK_SHOW_MODE_FOUR_EPTZ_REGION_WITH_ORIGINAL = 16, // 1F+4
    DPSDK_SHOW_MODE_DOUBLE_PANORAMA_WITH_ORIGINAL = 17,   // 1F+2p
    DPSDK_SHOW_MODE_FOUR_EPTZ_REGION_WITH_PANORAMA = 18, // 1p (F) +4
    DPSDK_SHOW_MODE_TWO_EPTZ_REGION = 19,                 // 2Frame
    DPSDK_SHOW_MODE_SINGLE = 20,                          // Single picture
    DPSDK_SHOW_MODE_FOUR_EPTZ_REGION                            = 21, // 4Frame
    DPSDK_SHOW_MODE_USER_DEFINED = 22,                          // User custom
    DPSDK_FISHEYE_CALIBRATE_MODE_ORIGINAL_PLUS_ONE_EPTZ_REGION = 23 //
1+1
    DPSDK_FISHEYE_CALIBRATE_MODE_ONE_EPTZ_REGION = 24,             // 1Frame
    DPSDK_SHOW_MODE_NUM
}DPSDK_FISH_SHOWMODES;
```

Father theme:structural morphology

DPSDK_FISH_MODEINITPARAM

```
Typedef Struct
{
     DPSDK_FISH_REGIONPARAM ArrstruRegionParam[9]; // The sequence number of an array corresponds to a window IDNumber, incoming configuration needs matching ID
     DPSDK_INT32 ICircularOffset;
     DPSDK_INT32 IPanoramaOffset;
     DPSDK_INT32 IUseRegionParam; // When it is valid, use this value to initialize it; if you don't save the information, please set it 0
     DPSDK_INT32 ArriReserved[1];
 }DPSDK_FISH_MODEINITPARAM;
```

Father theme:structural morphology

DPSDK_FISH_OUTPUTFORMAT

```
Typedef Struct
{
    DPSDK_FISH_SIZE StruMainShowSize; // Not enabled for the time being,  Main display ratio, 4:3, 16:9等, The optimal results are output based on the algorithm (no deformation case)Try to reach the ratio as far as possible)
    DPSDK_FISH_SIZE StruImgOutput; // Output image resolution(Pre scale), The image master correction mode is external input when the user custom mode is used, Other modes Internal return
    DPSDK_FISH_SIZE StruFloatMainShowSize; // Output double BufferWhen used, the main display window of the floating window is resolved for the time being with the old method of operation.The width to height ratio of the floating circle needs to be 1:1The width to height ratio  of the floating Wall panorama needs to be 16:9
    DPSDK_FISH_SUBMODE StruSubMode; // Subpattern information, The image master correction mode is external input when the user custom mode is used, Other patterns are internally returned
    DPSDK_INT32 ISubModeNum;      // Subpattern number, The image master correction mode is external input when the user custom mode is used, Other patterns are internally returned
    DPSDK_INT32 IOutputSizeRatio; // Not enabled for the time being, The scaling ratio of the corrected output imageQ8.format,Range 0-256, Two  hundred and fifty-sixTo maintain the maximum output resolution
    DPSDK_INT32 ArriReserved[1]; // Reserved bytes
}DPSDK_FISH_OUTPUTFORMAT;
```

Father theme:structural morphology

DPSDK_MHFPTZ_CONFIGPARAM

```
Typedef Struct
{
    /**Necessary Parameters* /
    DPSDK_INT32 IZoomType; // Multiple Control Mode---- Expected Adaptive And Double The Size Of The Box.
    DPSDK_INT32 IHCamWax; // The Expectation Multiplier Corresponds To The Ball Anglex (horizontal)
    DPSDK_INT32 IHCamWay; // The Expectation Multiplier Corresponds To The Ball Angley (vertical)
    DPSDK_INT32 IHCamWMul; // Expected Multiplier (benchmark Multiplier)
    DPSDK_INT32 ICfgType; // Configuration Method, Default Is1: Using Configuration Parameters 1:.using The Method Of Parameter Configuration, 0Use Device Type Collocation Method
    // Main Camera Parameters
    // Lens Parameters
    DPSDK_INT32 IPrmRE; // Projection Radius
    DPSDK_INT32 IPrmMul; // Projection Ratio
    DPSDK_INT32 IPrmDX; // XDirection Shift
    DPSDK_INT32 IPrmDY;  // YDirection Shift
    DPSDK_INT32 IPrmCW; // CMOSWide (practical Use)
    DPSDK_INT32 IPrmCH; // CMOSHigh (high Practical Use)
    // Main Camera And Slave Type Configuration (Cfg_type is 0, it is valid to set this parameter at the time/ / A) DefaultOne Hundred And ThirtyDegree, One Hundred And Thirty000 And Bolt200W65Speed Dome Cameras
    DPSDK_UINT32 IMLenType; // Main Camera Lens Type, See DPSDK_LEN_TYPE
    DPSDK_UINT32 IMCamType; // Main Camera Type, See DPSDK_CAM_TYPE
    DPSDK_UINT32 IHCamType; // From Camera Type, See DPSDK_CAM_TYPE
    // Ball Machine Parameters
    DPSDK_INT32 IHImgWidth; // From The Camera Image Width
    DPSDK_INT32 IHImgHeight; // From The Camera Image
    DPSDK_INT32 IPrmFax;
    /*Default Parameters* /
    //Main Camera Parameters
    DPSDK_INT32 IMCamFC;    // Equivalent Focal Length Of Camera
    DPSDK_INT32 IMCamCW;    // Lens  Target Height
    DPSDK_INT32 IMCamCH;    // Wide Shot Target
    DPSDK_INT32 ICamHeight; // Camera Height (meter), (temporarily Unused)
    DPSDK_INT32 IPrmMA;     // Focal Length
    // From Camera Parameters
    // Ball Machine Parameters
    DPSDK_INT32 IPrmHW;     // CMOS Width
    DPSDK_INT32 IPrmHH;     // CMOS Height
    DPSDK_INT32 IPrmFO;     // Equivalent Focal Length
    DPSDK_INT32 IPrmCA;     // Field Of View Parameter
    DPSDK_INT32 IPrmMMul; // Maximum Ratio
}DPSDK_MHFPTZ_CONFIGPARAM;
```

Father theme:structural morphology

DPSDK_FISH_EPTZCMD

An option for an electronic cloud platform to bloom and move

```
Typedef Enum
{
    DPSDK_EPTZ_CMD_INVALID = 0,
    DPSDK_EPTZ_CMD_ZOOM_IN = 1,              // enlarge
    DPSDK_EPTZ_CMD_ZOOM_OUT = 2,             // narrow
    DPSDK_EPTZ_CMD_UP = 3,                   // Move upwards
    DPSDK_EPTZ_CMD_DOWN = 4,                 // Move down
    DPSDK_EPTZ_CMD_LEFT = 5,                 // left
    DPSDK_EPTZ_CMD_RIGHT = 6,                // Move to the right
    DPSDK_EPTZ_CMD_ROTATE_CLOCKWISE_AUTO = 7,     // Clockwise rotation
    DPSDK_EPTZ_CMD_ROTATE_ANTICLOCKWISE_AUTO = 8, // Automatic counter
clockwise rotation
    DPSDK_EPTZ_CMD_STOP = 9,                 // Stop it
    DPSDK_EPTZ_CMD_SHOW_REGION = 10,         // Frame selection and enlargement
    DPSDK_EPTZ_CMD_EXIT_SHOW_REGION = 11,            // Exit frame selection and
enlargement
    DPSDK_EPTZ_CMD_DEFAULT = 12,             // Restore the default
    DPSDK_EPTZ_CMD_ORIGIN_ROTATE = 13,       // Circular rotation
    DPSDK_EPTZ_CMD_SET_CUR_REGION = 0x20,            // Configure the location
information of the specified window
    DPSDK_EPTZ_CMD_GET_CUR_REGION = 0x21,        // Get the location information of
the specified window
    DPSDK_EPTZ_CMD_IS_IN_PANORAMA_REGION = 0x22, // Whether the input point is in
the current panoramic point chain area
    DPSDK_EPTZ_CMD_TAP_VIEW = 0x23,              // Display the specified location,The
point is to see
    DPSDK_EPTZ_CMD_SET_FOCUS = 0x24,         // Setting window location information
    DPSDK_EPTZ_CMD_GET_FOCUS = 0x25,         // Get the window location information
    DPSDK_EPTZ_CMD_PTZ_CALI = 0x26,          // Calibration of fish ball linkage
    DPSDK_EPTZ_CMD_GET_PTZ_RLT= 0x27,            // Acquisition of fish ball linkage
location information
    DPSDK_EPTZ_CMD_NUM
} DPSDK_FISH_EPTZCMD;
```

Father theme:structural morphology

DPSDK_SUBORDINATE_CAMCONFIGPARAM

External configuration from camera parameters(Fish ball linkage)

