IPCAMERA SDK USE MANUAL

Version : 1.0.2.7

Edition update instruction

v1.0.2.2 2011-06-08

1、 HI_SDK_PTZControl和HI_SDK_PTZControlEx Add focus adjust and aperture change order

```
#define HI_CTRL_PTZ_FOCUSIN         0x3007  //focus in
#define HI_CTRL_PTZ_FOCUSOUT        0x3008  //focus out
#define HI_CTRL_PTZ_APERTUREIN      0x3009  //focus enlarger
#define HI_CTRL_PTZ_APERTUREOUT     0x3010  // shrink focus
```

2、 Add Display area Electronic amplification port：HI_SDK_DisplayAll

3、 Add network parameter port:， pls read this option(HI_CMD_NET_EXT) of HI_SDK_SetConfig and HI_SDK_GetConfig.Combine HI_CMD_NET_INFO and HI_CMD_HTTP_PORT.

v1.0.2.0 2010-03-10

1、Add the device reboot port, ， pls read the oftion(HI_SDK_SetConfig) of HI_CMD_REBOOT.

2、Add reset to factory defaults setting port，pls read the option(HI_CMD_RESET)of HI_SDK_SetConfig.

3、Add time synchronization port，pls read the option (HI_CMD_SERVER_TIME) of HI_SDK_SetConfig和HI_SDK_SetConfig.

v1.0.1.9 2010-02-21

1、Alter YUV call back function HI_SDK_SetDecCallBack，call back the callback function, video only call back the YUV data, not display it in the window.;

2、Add this port:HI_SDK_SetDrawWnd，the port is used to set display window, when pWnd is null, DDRAW of the window will be clean up, when pWnd is the window handle,DDRAW will be initialed and display image.;

3、Add this port:HI_SDK_GetMediaAttr to obtain the attitube parameter of setting play video or audio;

v1.0.1.8 2010-12-31

1、Modify record port and allow to add video continue time;

2、Combine Video library port, asf and combined flow videorecording use a public port, but the parameter is different.

v1.0.1.5 2010-12-4

1、Add PTZ original point and up/down/ right/left cruise port, currently only support the device in which the device information contain field “Z0”

v1.0.1.4 2010-12-1

1、Add capture real-time data port: HI_SDK_SaveRealData and HI_SDK_StopSaveRealData，and save to customize form record.

2、Add decode control port:HI_SDK_PauseDecode and HI_SDK_ResumeDecode。

Brief instruction

SDK main function

Preview, parameter settings, alarm, PTZ, record, playback, talkback, snapshot, sound control function

SDK library file instruction

SDK library	hi_sdk_api.h	Header library
	HISDK.lib	LIB library
	HISDK.dll	DLL library
Network library	hi_net_dev_sdk.h	Header library
	NetLib.lib	LIB library
	NetLib.dll	DLL library
Play library	HsPlayer.h	Header library
	HIPlayer.lib	LIB library
	HIPlayer.dll	DLL library
Searching library	hi_vscp_devs_cli.h	Header library
	SearchLib.lib	LIB library
	SearchLib.dll	DLL library
Public file	hi_dataType.h	Header library

Client SDK contains above components, users can choose some components according to their needs.

Following is explaination of each component's function and working conditions.

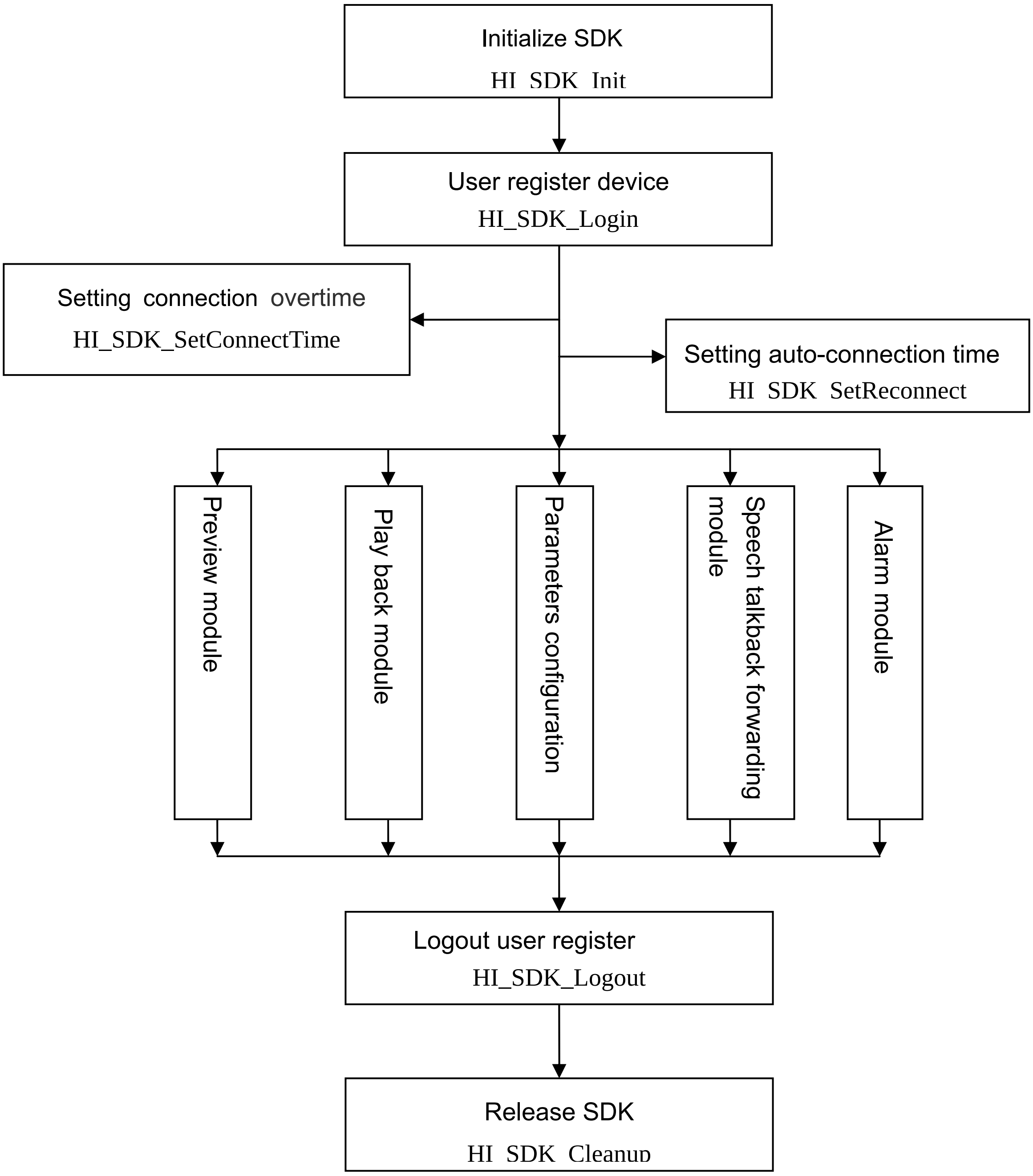
- SDK library：packaging HIPlayer.dll and NetLib.dll, using for non-platform development, please read **SDK user instruction** about physical interface.
- Play library: using for playing data streaming and files, and using for platform development, details please read **Network library user instruction**
- Network library：using for platform development. Pls read **Network library user instruction.**
- Searching library:please refer to<search SDK instruction> about interface instruction.

Programming exploitation platform: Using for network library development forward platform, playing and displaying library handling data, and play record file.

Client programming development: SDK library. Combined with network library and playing library function.

Programming guide

- **SDK** Port invoking main process



According different function, the process can be devided into 5 mmodules.

These four process are necessary during realizing the function: initialize SDK, user register device, logout device，and release SDK recourse。

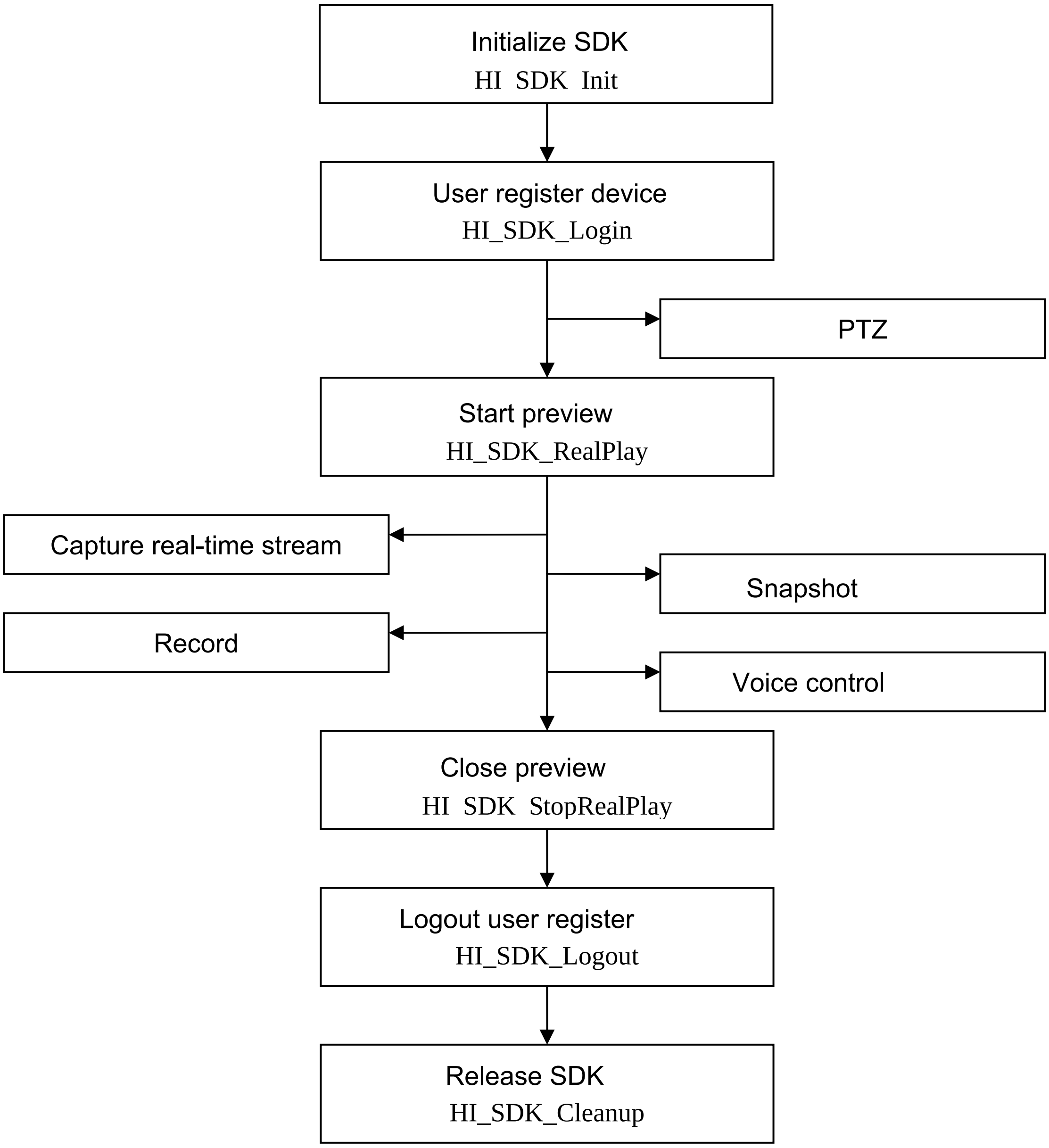
- Initialize SDK （HI_SDK_Init port）：Initialize the whole network SDK system.
- Setting connection overtime (HI_SDK_SetConnectTime port): This is a option choice. It is used to set the time of failure network connection, users can set

this value according their own need. If user do not invoke this port, it will use the SDK’s default value.

- Register user device（HI_SDK_Login port）：Realize the function of user register. After successful in register， the return ID will be used for the unique identifier of operating other function.
- Preview module：After starting preview,you can capture real time data,snapshot， record and control audio. PTZ operation dosenot need to start preview. For more details you can preview module process.
- Playback module：Playback function only support local files playback.
- Parameters configuration module: set up and get forecamera’s parameters. Including device parameters. network parameters. alarm parameters. unormal parameters. user configuration parameters and so on.
- Speech talkback forwarding module: Realizing audio data talkback and audio data capature of the front-end server. Audio Coding’s format can be appointed.

Alarm module: Dealing with all kinds of alarm signal of the front-end server. Alarm data can be devided into “motion alarm” and “input alarm”.

Preview module process



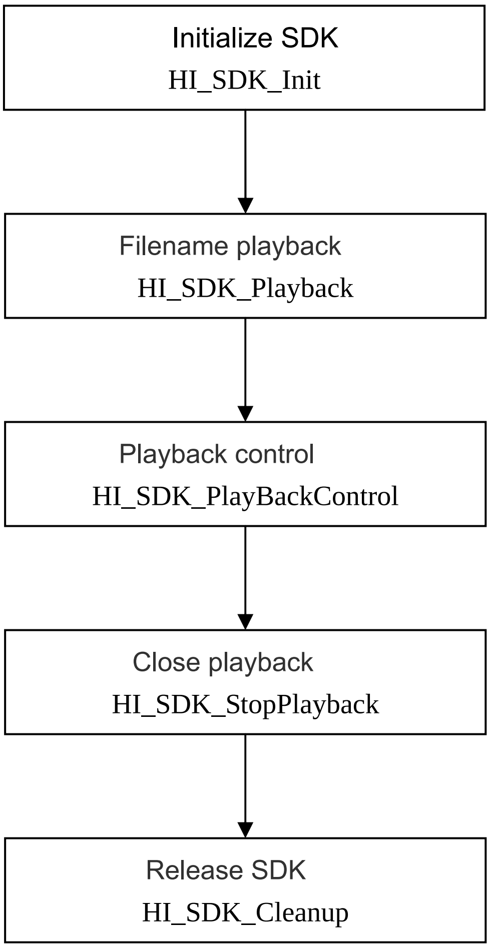
After starting preview, you can catch real-time stream data, snapshot, record, voice control. PTZ operation don't need to start preview.