```
Typedef Struct
{
    DPSDK_INT32 IHCamWax;    // The expectation multiplier corresponds to the dome  angle 
x(horizontal)
    DPSDK_INT32 IHCamWay;   // The expectation multiplier corresponds to the dome angle y 
(vertical)
    DPSDK_INT32 IHCamWMul;   // Expectation multiplier (benchmark multiplier)
    DPSDK_UINT32 UiHCamType; // From camera type,See DPSDK_CAM_TYPE
}DPSDK_SUBORDINATE_CAMCONFIGPARAM;
```

Father theme:structural morphology

DPSDK_IVSE_FUNC_TYPE

Enhanced support for functional enumeration

```
Typedef Enum
{
    DPSDK_IVSE_DEHAZE = 0,    // Go to the fog
    DPSDK_IVSE_DENOISE = 1, // Denoising
    DPSDK_IVSE_WB = 2,        // Color correction
    DPSDK_IVSE_LOWLIGHT = 3, // Low illumination enhancement
    DPSDK_IVSE_HDR = 4,       // Wide dynamic
    DPSDK_IVSE_NUM = 5        // Support enhanced number of functions
}DPSDK_IVSE_FUNC_TYPE;
```

Father theme:structural morphology

DPSDK_IVSE_ROI

ROIData type definition

```
Typedef Struct
{
    DPSDK_INT32 IX;       // Top left corner xcoordinate
    DPSDK_INT32 IY;      // Top left corner ycoordinate
    DPSDK_INT32 IWidth; // Regional width
    DPSDK_INT32 IHeight; // Regional height
}DPSDK_IVSE_ROI;
```

Father theme:structural morphology

AlarmDealWith_e

Alarm processing state

```
Typedef Enum
{
    DEALWITH_PENDING     = 1,        //In the process of processing
    DEALWITH_RESOLVE     = 2,        //resolved
    DEALWITH_SUGGESTTED = 3,          //False positives
    DEALWITH_IGNORED     = 4,        //ignore
    DEALWITH_UNPROCESSED = 5,         //Unsolved
}AlarmDealWith_e;
```

Father theme:structural morphology

DPSDK_EMAILADDRESS

E-mail address

```
Typedef Struct
{
    DPSDK_CHAR    SzEmailAddr[DPSDK_ALARM_EMAILRECEIVER_LEN];    //    E-mail
address
}DPSDK_EMAILADDRESS;
```

Father theme:structural morphology

DPSDK_ALARM_DETAILINFO

Alarm information

```
Typedef Struct
{
    DPSDK_CHAR SzAlarmId[DPSDK_ALARM_ALARMID_LEN];                    // Call police ID
    DPSDK_CHAR SzDeviceId[DPSDK_ALARM_DEVICEID_LEN];                  // equipment ID
    DPSDK_CHAR SzDeviceName[DPSDK_ALARM_DEVICENAME_LEN];              // Device name
    DPSDK_CHAR SzChannelId[DPSDK_ALARM_CHANNELID_LEN];                // channel ID
    DPSDK_CHAR SzChannelName[DPSDK_ALARM_CHANNELNAME_LEN];            // Channel name
    DPSDK_INT32 IAlarmGrade;                  //Alarm level(Reference resources AlarmLevel_e)
    DPSDK_INT32 IAlarmType;                   // Alarm type(Reference resources Alarm_type_e)
    DPSDK_INT32 IAlarmStatus;                 // Alarm state(Reference resources AlarmState_e)
    DPSDK_CHAR SzHandleUser[DPSDK_ALARM_HANDLERUSER_LEN];             // Alarm processing person
    DPSDK_CHAR SzHandleTime[DPSDK_ALARM_TIME_LEN];          // Alarm processing time Yyyymmddhhmmss
    DPSDK_INT32 IHandleStatus;                // Alarm processing state(Reference resources AlarmDealWith_e)
    DPSDK_CHAR SzHandleMessage[DPSDK_ALARM_HANDLEMESSAGE_LEN];        // Handling opinions
    DPSDK_CHAR SzAlarmCode[DPSDK_ALARM_ALARMCODE_LEN];      // Alarm code
    DPSDK_CHAR SzAlarmTime[DPSDK_ALARM_TIME_LEN];           // Alarm time Yyyymmddhhmmss
    DPSDK_CHAR SzAlarmPicture[DPSDK_ALARM_ALARMPICTURE_LEN];          // Alarm snapshot path
    DPSDK_UINT32 UiAlarmPictureSize;                  // Alarm snapshot size
    DPSDK_UINT32 UiEmailReceiverListSize; // The actual number of notification mailbox lists is not greater than that of the alarm
    DPSDK_EMAILRECEIVERLIST_SIZE）
    DPSDK_EMAILADDRESS
StruEmailReceiverList[DPSDK_ALARM_EMAILRECEIVERLIST_SIZE]; // Alarm processing notification mailbox list(Most return DPSDK_EMAILRECEIVERLIST_SIZEA mail address)
}DPSDK_ALARM_DETAILINFO;
```

Father theme:structural morphology

DPSDK_ALARMPROCESS_DETAILINFO

Alarm processing record

```
Typedef Struct
{
    DPSDK_CHAR SzHandleUser[DPSDK_ALARM_HANDLERUSER_LEN];              // Alarm processing person
    DPSDK_CHAR SzHandleTime[DPSDK_ALARM_TIME_LEN];              // Alarm processing time Yyyymmddhhmmss
    DPSDK_CHAR       SzHandleMessage[DPSDK_ALARM_HANDLEMESSAGE_LEN];       // Warning handling opinion
    DPSDK_INT32  IHandleStatus;                                   // Alarm processing state(Reference resources AlarmDealWith_e)
}DPSDK_ALARMPROCESS_DETAILINFO;
```

Father theme:structural morphology

DPSDK_PTZ_LOCKUSER

Lock holder information

```
Typedef Struct
{
    DPSDK_CHAR SzLockUserName[DPSDK_NAME_LEN]; // User name for lockin cloud
    DPSDK_INT32 ILockUserLevel;                // Lock user level of cloud
}DPSDK_PTZ_LOCKUSER;
```

Father theme:structural morphology

DPSDK_TVWALL_BASE_INFO

The basic information of the TV wall

```
Typedef Struct
{
    DPSDK_INT32 ITVWallId;                          // TV wall ID
    DPSDK_CHAR SzTVWallName[DPSDK_NAME_LEN];      // TV wall name
    DPSDK_CHAR SzOrgCode[DPSDK_ORG_CODE_LEN];     // TV issue coding
    DPSDK_INT32 IState;                           // Enabled state:0=Disable,1=Enable
    DPSDK_UINT32 UiVersion;                         // TV wall version number
    DPSDK_UINT32 UiSustainTime;                       // Duration(秒)
    DPSDK_CHAR SzTVWallDesc[DPSDK_MEMO_LEN];      // describe
}DPSDK_TVWALL_BASE_INFO;
```

Father theme:structural morphology

DPSDK_SCREEN_DECODER_LIST

A list of decoded channels for screen binding

```
Typedef Struct
{
    DPSDK_INT32 IScreenDecoderNum;                          // Decode channel number
    DPSDK_SCREEN_DECODER_INFO
StruScreenDecoder[DPSDK_MAX_SCREEN_DECODER_NUM]; //Decode channel list
}DPSDK_SCREEN_DECODER_LIST;
```

Father theme:structural morphology

DPSDK_TVWALL_TASK_BASE_INFO

TV wall task information

```
Typedef Struct
{
    DPSDK_INT32 ITVWallId;                  // TV wall ID
    DPSDK_INT32 ITaskId;                    // task ID
    DPSDK_CHAR SzTaskName[DPSDK_NAME_LEN]; // Task name
    DPSDK_CHAR SzTaskDesc[DPSDK_MEMO_LEN]; // Task description
}DPSDK_TVWALL_TASK_BASE_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_TASK_SCREEN_OPER_LIST

TV wall task screen operation information list

```
Typedef Struct
{
    DPSDK_UINT32 UiTotal;                    // The number of operation information of the TV wall task screen
    DPSDK_TVWALL_TASK_SCREEN_OPER_INFO* PScreenOperList; // TV wall task screen operation information list
}DPSDK_TVWALL_TASK_SCREEN_OPER_LIST;
```

Father theme:structural morphology

DPSDK_TVWALL_TASK_CHANNEL_EXT_LIST

A list of equipment information required by a video source in the decoding mode of a direct connection

```
Typedef Struct
{
     DPSDK_UINT32 UiTotal; // The number of device information required by a video source in the decoding mode of a direct connection.
     DPSDK_TVWALL_TASK_CHANNEL_EXT_INFO*  PChannelExtList; // A list of device information required by a video source in the decoding mode of a direct connection.
}DPSDK_TVWALL_TASK_CHANNEL_EXT_LIST;
```

Father theme:structural morphology

DPSDK_CURRENT_TVWALL_TASK_INFO

TV wall task information currently being executed

```
Typedef Struct
{
     DPSDK_INT32 ITVWallId;                    // TV wall ID
     DPSDK_INT32 ITaskId;                      // task ID
     DPSDK_CHAR SzTaskName[DPSDK_NAME_LEN]; // Task name
     DPSDK_INT32 IPlanId;                      // plan ID
     DPSDK_CHAR SzPlanName[DPSDK_NAME_LEN]; // Plan name
     DPSDK_INT32 IDataType;                    // Data types:0=The task,1=plan
}DPSDK_CURRENT_TVWALL_TASK_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_WINDOW_INFO

Window information

```
Typedef Struct
{
    DPSDK_TVWALL_SCREEN_POS          StruWndRect;          //tvwall_control_e          as
 TVWALL_WINDOW_OPEN, TVWALL_WINDOW_MOVINGTime effective
    DPSDK_INT32 IWindowId;             //window ID
    DPSDK_INT32   ISpliteNum;                                //Partition  number,Tvwall_control_e 为
TVWALL_SCREEN_SPLITTime effective
    DPSDK_INT32 IOrder;                 //window Zorder
}DPSDK_TVWALL_WINDOW_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_CONTROL_TYPE

TV wall control command

```
Typedef Enum
{
    TVWALL_PLAN_TASK = 0,          // Task wall/Task switching
    TVWALL_ONESCREEN_SHOW = 1,      // Binding video source
    TVWALL_ONESCREEN_CLOSE = 2,     // Cancelling the video source
    TVWALL_SCREEN_SPLIT = 3,        // Picture segmentation
    TVWALL_WINDOW_OPEN = 4,         // window
    TVWALL_WINDOW_CLOSE = 5,        // close the window
    TVWALL_WINDOW_MOVING = 6,       // window moving
    TVWALL_WINDOW_ZCONTROL = 7,     // Top
    TVWALL_POWER_CTRL = 8,          // Screen switch
    TVWALL_ONESCREEN_CLOSE_ALL  = 9, // SmartpssUse:  close a single  screen(All
windows on a single screen)
    TVWALL_CLOSE_PROJECT,           // Closure plan
    TVWALL_ONESCREEN_CLEAR  =  12,       // Clean up a single screen,It only needs to be
corresponding at this time_tvIndex, _screenId, _tvTypeas well as_tvWallDBId
    TVWALL_SCREEN_ADDFRAME  =  13,  //  The  window  highlights,  it  only  needs  to  be
corresponding at this time_tvIndex, _screenId, _subTvIndex, _tvTypeas well as _tvWallDBIdThis
is  used  here.  SplitNumIndicating  high  brightness  (1)  or  not  high  (0);  if  needed  RGBAColor
information reusable Position
    TVWALL_SPLITWIN_MAX = 14,       // Split screen magnification
    TVWALL_TOUR_PAUSE  =  15,                // The upper wall passage wheel pause, at this
time_screenIdif it is-1It is effective for the whole wall._subTvIndex -1To the whole Screen is
valid (but_screenIdIt must be a valid value)
    TVWALL_TOUR_RESUME  =  16,           // The upper wall channel wheel restores, the same
Fifteen
    TVWALL_SINGLEWINDOW_CHANGE_SOURCE = 17, // The single window switches the
video source to the previous upper wall._screenId, _subTvIndexAll must be effective
    TVWALL_SINGLEWINDOW_SOUND_SWITCH  =  18,  //  Single   window  audio  switch
control, It only needs to be corresponding at this time_tvIndex, _screenId, _subTvIndex, _tv Type
as well as_tvWallDBIdThis is used here. SplitNumPresentation is open (1) or shut down (0)
    TVWALL_OPENWINDOW_SPLIT = 22,        // Split window partition
}DPSDK_TVWALL_CONTROL_TYPE;
```

Father theme:structural morphology

DPSDK_TVWALL_SCREEN_POS

Screen position of TV wall

```
Typedef Struct
{
     DPSDK_FLOAT FLeft;    // The distance to the left of the screen, the percentage
     DPSDK_FLOAT FTop;     // The margin on the screen, the percentage
     DPSDK_FLOAT FWidth;   // Screen width, percentage
     DPSDK_FLOAT FHeight;  // Screen height, percentage
}DPSDK_TVWALL_SCREEN_POS;
```

Father theme:structural morphology

DPSDK_TVWALL_PROJECT_LIST

TV wall plan list

```
Typedef Struct
{
     DPSDK_UINT32 UiCount;                // Planned number
     DPSDK_TVWALL_PROJECT_INFO* PProjList; // Plan list
}DPSDK_TVWALL_PROJECT_LIST;
```

Father theme:structural morphology

DPSDK_SUB_CODE_TYPE

Organization query node type

```
Typedef Enum
{
    SUB_SIGNAL_CODE, // Primary subnode
    SUB_ALL_CODE,     // All subnodes
}DPSDK_SUB_CODE_TYPE;
```

Father theme:structural morphology

DPSDK_DEV_UNIT_TYPE

Unit type

```
Typedef Enum
{
    DEV_UNIT_UNKOWN,            // Unknown
    DEV_UNIT_ENC,               // Code
    DEV_UNIT_DEC,               // Decode
    DEV_UNIT_ALARMIN,           // Alarm input
    DEV_UNIT_ALARMOUT,          // Alarm output
    DEV_UNIT_TVWALLIN,          // Tv Wall input
    DEV_UNIT_TVWALLOUT,         // Tv Wall output
    DEV_UNIT_DOORCTRL,          // Access control
    DEV_UNIT_VOICE,             // Intercom
    DEV_UNIT_PE = 10,           // Ring PE,Originally called PE=>power Environment (Dynamic
environment)
    DEV_UNIT_POS = 11,          // POS
    DEV_UNIT_VIRTUAL = 12,      // Virtual unit All smart servers
    DEV_UNIT_ROADGATE = 14,     // Road gate
    DEV_UNIT_LED = 15,          // LED
    DEV_UNIT_DISPATCHER = 33,   // Dispatcher
}DPSDK_DEV_UNIT_TYPE;
```

Father theme:structural morphology

DPSDK_DEV_INFO

Basic equipment information

```
Typedef Struct
{
     DPSDK_CHAR SzDeviceID[DPSDK_DEVICE_ID_LEN]; // equipment ID
     DPSDK_CHAR SzDeviceName[DPSDK_NAME_LEN];    // Device name
     DPSDK_CHAR SzUserName[DPSDK_NAME_LEN];      // Device logon user
     DPSDK_CHAR SzUserPwd[DPSDK_PWD_LEN];        // Device login password
     DPSDK_CHAR SzIP[DPSDK_IP_LEN];              // Device additionIP
     DPSDK_USHORT UshPort;                       // Device add port
     DPSDK_CHAR SzDevIP[DPSDK_IP_LEN];           // True equipmentIP
     DPSDK_USHORT UshDevPort;                    // Device real port
     DPSDK_INT32 IManFac;                        // Manufacturer
     DPSDK_INT32  IStatus;                               // Equipment state See DPSDK_DEV_STATUS
Definition
     DPSDK_INT32  IDevType;                              // Equipment type See DPSDK_DEV_TYPE
Definition
}DPSDK_DEV_INFO;
```