- Capture real-time data:： When you capture the real-time data, you should re-call the process: HI_SDK_SetRealDataCallBack. The data contains data head.
- Record: The record file has two formats: one is .ASF file format. Another one is customize file format. Pls read record interface instruction（HI_SDK_StartRecord about the record file format and function instruction.
- Snapshot: There are two formats:JPG and BMP. BMP format use port HI_SDK_CapturePicture, JPG format use port HI_SDK_CaptureJPEGPicture.
- Voice control: The function contains: turn on/off volume, adjust the volume. Relating port ：HI_SDK_SetVolume、HI_SDK_GetVolume、HI_SDK_SetMute、H

I_SDK_GetMute and so on.

- PTZ：After successful register, you can operate PTZ, and It dosenot need to preview. Relating port: HI_SDK_PTZControl、HI_SDK_PTZControlEx、HI_SDK_PTZPreset and so on.

Playback module process



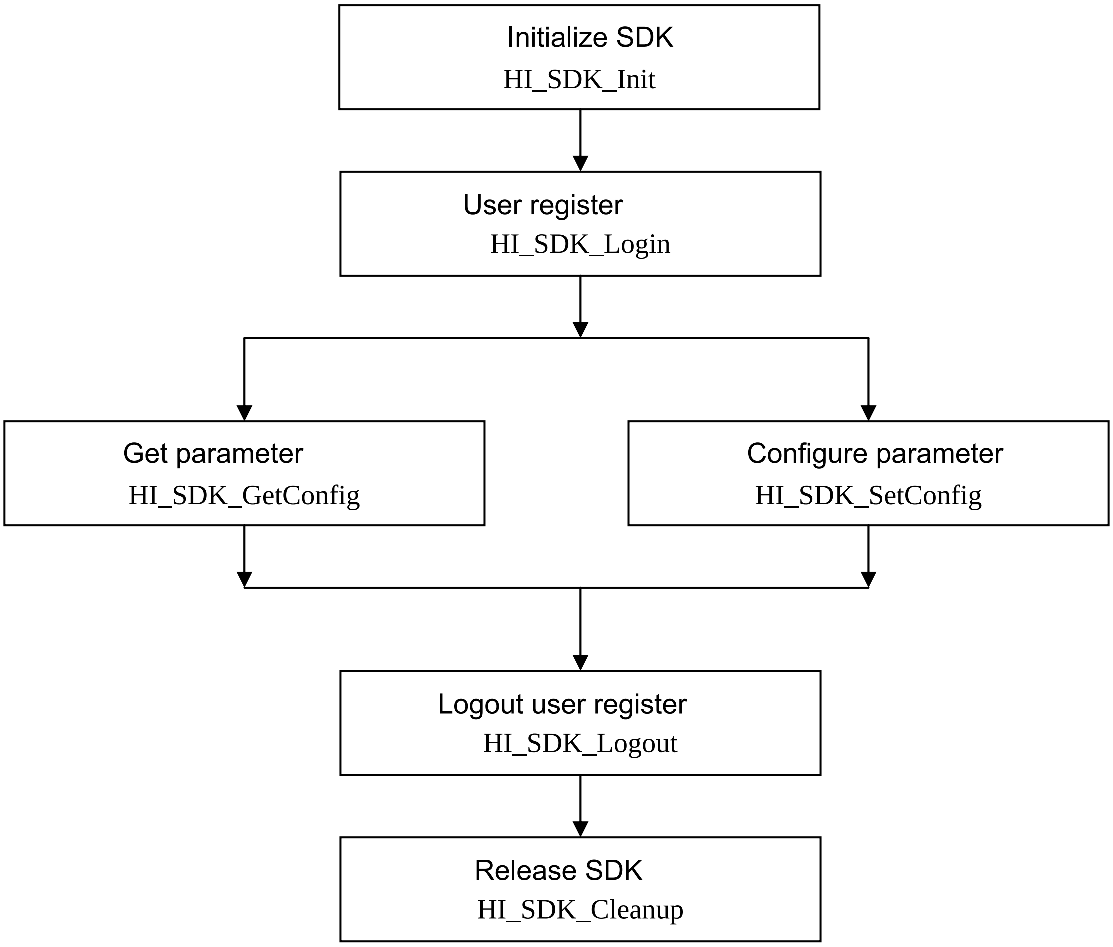
Playback function currently only support local file playback. Relative port: HI_SDK_Playback、HI_SDK_PlayBackControl、HI_SDK_StopPlayback。

Playback control has following function:

- Play: start to play file
- Stop: stop playing, and pointer return file head
- Pause: stop playing
- Adjust play speed: adjust speed
- Single frame: play one frame at one time
- Get/setting play location: skip
- Setting volume：adjust volume
- Get file total time and play time: total time and current play time