Father theme:structural morphology

DPSDK_ENC_CHANNEL_INFO

Coded channel information

```
Typedef Struct
{
    DPSDK_BASE_CHANNEL_INFO StruChannelInfo;
    DPSDK_INT32  ICameraType;                    // Camera type See DPSDK_CAMERA_TYPE Definition
    DPSDK_CHAR SzLatitude[DPSDK_GPS_LEN];   // latitude
    DPSDK_CHAR SzLongitude[DPSDK_GPS_LEN]; //  longitude
    DPSDK_INT32  ICameraFunction;              // 0No support function 1Support fish eye 2Support electric focusing
    DPSDK_CHAR SzMulticastIP[DPSDK_IP_LEN]; // Multicast IP
    DPSDK_USHORT UshMulticastPort;           // Multicast port
    DPSDK_CHAR SzNVR_IPCIP[DPSDK_IP_LEN];   // NVRFront endIPC IP
    DPSDK_INT32   IChannelRemoteType;                         //   Remote   channel   type   See DPSDK_CHANNEL_REMOTE_TYPE Definition
    // The type of unit in which the channel belongs
    DPSDK_INT32 ITrackID;                  // Flow type
    DPSDK_INT32  IStreamType;                 // Code stream type See DPSDK_STREAM_TYPE Definition
    DPSDK_BOOL BZeroEncode;                 // Does it support0Channel multi picture coding
}DPSDK_ENC_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_DEC_CHANNEL_INFO

Decoding channel information

```
Typedef Struct
{
    DPSDK_BASE_CHANNEL_INFO StruChannelInfo;
    DPSDK_INT32 IMaxSpliteNum; // Maximum division number equipment related
    // The type of unit in which the channel belongs
    DPSDK_INT32 IDecodeMode; // Decoding mode See DPSDK_DECODE_MODE Definition
}DPSDK_DEC_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_ALARMIN_CHANNEL_INFO

Alarm input channel information

```
Typedef Struct
{
    DPSDK_BASE_CHANNEL_INFO StruChannelInfo;
    DPSDK_INT32 IAlarmType; // Alarm type
    DPSDK_INT32 IAlarmLevel; // Alarm level
}DPSDK_ALARMIN_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_ALARMOUT_CHANNEL_INFO

Alarm output channel information

```
Typedef Struct
{
    DPSDK_BASE_CHANNEL_INFO StruChannelInfo;
    DPSDK_INT32 IAlarmType;                    // Alarm type
}DPSDK_ALARMOUT_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_TVWALLIN_CHANNEL_INFO

Large screen input channel data

```
Typedef Struct
{
    DPSDK_BASE_CHANNEL_INFO StruChannelInfo;
    DPSDK_INT32   lCameraType;                                  //   Camera   type   See
DPSDK_CAMERA_TYPEDefinition
    DPSDK_INT32   lChannelRemoteType;                     //   Remote   channel   type   See
DPSDK_CHANNEL_REMOTE_TYPEDefinition
}DPSDK_TVWALLIN_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_TVWALLOUT_CHANNEL_INFO

Large screen output channel data

```
Typedef Struct
{
    DPSDK_BASE_CHANNEL_INFO StruChannelInfo;
    DPSDK_INT32   IDecodeMode;                                    //   Decoding   mode   See
DPSDK_DECODE_MODEDefinition
}DPSDK_TVWALLOUT_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_DOOR_CHANNEL_INFO

Access access data

```
Typedef Struct
{
    DPSDK_BASE_CHANNEL_INFO StruChannelInfo;
    // Unit attributes of a channel
    DPSDK_INT32 IThirdControl;      // Whether third party control is allowed 0no 1yes
}DPSDK_DOOR_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_VOICE_CHANNEL_INFO

Voice Channel Data

```
typedef struct
{
    DPSDK_BASE_CHANNEL_INFO struChannelInfo;
    // Channel Cell Attribute
    DPSDK_CHAR szVoiceIP[DPSDK_IP_LEN];                 // Voice Service Address
    DPSDK_CHAR szClientIP[DPSDK_IP_LEN];                // Voice Client Address
    DPSDK_USHORT ushVoicePort;                          // Voice Service Port
    DPSDK_USHORT ushStatusPort;                         // Voice Status Port
}DPSDK_VOICE_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_ROADGATE_CHANNEL_INFO

Channel gate data

```
Typedef Struct
{
    DPSDK_BASE_CHANNEL_INFO StruChannelInfo;
    DPSDK_CHAR SzSluiceType[DPSDK_TYPE_LEN]; // channel gate type
}DPSDK_ROADGATE_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_LED_CHANNEL_INFO

LEDChannel data

```
Typedef Struct
{
    DPSDK_BASE_CHANNEL_INFO StruChannelInfo;
    DPSDK_INT32 IFreeParkingSpace;          // Residual parking space
    DPSDK_CHAR SzLEDChnlDesc[DPSDK_URL_LEN]; // Description information
}DPSDK_LED_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_DISPATCHER_CHANNEL_INFO

Dispatcher channel data

```
Typedef Struct
{
    DPSDK_BASE_CHANNEL_INFO StruChannelInfo;
    DPSDK_CHAR SzCallNum[DPSDK_TYPE_LEN];     // Phone number
}DPSDK_DISPATCHER_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_POS_CHANNEL_INFO

POSChannel data

```
Typedef Struct
{
    DPSDK_BASE_CHANNEL_INFO StruChannelInfo;
    DPSDK_CHAR SzLinkChnl[DPSDK_CHANNEL_ID_LEN]; // POS Channel binding video 
source
}DPSDK_POS_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_VIRTUAL_CHANNEL_INFO

Virtual Channel Data

```
typedef struct
{
    DPSDK_BASE_CHANNEL_INFO struChannelInfo;
}DPSDK_VIRTUAL_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_DEV_TYPE

Device type， match web