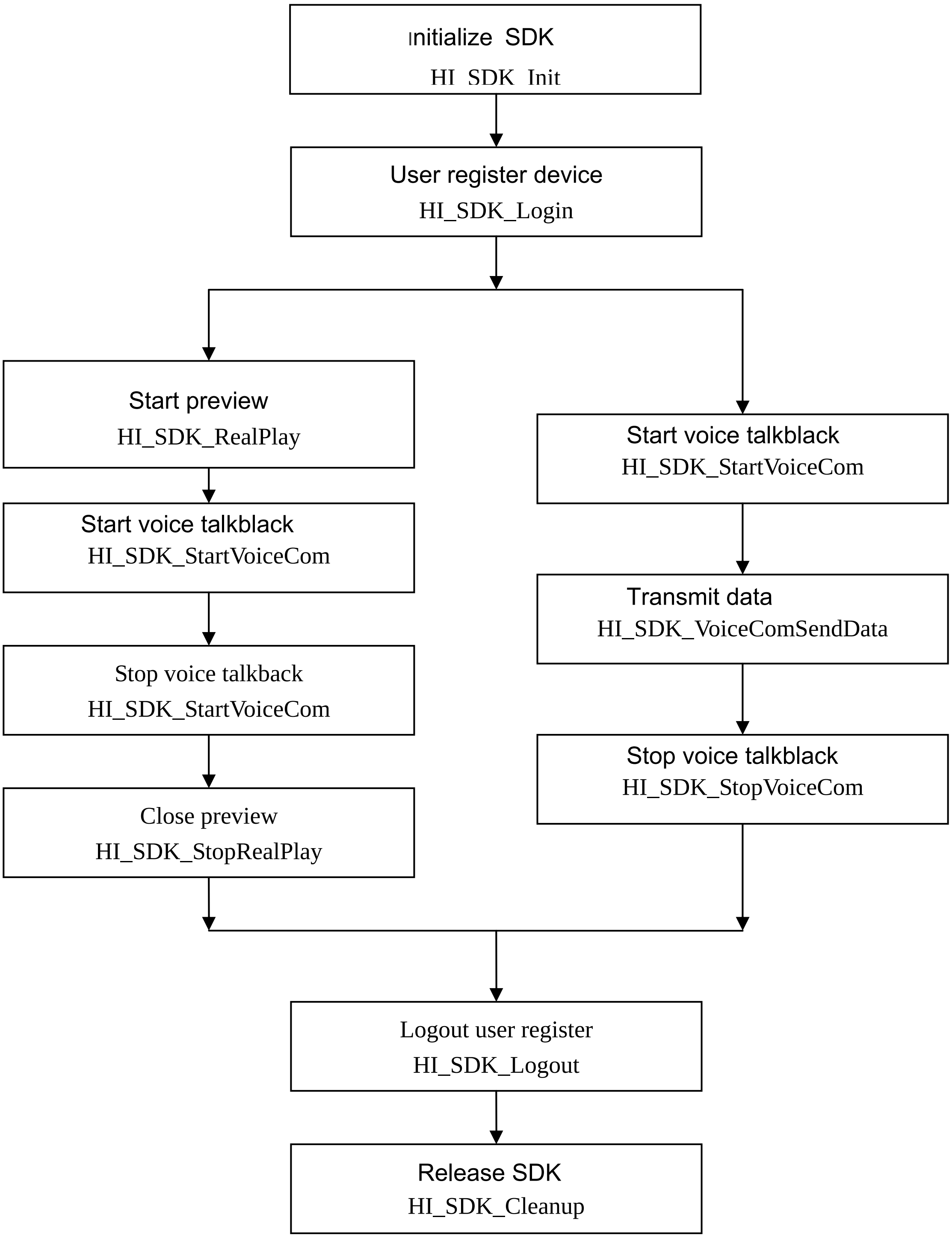
Pls read HI_SDK_PlayBackControl about how to use.

Parameter configuration module process



If you want to realize Parameter Configuration, first you should initialize SDK and register user, and use the handle of user register port. as the first parameter of configure port. Suggestion: Before setting one kind of parameter, pls invoke the port （HI_SDK_GetConfig）of obtaining parameters to get the whole parameter structure, alter the enter parameter, then invoke parameter configuration port（HI_SDK_SetConfig）, if returns success, that means setting successfully.

Voice talkback transmit module process

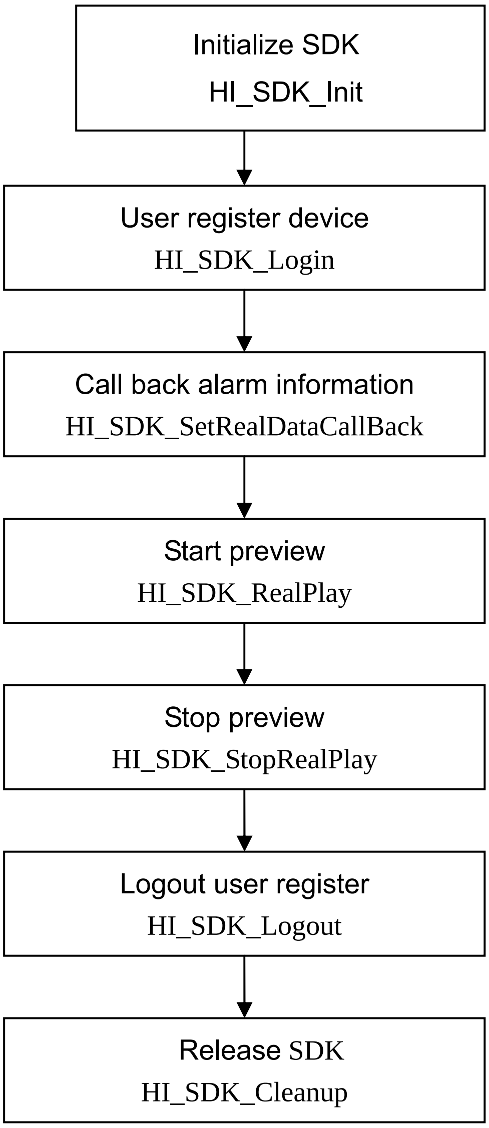


Voice talkback have two way:

1 Getting audio data from PC（data can be obtained from play library, it has been coded）,then send the coded data to camera through SDK.

2 Send prepared voice date to camera, but the audio data format must keep the same with the camera’s format.

Alarm module process:



Alarm call-back have two kinds of method, include “motion alarm” and “input alarm”

- Motion alarm: when montion detection is triggered, call-back function will output the motion detection data.
- input alarm: When the parameters of camera is changed, it will produce input alarm.

Pls read alarm call back functionHI_SDK_SetMessageCallBack.

。

Data type definition instruction:

```
typedef unsigned char           HI_U8;
typedef unsigned char           HI_UCHAR;
typedef unsigned short          HI_U16;
typedef unsigned int            HI_U32;

typedef signed char             HI_S8;
typedef short                   HI_S16;
typedef int                     HI_S32;

#ifndef _M_IX86
typedef unsigned long long      HI_U64;
typedef long long               HI_S64;
#else
typedef __int64                 HI_U64;
typedef __int64                 HI_S64;
#endif

typedef char                    HI_CHAR;
typedef char*                   HI_PCHAR;

typedef float                   HI_FLOAT;
typedef double                  HI_DOUBLE;
typedef void                    HI_VOID;

typedef unsigned long           HI_SIZE_T;
typedef unsigned long           HI_LENGTH_T;


typedef enum {
    HI_FALSE     = 0,
    HI_TRUE      = 1,
} HI_BOOL;

#ifndef NULL
#define NULL                 0L
#endif
#define HI_NULL                 0L
#define HI_NULL_PTR          0L

#define HI_SUCCESS              0
#define HI_FAILURE           (-1)
```

Error definition instruction

```
#define  HI_ERR_SDK_HANDLE            0x30001   //operation handle error
#define  HI_ERR_PLAYER_NULLPTR        0x30002   //play handle error
#define  HI_ERR_DRAW_NULLPTR          0x30003   //draw handle error
#define  HI_ERR_CMD_NULLPTR           0x30004   //parameter is null
#define  HI_ERR_CMD_INVALID_ARG       0x30005   //parameter format error
#define  HI_ERR_CMD_DISCONNECT        0x30006   //connection state is non-connection
#define  HI_ERR_PLAYER_WNDHWND        0x30008   //display handle error
#define  HI_ERR_STATE_IS_PLAYING      0x30009   //play state
#define  HI_ERR_STATE_IS_STOP         0x30010   //stop state
#define  HI_ERR_PLAYER_STOP           0x30011   //stop playing failure
#define  HI_ERR_PLAYER_DEC            0x30012   //decoding failure
#define  HI_ERR_PLAYER_SNAP           0x30013   //snapshot failure
#define  HI_ERR_PLAYER_PLAY           0x30014   //play failure
#define  HI_ERR_PLAYER_STOP_TALK      0x30015   //stop talkback failure
#define  HI_ERR_PLAYER_START_TALK     0x30016   //start talkback failure
#define  HI_ERR_PLAYER_PAUSE          0x30017   //pause failure
#define  HI_ERR_PLAYER_SETRATE        0x30018   //set play speed failure
#define  HI_ERR_PLAYER_ONEBYONE       0x30019   //play single frame failure
#define  HI_ERR_PLAYER_SETPOS         0x30020   //set play location failure
#define  HI_ERR_PLAYER_GETPOS         0x30021   //obtain play location failure
#define  HI_ERR_PLAYER_SETMUTE        0x30022   //set mute failure
#define  HI_ERR_PLAYER_GETMUTE        0x30023   //obtain volumn failure
#define  HI_ERR_PLAYER_SETVOLUME      0x30024   //set volumn failure
#define  HI_ERR_PLAYER_GETVOLUME      0x30025   //obtain volumn failur
#define  HI_ERR_PLAYER_MEDIA_ATTR     0x30026   //set play abttibure failure
#define  HI_ERR_CALLBACK_DRAW         0x32001   //draw call back register failure
#define  HI_ERR_CALLBACK_STATE        0x32002   //state call back form failure
#define  HI_ERR_REC_RECORDING         0x30050   //record state
#define  HI_ERR_REC_START_FAIL        0x30051   //start recording failure
#define  HI_ERR_REC_STOP_FAIL         0x30052   //close recording failure
#define  HI_ERR_TALK_STARTING         0x30081   //talkback state
#define  HI_ERR_TALK_NOSTARTING       0x30082   //talkback close state
#define  HI_ERR_TALK_START_FAIL       0x30083   //open talkback failure
#define  HI_ERR_TALK_SEND_FAIL        0x30084   //talkback transmimission failure
#define  HI_ERR_TALK_STOP_FAIL        0x30085   //stop talkback failure
#define  HI_ERR_PLAYER_OPENFILE       0x30100   //open file failure
#define  HI_ERR_PLAYER_CLOSEFILE      0x30100   //close file failure
```

```
#define    HI_ERR_NET_PLAY                      0x31001      //start                  network
                                                transmission    failure
#define    HI_ERR_NET_STOP                     0x31002      //close                  network
transmission failure
#define    HI_ERR_ATTR_NOSUPPORT               0x31003      //cannot support attribute
```

The Main part 1 Ⅰ SDK user instruction

1.1 initialize SDK

HI_SDK_Init

Initialize, use following SDK port, only use in Initialize

```
HI_S32   HI_SDK_Init (
);
```

Return Values

HI- SUCCESS means successful , HI-FAILRE means fail

HI_SDK_Cleanup

Release SDK， this function should put in the end.

```
HI_S32   HI_SDK_Cleanup (
);
```

Return Values

HI- SUCCESS means successful , HI-FAILRE means fail

Remarks

HI_SDK_Init、HI_SDK_Cleanup only use one time in Initialize, Initialize socket

1.2 User register

HI_SDK_Login

User device register， back handle for user operate this device

```
HI_HANDLE   HI_SDK_Login (
      const HI_CHAR*  psHost,
      const HI_CHAR*  psUsername,
      const HI_CHAR*  psPassword,
      HI_U16          u16Port,
      HI_S32 *              ps32Err
);
```

Parameters

psHost

[IN] main engine, can be IP add, also can be domain name

psUsername

[IN] user name

psPassword

[IN] password

u16Port

[IN] Port numbe

ps32Err

[OUT] error output information

Return Values

Come back HI_HANDLE. Fail back is 0

HI_SDK_Logout

User cancel log-in

```
HI_S32   HI_SDK_Logout t(
        HI_HANDLE       lHandle
);
```

Parameters

lHandle

[IN] Handle operate

Return Values

Success-- HI_SUCCESS， fail—error code

HI_SDK_SetConnectTime

Setting connect timeout， default overtime is 5 second， unit is second

```
HI_S32   HI_SDK_SetConnectTimeout (
        HI_HANDLE       lHandle,t
        HI_U32          u32Timeout
);
```

Parameters

lHandle

[IN] Handle operate

u32Timeout

[IN] timeout， unit is second

Return Values

Success-- HI_SUCCESS， Fail—error code

HI_SDK_SetReconnect

Setting auto connect gap time, default 10second , 0 is no reunion unit is second

```
HI_S32   HI_SDK_SetReconnect (
        HI_HANDLE       lHandle,
        HI_U32              u32Interval
    );
```

Parameters

lHandle

[IN] Handle operate

u32Interval

[IN] Setting auto connect gap time, default 10second , 0 is no reunion unit is second

Return Values

Success-- HI_SUCCESS， Fail—error code

1.3 Timing preview

HI_SDK_RealPlay

Timing data

```
    HI_S32   HI_SDK_RealPlay (
        HI_HANDLE                lHandle,
        HI_VOID*                 pWnd,
        HI_S_STREAM_INFO*        pstruStreamInfo
    );
```

Parameters

lHandle

[IN] Handle operate

PWnd

[IN] appear window handle

pstruStreamInfo

[IN] operate Handle

Return Values

Success-- HI_SUCCESS， Fail—error code

Remarks

// Start flow transmit

```
    typedef struct
    {
        HI_U32       u32Channel;               // Channels
        HI_BOOL      blFlag;              //0-lord streaming，1- Times streaming
        HI_U32       u32Mode;             // Network connection mode
        HI_U8        u8Type;              // Streaming data types, video, audio, other data
    } HI_S_STREAM_INFO;
```

```
// Device channel number，currently only support one channel.
#define HI_CHANNEL_1       1
//#define HI_CHANNEL_2       2
//#define HI_CHANNEL_3       3
//#define HI_CHANNEL_4       4

// Connect internet connect model。 Currently only support TCP
#define HI_STREAM_MODE_TCP    0

// Streaming data types, the present data do not support the heartbeat

// Second data stream does not support the police and the heartbeat data

#define HI_STREAM_VIDEO_ONLY      0x01
#define HI_STREAM_AUDIO_ONLY            0x02
#define HI_STREAM_VIDEO_AUDIO     0x03
#define HI_STREAM_VIDEO_DATA      0x05
#define HI_STREAM_AUDIO_DATA      0x06
#define HI_STREAM_ALL             0x07
```