```
typedef enum
{
    DEV_TYPE_ENC_BEGIN          = 0,                              // encode device
    DEV_TYPE_DVR               = DEV_TYPE_ENC_BEGIN + 1,           // DVR
    DEV_TYPE_IPC               = DEV_TYPE_ENC_BEGIN + 2,           // IPC
    DEV_TYPE_NVS               = DEV_TYPE_ENC_BEGIN + 3,           // NVS
    DEV_TYPE_MCD               = DEV_TYPE_ENC_BEGIN + 4,           // MCD
    DEV_TYPE_MDVR              = DEV_TYPE_ENC_BEGIN + 5,           // MDVR
    DEV_TYPE_NVR               = DEV_TYPE_ENC_BEGIN + 6,           // NVR
    DEV_TYPE_SVR               = DEV_TYPE_ENC_BEGIN + 7,           // SVR
     DEV_TYPE_PCNVR              = DEV_TYPE_ENC_BEGIN + 8,            // PCNVR,  PSS self-
carried small server
    DEV_TYPE_PVR               = DEV_TYPE_ENC_BEGIN + 9,           // PVR
    DEV_TYPE_EVS               = DEV_TYPE_ENC_BEGIN + 10,          // EVS
    DEV_TYPE_MPGS              = DEV_TYPE_ENC_BEGIN + 11,          // MPGS
    DEV_TYPE_SMART_IPC          = DEV_TYPE_ENC_BEGIN + 12,          // SMART_IPC
    DEV_TYPE_SMART_TINGSHEN     = DEV_TYPE_ENC_BEGIN + 13,          // hearing host
    DEV_TYPE_SMART_NVR          = DEV_TYPE_ENC_BEGIN + 14,          // SMART_NVR
    DEV_TYPE_PRC               = DEV_TYPE_ENC_BEGIN + 15,          // capsule
    DEV_TYPE_JT808             = DEV_TYPE_ENC_BEGIN + 18,          // JT808
    DEV_TYPE_THDVR              = DEV_TYPE_ENC_BEGIN + 19,          // enter via third party
server
    DEV_TYPE_VTT               = DEV_TYPE_ENC_BEGIN + 21,          // VTT
    DEV_TYPE_DSJ               = DEV_TYPE_ENC_BEGIN + 27,          // DSJ
     DEV_TYPE_MASTERSLAVE         = DEV_TYPE_ENC_BEGIN + 34,            // master slave
tracking all-in-one IPC (multiple channel)
    DEV_TYPE_MCS               = DEV_TYPE_ENC_BEGIN + 35,          // micro wave
    DEV_TYPE_ENC_END,
    DEV_TYPE_TVWALL_BEGIN       = 100,
    DEV_TYPE_BIGSCREEN          = DEV_TYPE_TVWALL_BEGIN + 1,        // video wall
    DEV_TYPE_TVWALL_END,
    DEV_TYPE_DEC_BEGIN          = 200,                            // decode device
    DEV_TYPE_NVD               = DEV_TYPE_DEC_BEGIN + 1,           // NVD
    DEV_TYPE_SNVD              = DEV_TYPE_DEC_BEGIN + 2,           // SNVD
    DEV_TYPE_UDS               = DEV_TYPE_DEC_BEGIN + 5,           // UDS
    DEV_TYPE_DEC_END,
    DEV_TYPE_MATRIX_BEGIN       = 300,                            // matrix device
    DEV_MATRIX_M60             = DEV_TYPE_MATRIX_BEGIN  + 1,       // M60
    DEV_MATRIX_NVR6000          = DEV_TYPE_MATRIX_BEGIN + 2,        // NVR6000
    DEV_TYPE_MATRIX_END,
    DEV_TYPE_IVS_BEGIN          = 400,                              // IVS device
    DEV_TYPE_ISD                = DEV_TYPE_IVS_BEGIN + 1,           // ISD smart dome
    DEV_TYPE_IVS_B              = DEV_TYPE_IVS_BEGIN + 2,           // IVS-B
    DEV_TYPE_IVS_V              = DEV_TYPE_IVS_BEGIN + 3,           // IVS-V
    DEV_TYPE_IVS_FR             = DEV_TYPE_IVS_BEGIN + 4,           // IVS-FR
    DEV_TYPE_IVS_PC             = DEV_TYPE_IVS_BEGIN + 5,           // IVS-PC
    DEV_TYPE_IVS_M              = DEV_TYPE_IVS_BEGIN + 6,           // IVS_M
    DEV_TYPE_IVS_PC_BOX         = DEV_TYPE_IVS_BEGIN + 7,           // IVS-PC
    DEV_TYPE_IVS_B_BOX          = DEV_TYPE_IVS_BEGIN + 8,           // IVS-B
    DEV_TYPE_IVS_M_BOX          = DEV_TYPE_IVS_BEGIN + 9,           // IVS-M
    DEV_TYPE_IVS_PRC            = DEV_TYPE_IVS_BEGIN + 10,          // capsule
    DEV_TYPE_IVS_END,
    DEV_TYPE_BAYONET_BEGIN      = 500,                              // -C relative device
    DEV_TYPE_CAPTURE            = DEV_TYPE_BAYONET_BEGIN + 1,       // ANPR device
     DEV_TYPE_SPEED             = DEV_TYPE_BAYONET_BEGIN + 2,       // measure speed
device
    DEV_TYPE_TRAFFIC_LIGHT      = DEV_TYPE_BAYONET_BEGIN + 3,       // run red light
device
     DEV_TYPE_INCORPORATE       = DEV_TYPE_BAYONET_BEGIN + 4,       // all in one
device
     DEV_TYPE_PLATEDISTINGUISH    =  DEV_TYPE_BAYONET_BEGIN + 5,       // plate
recognition device
     DEV_TYPE_VIOLATESNAPPIC      = DEV_TYPE_BAYONET_BEGIN + 6,       // illegal
parking detection device
     DEV_TYPE_PARKINGSTATUSDEV   = DEV_TYPE_BAYONET_BEGIN + 7,       // parking
detection device
     DEV_TYPE_ENTRANCE           = DEV_TYPE_BAYONET_BEGIN + 8,       // entrance/exit
device
     DEV_TYPE_VIOLATESNAPBALL    = DEV_TYPE_BAYONET_BEGIN + 9,       // illegal
parking snapshot speed dome
     DEV_TYPE_THIRDBAYONET        =  DEV_TYPE_BAYONET_BEGIN + 10,       // third
parking ANPR device
     DEV_TYPE_ULTRASONIC         = DEV_TYPE_BAYONET_BEGIN + 11,      // ultrasonic
parking detector
     DEV_TYPE_FACE_CAPTURE        =  DEV_TYPE_BAYONET_BEGIN + 12,       // face
snapshot device
     DEV_TYPE_ITC_SMART_NVR      = DEV_TYPE_BAYONET_BEGIN + 13,      // ANPR
intelligent NVR device
    DEV_TYPE_PARKINGAREASNAP    = DEV_TYPE_BAYONET_BEGIN + 14,      // parking
zone snapshot device
     DEV_TYPE_ITC_EVS            = DEV_TYPE_BAYONET_BEGIN + 15,      // EVS storage
device
    DEV_TYPE_BAYONET_END,
    DEV_TYPE_ALARM_BEGIN       = 600,                           // alarm device
     DEV_TYPE_ALARMHOST              =  DEV_TYPE_ALARM_BEGIN + 1,             //  alarm
controller
    DEV_TYPE_ALARM_END,
    DEV_TYPE_DOORCTRL_BEGIN     = 700,
    DEV_TYPE_DOORCTRL_DOOR      = DEV_TYPE_DOORCTRL_BEGIN + 1,      // A&C
    DEV_TYPE_DOORCTRL_END,
    DEV_TYPE_PE_BEGIN          = 800,
    DEV_TYPE_PE_PE             = DEV_TYPE_PE_BEGIN + 1,           // PE
    DEV_TYPE_PE_AE6016          = DEV_TYPE_PE_BEGIN + 2,           // AE6016 device
     DEV_TYPE_PE_NVS              = DEV_TYPE_PE_BEGIN + 3,            // NVS device with PE
function
    DEV_TYPE_PE_END,
    DEV_TYPE_VOICE_BEGIN       = 900,                           // ip talk
    DEV_TYPE_VOICE_MIKE        = DEV_TYPE_VOICE_BEGIN + 1,
    DEV_TYPE_VOICE_NET         = DEV_TYPE_VOICE_BEGIN + 2,
    DEV_TYPE_VOICE_END,
     DEV_TYPE_IP_BEGIN            = 1000,                             // IP device (connect device via
network)
    DEV_TYPE_IP_SCNNER         = DEV_TYPE_IP_BEGIN + 1,           // scanner
    DEV_TYPE_IP_SWEEP          = DEV_TYPE_IP_BEGIN + 2,           // sweep
    DEV_TYPE_IP_POWERCONTROL      =  DEV_TYPE_IP_BEGIN +  3,                 //  power
controller
    DEV_TYPE_IP_END,
     DEV_TYPE_MULTIFUNALARM_BEGIN= 1100,                              // multi-functional alarm
controller
     DEV_TYPE_VEDIO_ALARMHOST     = DEV_TYPE_MULTIFUNALARM_BEGIN + 1, //
video alarm controller
    DEV_TYPE_MULTIFUNALARM_END,
    DEV_TYPE_SLUICE_BEGIN      = 1200,
     DEV_TYPE_SLUICE_DEV         = DEV_TYPE_SLUICE_BEGIN + 1,        // ANPR barrier
device
     DEV_TYPE_SLUICE_PARKING      =  DEV_TYPE_SLUICE_BEGIN +  2,           //  parking
barrier device
    DEV_TYPE_SLUICE_STOPBUFFER  = DEV_TYPE_SLUICE_BEGIN + 3,        // video stop
buffer
    DEV_TYPE_SLUICE_END,
    DEV_TYPE_ELECTRIC_BEGIN     = 1300,
    DEV_TYPE_ELECTRIC_DEV       = DEV_TYPE_ELECTRIC_BEGIN + 1,      // grid device
    DEV_TYPE_ELECTRIC_END,
    DEV_TYPE_LED_BEGIN         = 1400,
    DEV_TYPE_LED_DEV           = DEV_TYPE_LED_BEGIN + 1,          // LED device
    DEV_TYPE_LED_END,
    DEV_TYPE_VIBRATIONFIBER_BEGIN    = 1500,
    DEV_TYPE_VIBRATIONFIBER_DEV      = DEV_TYPE_VIBRATIONFIBER_BEGIN + 1,    //
vibration fiber device
    DEV_TYPE_VIBRATIONFIBER_END,
    DEV_TYPE_PATROL_BEGIN           = 1600,
     DEV_TYPE_PATROL_DEV                = DEV_TYPE_PATROL_BEGIN + 1,              // patrol
wand device
     DEV_TYPE_PATROL_SPOT              = DEV_TYPE_PATROL_BEGIN + 2,              // patrol
point device
    DEV_TYPE_PATROL_END,
    DEV_TYPE_SENTRY_BOX_BEGIN       = 1700,
     DEV_TYPE_SENTRY_BOX_DEV          = DEV_TYPE_SENTRY_BOX_BEGIN + 1,          //
sentry box device
    DEV_TYPE_SENTRY_BOX_END,
    DEV_TYPE_COURT_BEGIN            = 1800,
     DEV_TYPE_COURT_DEV                  = DEV_TYPE_COURT_BEGIN + 1,                // court
device
    DEV_TYPE_COURT_END,
    DEV_TYPE_VIDEO_TALK_BEGIN       = 1900,
    DEV_TYPE_VIDEO_TALK_VTNC        = DEV_TYPE_VIDEO_TALK_BEGIN + 1,
    DEV_TYPE_VIDEO_TALK_VTO         = DEV_TYPE_VIDEO_TALK_BEGIN + 2,
    DEV_TYPE_VIDEO_TALK_VTH         = DEV_TYPE_VIDEO_TALK_BEGIN + 3,
    DEV_TYPE_VIDEO_TALK_DOORLOCK_VTH = DEV_TYPE_VIDEO_TALK_BEGIN + 6,
    DEV_TYPE_VIDEO_TALK_END,
    DEV_TYPE_BROADCAST_BEGIN        = 2000,
        DEV_TYPE_BROADCAST_ITC_T6700R      = DEV_TYPE_BROADCAST_BEGIN + 1,
            // ITC_T6700R broadcast device
    DEV_TYPE_BROADCAST_END,
    DEV_TYPE_VIDEO_RECORD_SERVER_BEGIN   = 2100,
                                DEV_TYPE_VIDEO_RECORD_SERVER_BNVR                          =
DEV_TYPE_VIDEO_RECORD_SERVER_BEGIN + 1, // BNVR device
                          DEV_TYPE_VIDEO_RECORD_SERVER_OE                                  =
DEV_TYPE_VIDEO_RECORD_SERVER_BEGIN + 2, // surgery device(operation equipment)
    DEV_TYPE_VIDEO_RECORD_SERVER_END,
    DEV_TYPE_DISPATCHER_BEGIN           = 2200,
      DEV_TYPE_DISPATCHER                    = DEV_TYPE_DISPATCHER_BEGIN + 1,
//dispatch device
    DEV_TYPE_DISPATCHER_END,
    DEV_TYPE_ALARM_STUB_BEGIN       = 3400,                              // alarm tower device
typpw
    DEV_TYPE_ALARM_STUB_VTA         = DEV_TYPE_ALARM_STUB_BEGIN + 1,
    DEV_TYPE_ALARM_STUB_END,
    DEV_TYPE_POS_BEGIN              = 4000,
    DEV_TYPE_POS_BOX                = DEV_TYPE_POS_BEGIN + 1,           // POS box
    DEV_TYPE_POS_END,
}DPSDK_DEV_TYPE;
```

Father theme:structural morphology

DPSDK_BASE_CHANNEL_INFO

Channel basic information