If need catch timing streaming data, can invoking port HI_SDK_SetRealDataCallBack or HI_SDK_SetDecCallBack Register catch streaming call-back function， and in call back function deal with by itself

HI_SDK_StopRealPlay

Stop data streaming

```
    HI_S32    HI_SDK_StopRealPlay (
        HI_HANDLE        lHandle
    );
```

Parameters

lHandle

[IN] Handle operate

Return Values

Success-- HI_SUCCESS， Fail—error code

1.4 Cameras feature setting

Camera support feature or not， can get trough HI_GET_PRODUCT_VENDOR's sProduct

HI_SDK_SetConfig

Features of setting camera.

```
    HI_S32   HI_SDK_SetConfig (
        HI_U32              u32Handle
        HI_U32              u32Command,
        HI_VOID*            pBuf,
        HI_U32              u32BufLen
    );
```

Parameters

u32Handle

[IN] handle operate

u32Command

[IN] handle feature order

Macro Defined	Macro Defined Values	Implication
HI_CMD_DISPLAY	0x1001	picture feature
HI_CMD_DISPLAY_EXT	0x1002	rollover
HI_CMD_INFRARED	0x1003	infrared
HI_CMD_VIDEO_PARAM	0x1004	video feature
HI_CMD_OSD_PARAM	0x1005	OSD feature
HI_CMD_AUDIO_PARAM	0x1006	voice feature
HI_CMD_AUDIO_INPUT	0x1007	voice input
HI_CMD_RESOLUTION	0x1008	Picture image resolution
HI_CMD_FREQUENCY	0x1009	frequency
HI_CMD_PTZ_PARAM	0x1010	PTZ information
HI_CMD_MD_PARAM	0x1011	motion alarm information
HI_CMD_NET_INFO	0x1012	internet information
HI_CMD_HTTP_PORT	0x1013	webpage port number
HI_CMD_SERVER_TIME	0x1017	setting camera's time
HI_CMD_REBOOT	0x1018	reboot

HI_CMD_RESET	0x1019	recover default setting
HI_CMD_NET_EXT	0x1022	setting hiding area
HI_CMD_ATTR_EXT	0x1026	setting OSD coordinate

pBuf

[IN] setting data

u32BufLen

[IN] data length

Return Values

Success-- HI_SUCCESS， Fail—error code

Remarks

1、HI_CMD_DISPLAY

```
typedef struct HI_Display
{
        HI_U32       u32Brightness;    // brightness, area [0~255]
        HI_U32       u32Saturation;         // saturation, area [0~255]
        HI_U32       u32Contrast;      // contrast, area [0~255]，highdefinition [1~7]
} HI_S_Display;
```

Remark: u32Brightness value equal -1， setting default value。

See Appendix factory color code support and equipment

Device types, define the S field.

Example：

```
HI_S_Display sDisplay;
// sDisplay.u32Brightness = -1;   // setting default value
sDisplay.u32Brightness = 100;
sDisplay.u32Saturation = 100;
sDisplay.u32Contrast = 100;
sDisplay.u32Hue = 100;
HI_SDK_SetConfig (  lHandle,                 // HI_SDK_GetConfig
                    HI_CMD_DISPLAY,
                    &sDisplay,
                    sizeof(HI_S_Display));
```

2、HI_CMD_DISPLAY_EXT

```
typedef struct HI_Display_Ext
{
        HI_BOOL      bIFlip;           // up and down overturn
        HI_BOOL      bIMirror;         // left and right overturn
        HI_S32       s32Scene;         // Scene，auto、indoor、outdoor
} HI_S_Display_Ext;
```

Macro definition	Macro value	Definition
HI_SCENE_AUTO	0	auto
HI_SCENE_INDOOR	1	indoor
HI_SCENE_OUTDOOR	2	outdoor

Example：

```
HI_S_Display_Ext sDisplayEx;
sDisplayEx.blFlip = HI_FALSE;
sDisplayEx.blMirror = HI_FALSE;
sDisplayEx.s32Scene = HI_SCENE_AUTO;
HI_SDK_SetConfig (  lHandle,                  // HI_SDK_GetConfig
                    HI_CMD_DISPLAY_EXT,
                    &sDisplayEx,
                    sizeof(HI_S_Display_Ext));
```

Note: See Appendix device support and device type defined in the manufacturers code the S field.

3、HI_CMD_INFRARED

```
typedef struct HI_Infrared
{
      HI_S32        s32Infrared;        //红外状态开关
} HI_S_Infrared;
```

Macro definition	Macro value	Definition
HI_INFRARED_AUTO	0	auto
HI_INFRARED_ON	1	on
HI_INFRARED_OFF	2	off

Example：

```
HI_S_Infrared sInfrared;
sInfrared.s32Infrared = HI_INFRARED_AUTO;
HI_SDK_SetConfig (  lHandle,                  // HI_SDK_GetConfig
                    HI_CMD_INFRARED,
                    &sInfrared,
                    sizeof(HI_S_Infrared));
```

Note: See Appendix device support and device type defined in the manufacturers code the S field.

4、HI_CMD_VIDEO_PARAM

```
typedef struct HI_Video
{
      HI_U32        u32Channel;        // Channels
```

```
        HI_BOOL     blFlag;         //0-lord streaming，1- Times streaming
        HI_U32      u32Bitrate;     //code rate Kb
        HI_U32      u32Frame;       // frame rate
        HI_U32      u32Iframe;      // Main frame interval（1-300）
        HI_BOOL     blCbr;          //0- VBR，1- CBR
        HI_U32      u32ImgQuality;  // video encoding quality（1-6）
} HI_S_Video;
```

Note: u32Channel and HI_SDK_RealPlay parameters HI_S_STREAM_INFO in u32Channel consistent. Should get and set the same.