```
Typedef Struct
{
    DPSDK_CHAR SzChannelID[DPSDK_CHANNEL_ID_LEN]; // channel ID
    DPSDK_CHAR SzChannelName[DPSDK_NAME_LEN]; // Channel name
    DPSDK_CHAR SzChnlSN[DPSDK_SN_LEN];          // channel SN code
    DPSDK_INT32 IChannelType;                   // Channel type See DPSDK_CHANNEL_TYPE
Definition Only coded channels are currently available classification
    DPSDK_INT32 IStatus;                        // Channel state See DPSDK_DEV_STATUS
Definition
}DPSDK_BASE_CHANNEL_INFO;
```

Father theme:structural morphology

DPSDK_CAMERA_TYPE

Camera type

```
Typedef Enum
{
    CAMERA_TYPE_NORMAL,     // Bolt
    CAMERA_TYPE_SD,         // Speed Dome Cameras
    CAMERA_TYPE_HALFSD,     // hemisphere
    CAMERA_TYPE_EVIDENCE, // Evidence channel
}DPSDK_CAMERA_TYPE;
```

Father theme:structural morphology

DPSDK_CHANNEL_REMOTE_TYPE

M60 M30 M70Remote channel type

```
Typedef Enum
{
    REMOTE_TYPE_UNKNOW = 0,
    REMOTE_TYPE_LOCAL_CHAN = 1,     //Local coded channel
    REMOTE_TYPE_REMOTE_CHAN = 2, //Remote channel
    REMOTE_TYPE_LOWER_CHAN = 3,    //Cascaded channel
    REMOTE_TYPE_MATRIX_CHAN = 4, //Analog matrix channel
}DPSDK_CHANNEL_REMOTE_TYPE;
```

Father theme:structural morphology

DPSDK_DECODE_MODE

Decoder video source

```
Typedef Enum
{
    DECODE_MODE_UNDEFINE,   // Undefined
    DECODE_MODE_ACTIVE,     // active
    DECODE_MODE_PASSIVE,    // passive
    DECODE_MODE_PUSH,       // Push flow
}DPSDK_DECODE_MODE;
```

Father theme:structural morphology

DPSDK_LEN_TYPE

Lens type

```
Typedef Enum
{
     DPSDK_LENTYPE_NORM = 0,        // Distortion free lens
     DPSDK_LENTYPE_Lens0361 = 1,    // Three point sixMm bolt shot
     DPSDK_LENTYPE_Lens2880 = 2,        // One hundred and thirtyThe degree of wide-angle 
lens.
     DPSDK_LENTYPE_Lens0362 = 3,     // Three point sixMm bolt shot
     DPSDK_LENTYPE_Lens0401 = 4,     // FourMm bolt shot
     DPSDK_LENTYPE_TEST1 = 100,      // Debug parameters
}DPSDK_LEN_TYPE;
```

Father theme:structural morphology

DPSDK_CAM_TYPE

Bolt type

```
Typedef Enum
{
     // Bolt type
     DPSDK_IPCTYPE_200WN = 0,
     DPSDK_IPCTYPE_130WN = 1,
     DPSDK_IPCTYPE_D1WN = 2,
     DPSDK_IPCTYPE_100WN = 3,
     DPSDK_IPCTYPE_FE = 4, // fisheye
     // Type of ball machine
     DPSDK_SPCTYPE_D6501 = 100,     // Sony Movement Sixty-fiveSpeed Dome Cameras
     DPSDK_HSPCTYPE_D6A2030E = 101, // core 2030E,   6ASpeed Dome Cameras
}DPSDK_CAM_TYPE;
```

Father theme:structural morphology

DPSDK_DEMUXDEC_CALLBACK

Data callback for the analysis of source data

```
Typedef DPSDK_VOID (* DPSDK_DEMUXDEC_CALLBACK) (
    DPSDK_LPVOID PUserData,
    DPSDK_INT32 IEncode
);
```

Father theme:structural morphology

DPSDK_EVENT_CALLBACK

event callbacks