Example:

```
HI_S_Video sVideo;
// Note: u32Channel consistent with HI_S_STREAM_INFO
sVideo.u32Channel = HI_CHANNEL_1;
sVideo.blFlag = HI_TRUE;
sVideo.u32Bitrate = 1024;
sVideo.u32Frame = 25;
sVideo.u32Iframe = 50;
sVideo.blCbr = HI_FALSE;
sVideo.u32ImgQuality = 1;
HI_SDK_SetConfig (   lHandle,             // HI_SDK_GetConfig
                     HI_CMD_VIDEO_PARAM,
                     &sVideo,
                     sizeof(HI_S_Video));
```

5、HI_CMD_OSD_PARAM

```
typedef struct HI_OSD
{
        HI_BOOL     blEnTime;       // overlying time
        HI_BOOL     blEnName;       // overlying name
        HI_CHAR     sName[64];      // OSD name// largest is 18 byte
} HI_S_OSD;
```

Example:

```
HI_S_OSD sOSD;
sOSD.blEnTime = HI_TRUE;
sOSD.blEnName = HI_TRUE;
strcpy(sOSD. sName, “IPCAM”);
HI_SDK_SetConfig (   lHandle,             // HI_SDK_GetConfig
                     HI_CMD_OSD_PARAM,
                     &sOSD,
                     sizeof(HI_S_OSD));
```

6、HI_CMD_AUDIO_PARAM

```
typedef struct HI_Audio
```

```
{
        HI_U32        u32Channel;        // channel
        HI_BOOL       blFlag;            //0-lord streaming，1- Times streaming
        HI_BOOL       blEnable;                 // whether collect voice
        HI_U32        u32Type;                  // voice format
} HI_S_Audio;
```

Note: u32Channel and HI_SDK_RealPlay parameters HI_S_STREAM_INFO in u32Channel consistent. Should get and set the same.

u32Type format as following excel:

Macro definition	**Macro value**	**Definitionr**
HI_AUDIO_TYPE_G711	0	G711
HI_AUDIO_TYPE_G726	1	G726
HI_AUDIO_TYPE_AMR	2	AMR

Example:

```
HI_S_Audio sAudio;
// 注：u32Channel 与 HI_S_STREAM_INFO 一致
sAudio.u32Channel = HI_CHANNEL_1;
sAudio.blFlag = HI_TRUE;
sAudio.blEnable = HI_TRUE;
sAudio.u32Type = HI_AUDIO_TYPE_G711;
HI_SDK_SetConfig (  lHandle,                    // HI_SDK_GetConfig
                    HI_CMD_AUDIO_PARAM,
                    &sAudio,
                    sizeof(HI_S_Audio));
```

7、HI_CMD_AUDIO_INPUT

```
typedef enum HI_AudioInput
{
        AUDIO_INPUT_MIC = 100,            // mike input
        AUDIO_INPUT_LINE = 10        // linear input
} HI_E_AudioInput;
```

Example:

```
HI_S32 audioInput = AUDIO_INPUT_MIC;
HI_SDK_SetConfig (  lHandle,                    // HI_SDK_GetConfig
                    HI_CMD_AUDIO_INPUT,
                    &audioInput,
                    sizeof(HI_S32));
```

8、HI_CMD_RESOLUTION

```
typedef struct HI_Resolution
{
        HI_U32        u32Channel;        //channel
        HI_BOOL      bIFlag;            //0-lord streaming，1- Times streaming
        HI_U32       u32Resolution;     // clarity
} HI_S_Resolution;
```

Note: u32Channel and HI_SDK_RealPlay parameters HI_S_STREAM_INFO in u32Channel consistent. Should get and set the same.

u32Type format as following excel:

Macro definition	Values	**Definitionr**
HI_RESOLUTION_VGA	0	VGA：640x480
HI_RESOLUTION_QVGA	1	QVGA：320x240
HI_RESOLUTION_QQVGA	2	QQVGA：160x120，160x112
HI_RESOLUTION_D1	3	D1：704x576，704x480
HI_RESOLUTION_CIF	4	CIF：352x288，352x240
HI_RESOLUTION_QCIF	5	QCIF：176x144，176x120，176x112
HI_RESOLUTION_720P	6	720P：1280x720

Example：

```
HI_S_Resolution sResolution;
// Note: u32Channel consistent with HI_S_STREAM_INFO
sResolution.u32Channel = HI_CHANNEL_1;
sResolution.blFlag = HI_TRUE;
sResolution.u32Resolution = HI_RESOLUTION_CIF;
HI_SDK_SetConfig (   lHandle,                    // HI_SDK_GetConfig
                     HI_CMD_RESOLUTION,
                     &sResolution,
                     sizeof(HI_S_Resolution));
```

Note: See Appendix resolution device support and device type defined in the manufacturers code the S field.

9、HI_CMD_FREQUENCY

```
typedef enum HI_Frequency
{
        FREQ_50HZ_PAL = 50,             //50HZ
        FREQ_60HZ_NTSC = 60        //60HZ
} HI_E_Frequency;
```

Example：

```
HI_U32 sFrequency = FREQ_50HZ_PAL;
HI_SDK_SetConfig (   lHandle,                    // HI_SDK_GetConfig
                     HI_CMD_FREQUENCY,
                     &sFrequency,
                     sizeof(HI_U32));
```

Note: Appendix manufacturer code and device type definition, does not currently support the frequency of the equipment set S1, S2 field.

10、 HI_CMD_PTZ_PARAM

```
typedef struct HI_PTZ
{
        HI_U32        u32Protocol;        // protocol
        HI_U32        u32Address;         //[0~255] add code，  area [0~255]
        HI_U32        u32Baud;                   // Baud rate
        HI_U32        u32DataBit;         // data bit
        HI_U32        u32StopBit;         // StopBit
        HI_U32        u32Parity;          // verify
} HI_S_PTZ;
```

u32Protocol format as following excel

Macro definition	**Macro value**	**Definitionr**
HI_PTZ_PRO_PELCOD	0	PELCO-D
HI_PTZ_PRO_PELCOP	1	PELCO-P

u32Baud Baud rate data as following excel:

Macro definition	**Macro value**	**Definitionr**
HI_PTZ_B110	110	110
HI_PTZ_B300	300	300
HI_PTZ_B1200	1200	1200
HI_PTZ_B2400	2400	2400
HI_PTZ_B4800	4800	4800
HI_PTZ_B9600	9600	9600
HI_PTZ_B19200	19200	19200
HI_PTZ_B38400	38400	38400
HI_PTZ_B57600	57600	57600

u32DataBit data bit as following excel:

Macro definition	**Macro value**	**Definitionr**
HI_PTZ_DATA_5	5	
HI_PTZ_DATA_6	6	
HI_PTZ_DATA_7	7	
HI_PTZ_DATA_8	8	

u32StopBit stop data as following

Macro definition	**Macro value**	**Definitionr**
HI_PTZ_STOP_1	1	
HI_PTZ_STOP_2	2	

u32Parity checking data as following

Macro definition	**Macro**	**Definitionr**

	value	
HI_PTZ_PARITY_NONE	0	No
HI_PTZ_PARITY_ODD	1	Odd
HI_PTZ_PARITY_EVEN	2	Even parity

Example:

```
HI_S_PTZ sPtz;
sPtz. u32Protocol = HI_PTZ_PRO_PELCOD;
sPtz. u32Address = 1;
sPtz. u32Baud = HI_PTZ_B9600;
sPtz. u32DataBit = HI_PTZ_DATA_8;
sPtz. u32StopBit = HI_PTZ_STOP_1;
sPtz. u32Parity = HI_PTZ_PARITY_NONE;
HI_SDK_SetConfig (  lHandle,                // HI_SDK_GetConfig
                    HI_CMD_PTZ_PARAM,
                    &sPtz,
                    sizeof(HI_S_PTZ));
```