```
Typedef DPSDK_VOID (* DPSDK_EVENT_CALLBACK) (
     DPSDK_INT32 IEventType,
     DPSDK_INT32 IMediaSessionID,
     DPSDK_VOID* PUserParam
);
```

Father theme:structural morphology

DPSDK_TVWALL_PLAYBACK_CALLBACK

Back wall callback function

```
Typedef DPSDK_INT32 (* DPSDK_TVWALL_PLAYBACK_CALLBACK) (
     DPSDK_CHAR* PData,
     DPSDK_INT32 IDataLen,
     DPSDK_VOID* PUserParam
);
```

Father theme:structural morphology

DPSDK_PTZ_PRESETPOINT_INFO

Get the preset point information

```
Typedef Struct
{
    DPSDK_CHAR SzPointName[DPSDK_PRESETPOINT_NAME_LEN]; // Preset point name
    DPSDK_CHAR    SzPointCode[DPSDK_PRESETPOINT_CODE_LEN];    //    Preset    point
encoding, from 1start
    DPSDK_INT32 IPointType; // Preset point type,0=Ordinary preset point,1=Preset points that
have been set for intelligent rules
}DPSDK_PTZ_PRESETPOINT_INFO;
```

Father theme:structural morphology

DPSDK_SCREEN_DECODER_INFO

A list of decoded channels for screen binding

```
Typedef Struct
{
    DPSDK_CHAR  SzDeviceId[DPSDK_DEVICE_ID_LEN];     // Device for decoding channel ID
    DPSDK_CHAR SzChannelId[DPSDK_CHANNEL_ID_LEN]; // Decoding channel ID
    DPSDK_INT32   IChannelNo;       // Channel number,  corresponding XMLIn (SEQChannel number-1To show whether the screen is fused)
    DPSDK_INT32 IUnit;        // Subordinate unit
    DPSDK_INT32          IDecoderType;          //          Decoder          type,          reference DPSDK_TVWALL_DEVICE_TYPE
    DPSDK_BOOL BIsChildrenDecoder; // Splice+The decoder wall scheme is used.True:The video output channel bound by this screen is a splice+Decoder The.
    DPSDK_INT32 IScreenId;                  // screen ID（WebEnd configuration ID）
    DPSDK_CHAR SzScreenName[DPSDK_NAME_LEN]; // The name of the screen
    DPSDK_TVWALL_SCREEN_POS StruScreenPos;   // Screen position
    DPSDK_BOOL BIsCombinedScreen;            // Is it a fusion screen?XMLIn Type is 1 or 3 as it is a combination screen)
    DPSDK_BOOL BIsScreenAlarmWall;           // Is it an alarm linkage screen
    DPSDK_INT32 ICombinedScreenNum;            // The number of ordinary screens contained in the fusion screen
    DPSDK_COMBINED_SCREEN_INFO
SzCombinedScreen[DPSDK_MAX_COMBINED_SCREEN_NUM];      //      Common      screen information contained in the fusion screen
}DPSDK_SCREEN_DECODER_INFO;
```

Father theme:structural morphology

DPSDK_COMBINED_SCREEN_INFO

Common screen information under the fusion screen

```
Typedef Struct
{
    DPSDK_INT32 IId;                     // screen ID
    DPSDK_INT32 IChannelNo;                  // Channel sequence number
    DPSDK_TVWALL_SCREEN_POS StruScreenPos; // Screen position
    DPSDK_INT32 IScreenSeq;                  // Screen serial number, logical screen valid
}DPSDK_COMBINED_SCREEN_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_TASK_CHANNEL_EXT_INFO

The device information required by a video source for a decode device in a direct decoding mode

```
Typedef Struct
{
    DPSDK_CHAR SzID[DPSDK_CHANNEL_ID_LEN]; // channel ID
    DPSDK_CHAR SzIP[DPSDK_IP_LEN];          // device IP(unicast) or multicast IP(multicast)
    DPSDK_INT32 IPort;                   // Port (unicast) or multicast port (multicast)
    DPSDK_CHAR SzUserName[DPSDK_NAME_LEN]; // User name
    DPSDK_CHAR SzPassword[DPSDK_PWD_LEN];   // Password
    DPSDK_INT32 INo;                       // Channel number
    DPSDK_INT32 IChannelNum;                 // Total number of channels
    DPSDK_INT32 IType;                     // device type
    DPSDK_CHAR SzChannelName[DPSDK_CHANNEL_NAME_LEN]; // Channel name
}DPSDK_TVWALL_TASK_CHANNEL_EXT_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_TASK_SCREEN_OPER_INFO

TV wall task screen operation information

```
Typedef Struct
{
    DPSDK_INT32 IScreenID;                         // Screen ID
    DPSDK_INT32 ISplitNum;                         // Screen segmentation/Number of windows
    DPSDK_INT32 IScreenMode;                         // Division=1Open the window=2
    DPSDK_UINT32 UiTotal;                          // Number of window list
    DPSDK_TVWALL_WND_INFO* PWndList;              // Window list
    // Use to save tasks
    DPSDK_CHAR SzDecodeId[DPSDK_DEVICE_ID_LEN]; // Decode device ID
    DPSDK_INT32 ITvIdx;                          // Channel number-1.Representative fusion window
    DPSDK_INT32 IVisitorMode;      // Direct connection (1), pulla (2), push flow to the decoder
(3) DPSDK_DECODER_MODEDefinition
    DPSDK_FLOAT FLeft;                           // The position of the screen, left
    DPSDK_FLOAT FTop;                            // The position of the screen, up
    DPSDK_FLOAT FWidth;                           // The position of the screen, wide
    DPSDK_FLOAT FHeight;                          // The position of the screen, high
    DPSDK_CHAR SzScreenName[DPSDK_NAME_LEN];     // The name of the screen
}DPSDK_TVWALL_TASK_SCREEN_OPER_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_WND_INFO

TV wall window information

```
Typedef Struct
{
     DPSDK_INT32 IWndID;                              // window ID
     DPSDK_FLOAT FLeft;                               // window position left
     DPSDK_FLOAT FTop;                                // up
     DPSDK_FLOAT FWidth;                              // width
     DPSDK_FLOAT FHeight;                             // height
     DPSDK_INT32 IZorder;                                  // zOrder (screen segmentation can ignore this
parameter)
     DPSDK_CHAR  SzName[DPSDK_NAME_LEN];                          // Window title (screen
segmentation"")
     DPSDK_INT32 IsAlarm;                             // Alarm wall=1The upper wall of the client=0
     DPSDK_INT32 IsHighLight;                          // Highlight=1Not bright=0Not handling=-1
     DPSDK_INT32 IsOpenAudio;                                // Audio open=1The audio is closed; not
handled.=-1
     DPSDK_UINT32 UiVideoSourceNum;                    // Video source list length
     DPSDK_TVWALL_VIDEO_SOURCE_INFO* PVideoSourceList; // Video source list
     DPSDK_INT32 IScreenMode;                         // Division=1Open the window=2
     DPSDK_INT32 ISubWinNum;                          // Number of sub windows
     DPSDK_TVWALL_SUBWND_INFO* PSubWndList;             // Operation information of the
window
}DPSDK_TVWALL_WND_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_VIDEO_SOURCE_INFO

Video source information

```
Typedef Struct
{
    DPSDK_CHAR SzChannelCode[DPSDK_CHANNEL_ID_LEN]; // Channel number
    DPSDK_CHAR SzChannelName[DPSDK_CHANNEL_NAME_LEN]; // Channel name
    DPSDK_CHAR SzDeviceCode[DPSDK_DEVICE_ID_LEN]; // device ID
    DPSDK_INT32 IChnlNo; // Channel sequence number
    DPSDK_INT32 ISubStream; // Code stream type See DPSDK_STREAM_TYPEDefinition
    DPSDK_INT32 ITimeSpan; // residence time
    DPSDK_INT32 IPrePoint; // Preset point sequence number
    DPSDK_INT32 IProvider; // Equipment manufacturer
    DPSDK_CHAR SzOsdText[DPSDK_MEMO_LEN]; // OSDinformation
    DPSDK_BOOL BEnableOsd; // OSDEnable
    DPSDK_INT32 IPatrolMode; // Wheel patrol mode Wheel guard=0A non wheel patrolling wall=1A round tour Preview=2；
    DPSDK_INT32 IFishFitMode; // Fish eye installation mode Unknown=-1Wall=0Top loading=1;=2；
    DPSDK_INT32 IFishShowMode; // Fish eye display model Unknown=-1Panoramic view=0Double panoramic view=1A single picture PTZ=2The four picture PTZ=3 ；output+3Pattern=4The original model=5
    DPSDK_CHAR SzDepartmentCode[DPSDK_ORG_CODE_LEN]; // The organization of an organization. ID
    // Use to save tasks
    DPSDK_INT32 ITrackId; // 501PSPackage,Six hundred and oneOriginal frame,Seven hundred and one frame,Eight hundred and one standard frame,Nine hundred and one TSPackage,101RTPStandard flow
    DPSDK_INT32 IConnType; // 0：TCP；1：UDP；2: multicast;3Domain name;4: active registration;5：ONVIF；6：GB28181；7：HTTPWebpage
}DPSDK_TVWALL_VIDEO_SOURCE_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_SUBWND_INFO

TV wall window information

```
Typedef Struct
{
     DPSDK_INT32 IWndID;                              // window ID
     DPSDK_FLOAT FLeft;                               // window position left
     DPSDK_FLOAT FTop;                                // up
     DPSDK_FLOAT FWidth;                              // width
     DPSDK_FLOAT FHeight;                             // height
     DPSDK_INT32 IZorder;                                 // zOrder (screen segmentation can ignore this
parameter)
     DPSDK_CHAR  SzName[DPSDK_NAME_LEN];                          // Window title (screen
segmentation"")
     DPSDK_INT32 IsAlarm;                             // Alarm wall=1The upper wall of the client=0
     DPSDK_INT32 IsHighLight;                         // Highlight=1Not bright=0Not handling=-1
     DPSDK_INT32  IsOpenAudio;                              // Audio open=1The audio is closed; not
handled.=-1
     DPSDK_UINT32 UiVideoSourceNum;                   // Video source list length
     DPSDK_TVWALL_VIDEO_SOURCE_INFO* PVideoSourceList; // Video source list
}DPSDK_TVWALL_SUBWND_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_PROJECT_INFO

TV wall plan information

```
Typedef Struct
{
    DPSDK_CHAR SzProjName[DPSDK_NAME_LEN]; //Plan name
    DPSDK_INT32 IType;                     //Plan type
    DPSDK_INT32 ITaskNum;                  //Number of tasks
    DPSDK_TVWALL_PROJECT_TASK_INFO* PTaskInfoList; //Task information
}DPSDK_TVWALL_PROJECT_INFO;
```

Father theme:structural morphology

DPSDK_TVWALL_PROJECT_TASK_INFO

Task information

```
Typedef Struct
{
    DPSDK_INT32 lTaskId;    // task ID
    DPSDK_TIMET TBeginTime; // start time
    DPSDK_TIMET TEndTime;   // end time
    DPSDK_INT32 lSpan;      // residence time
}DPSDK_TVWALL_PROJECT_TASK_INFO;
```

Father theme:structural morphology

DPSDK_FISH_REGIONPARAM

```
ypedef Struct
{
    DPSDK_INT32 IX;
    DPSDK_INT32 IY;
    DPSDK_INT32 IHAngle;
    DPSDK_INT32 IVAngle;
    DPSDK_INT32 IAvailable;
    DPSDK_INT32 ArriReserved[3];
}DPSDK_FISH_REGIONPARAM;
```

Father theme:structural morphology

DPSDK_FISH_SUBMODE

```
Typedef Struct
{
    DPSDK_UINT32 UiSubMountMode;      // Sub image installation mode, Only when the image master correction mode is user defined mode, This value is valid, See DPSDK_FISH_MOUNTMODE
    DPSDK_UINT32 UiSubCalibrateMode; // Subimage correction model, Only when the image master correction mode is user defined mode, This value is valid, See DPSDK_FISH_SHOWMODES
    DPSDK_FISH_SIZE StruImgOutput;    // Subimage output resolution
    DPSDK_FISH_POINT2D StruUpperLeft; // Subimage offset
    DPSDK_INT32 ArriReserved[3];      // Reserved bytes
}DPSDK_FISH_SUBMODE;
```

Father theme:structural morphology

DPSDK_FISH_POINT2D

```
Typedef Struct
{
    DPSDK_SHORT ShX;
    DPSDK_SHORT ShY;
}DPSDK_FISH_POINT2D;
```

Father theme:structural morphology

DPSDK_FISHEYE_CALLBACK

```
Typedef DPSDK_VOID (*DPSDK_FISHEYE_CALLBACK) (
     DPSDK_UCHAR UszCorrectMode,
     DPSDK_USHORT URadius,
     DPSDK_USHORT UCircleX,
     DPSDK_USHORT UCircleY,
     DPSDK_UINT32 UWidthRatio,
     DPSDK_UINT32 UHeigthRatio,
     DPSDK_UCHAR UszGain,
     DPSDK_UCHAR UszDenoiseLevel,
     DPSDK_UCHAR UszInstallStyle,
     DPSDK_LPVOID PUserData
);
```

Father theme:structural morphology

DPSDK_COLLECTION_ORG_INFO_T

Collect Organizational Info

```
typedef struct DPSDK_COLLECTION_ORG_INFO_T
{
    DPSDK_CHAR szOrgCode[DPSDK_ORG_CODE_LEN];                       // Organization ID
     DPSDK_CHAR  szOrgName[DPSDK_ORG_NAME_LEN];                              // Organization
Name
    DPSDK_INT32 iSort;                                              // Sorting Value
     DPSDK_BOOL bHasData;                                           // Is there directly subordinate data?
(Non-organizational node; it can be channel)
       DPSDK_CHAR  szParentCode[DPSDK_ORG_CODE_LEN];                                 // Parent
Organization ID
    DPSDK_INT32 iSubOrgNum;                                         // Number of Sub-organizations
      DPSDK_COLLECTION_ORG_INFO_T* pSuvOrgList;                                // List of Sub-
organizations
}DPSDK_COLLECTION_ORG_INFO;
```

Father theme:structural morphology

DPSDK_DECODE_TYPE

Decoding Type

```
typedef enum
{
    DPSDK_DECODE_SW = 0,                          //CPU Decoding
    DPSDK_DECODE_HW = 1,                          //GPU Decoding
    DPSDK_DECODE_HW_FAST = 2,                     //GPU Fast Decoding
}DPSDK_DECODE_TYPE;
```

Father theme:structural morphology

DPSDK_STREAM_MODE

Playing Mode

```
typedef enum
{
    DPSDK_STREAM_REAL_MODE       = 0,        // Real-time Priority Mode
   DPSDK_STREAM_SMOOTH_MODE     = 1,        // Fluency Priority Mode
   DPSDK_STREAM_POISE_MODE      = 2,        // Balance Priority Mode
   DPSDK_STREAM_CUSTOM_MODE     = 3,        // Custom Priority Mode
}DPSDK_STREAM_MODE;
```

Father theme:structural morphology

DPSDK_REALDATA_CALLBACK

Media stream callback function

```
Typedef DPSDK_INT32 (* DPSDK_REALDATA_CALLBACK) (
    DPSDK_INT32 IMediaType,
    DPSDK_CHAR* PData,
    DPSDK_INT32 IDataLen,
    DPSDK_VOID* PUserParam
);
```

Father theme:structural morphology

DPSDK_DRAW_CALLBACK

Video plotting callback

```
Typedef DPSDK_VOID (* DPSDK_DRAW_CALLBACK) (DPSDK_HDC HDc,
                                            HCWND PWnd,
                                            DPSDK_LPVOID PUserData);
```

Father theme:structural morphology

DPSDK_CHANNEL_TYPE

Channel type

```
Typedef Enum
{
    CHNL_TYPE_ENC_BEGIN, // Coding channel
    CHNL_TYPE_STREAM = 1, // video
    CHNL_TYPE_PIC,         // picture
    CHNL_TYPE_MIX,         // Double bit stream
    CHNL_TYPE_POS,         // POS channel
    CHNL_TYPE_ENC_END = 10,
}DPSDK_CHANNEL_TYPE;
```

Father theme:structural morphology