11、 HI_CMD_MD_PARAM

```
typedef struct HI_MD_PARAM
{
    HI_U32      u32Channel;      //channel
    HI_U32      u32Area;             // rectangular aera（1~4）
    HI_BOOL     blEnable;            // enable or not
    HI_U32      u32Sensitivity;  // sensitivity（0~100）
    HI_U32      u32X;            // X coordinate
    HI_U32      u32Y;            // Y coordinate
    HI_U32      u32Width;        // rectangular width
    HI_U32      u32Height;       // rectangular high
} HI_S_MD_PARAM;
```

Example:

```
HI_S_MD_PARAM sMdParam;
// Note: u32Channel consistent with HI_S_STREAM_INFO
sMdParam.u32Channel = HI_CHANNEL_1;
sMdParam.u32Area = 1;
sMdParam.bEnable = HI_TRUE;
sMdParam.u32Sensitivity = 50;
sMdParam.u32X = 100;
sMdParam.u32Y = 100;
sMdParam.u32Width = 200;
sMdParam.u32Height = 200;
HI_SDK_SetConfig (  lHandle,                // HI_SDK_GetConfig
                    HI_CMD_MD_PARAM,
                    &sMdParam,
                                        sizeof(HI_S_MD_PARAM));
```

Note: The second stream does not support the motion detection.

12、 HI_CMD_NET_INFO

```
typedef struct tagHI_NETINFO
{
        HI_CHAR     aszServerIP[40];      // IP address
        HI_CHAR     aszNetMask[40];       // subnet mask
        HI_CHAR     aszGateWay[40];       // gateway
        HI_CHAR     aszMacAddr[40];       // MAC address
        HI_CHAR     aszFDNSIP[40];        //first DNSIP
        HI_CHAR     aszSDNSIP[40];        //DNSIP
        HI_S32      s32DhcpFlag;                 //DHCP
        HI_S32      s32DnsDynFlag;        // motion distribute signal */
}HI_S_NETINFO, *PHI_S_NETINFO;
```

Example:

```
HI_S_NETINFO sNetInfo;
strcpy(sNetInfo. aszServerIP, "192.168.1.88");
… …
HI_SDK_SetConfig (   lHandle,                  // HI_SDK_GetConfig
                     HI_CMD_NET_INFO,
                     &sNetInfo,
                     sizeof(HI_S_NETINFO));
```

13、 HI_CMD_HTTP_PORT

```
typedef struct HI_HTTPPORT
{
        HI_U32      u32HttpPort;
} HI_S_HTTPPORT;
```

Example:

```
HI_S_HTTPPORT sHttpPort;
sHttpPort.u32HttpPort = 80;
HI_SDK_SetConfig (   lHandle,                  // HI_SDK_GetConfig
                     HI_CMD_HTTP_PORT,
                     &sHttpPort,
                     sizeof(HI_S_HTTPPORT));
```

14、 HI_CMD_SERVER_TIME

Setting camera's fore time

```
 typedef struct hiSERVERTIME_INFO_S
 {
        HI_CHAR sTime[32];         //   camera's   time   ,   format   is
2011.03.11.09.12.08
 } HI_S_SERVERTIME;
```

sTime is camera's time，format is 2011.03.11.09.12.08， 2011-3-11 09:12:08

Example：

```
HI_S_SERVERTIME sServerTime;
memcpy(sServerTime.sTime,                    "2011.03.11.09.12.08"                    ,
sizeof(sServerTime.sTimezone));
HI_SDK_SetConfig (   lHandle,
                            HI_CMD_SERVER_TIME,
                            &sServerTime,
                            sizeof(HI_S_SERVERTIME));
```

15、 HI_CMD_REBOOT

reboot camera

Example：

```
HI_SDK_SetConfig (lHandle, HI_CMD_REBOOT,NULL,0);
```

16、 HI_CMD_RESET

recover default setting

Example：

```
HI_SDK_SetConfig (lHandle, HI_CMD_RESET,NULL,0);
```

17、 HI_CMD_COVER_PARAM

setting hiding area, can not see this area image

```
#define HI_NET_DEV_COVER_AREA_MAX       4
#define HI_NET_DEV_COVER_AREA_1 1
#define HI_NET_DEV_COVER_AREA_2 2
#define HI_NET_DEV_COVER_AREA_3 3
#define HI_NET_DEV_COVER_AREA_4 4

typedef struct HI_COVER_PARAM
{
HI_U32 u32Area; // hiding area, best offer is 4 areas
HI_BOOL bExpress; // appear or not HI_TRUE- appear, HI_FALSE-disappear
HI_U32 u32X; //X coordinate
HI_U32 u32Y; // Y coordinate
HI_U32 u32Width; // Width
HI_U32 u32Height; //Height
HI_U32 u32Color; // color decimal system
} HI_S_COVER_PARAM;
```

Example：

```
HI_S_COVER_PARAM sCover;
sCover.u32Area = HI_NET_DEV_COVER_AREA_1;
sCover.bExpress = HI_TRUE;
sCover.u32X = 100;
sCover.u32Y = 100;
sCover.u32Width = 100;
sCover.u32Height = 100;
sCover.u32Color = 0;
HI_SDK_SetConfig ( lHandle,
HI_CMD_COVER_PARAM,
&sCover,
Sizeof(HI_S_COVER_PARAM));
```

Note: If you set the coordinates within the image area is not large in the image area, or high, the input is invalid.

18、 HI_CMD_OSDEX_PARAM

Set OSD coordinates

```
#define HI_OSD_TIME 0 //time area
#define HI_OSD_NAME 1 //name area
typedef struct HI_OSD_EX
{
HI_U32 u32Area; //region type
HI_U32 u32X; //X coordinate
HI_U32 u32Y; //Y coordinate
} HI_S_OSD_EX;
```

Example:

```
HI_S_OSD_EX sOsdEx;
sOsdEx.u32Area = HI_OSD_TIME;
sOsdEx.u32X = 100;
sOsdEx.u32Y = 100;
HI_SDK_SetConfig ( lHandle,
HI_CMD_OSDEX_PARAM,
&sOsdEx,
Sizeof(sOsdEx));
```

Note: If you set the coordinates are not within the image area, coordinate the input is invalid.

HI_SDK_GetConfig

Get setting cameras feature

```
HI_S32   HI_SDK_GetConfig (
        HI_U32          u32Handle
        HI_U32          u32Command,
        HI_VOID*        pBuf,
        HI_U32          u32BufLen
);
```

Parameters

u32Handle

[IN] Handle operate

u32Command

[IN] operate feature order

Macro definition	Macro value	Definitionr
HI_GET_PRODUCT_VENDOR	0x1000	manufacturer information
HI_CMD_DISPLAY	0x1001	image feature
HI_CMD_DISPLAY_EXT	0x1002	up&down discolor balance
HI_CMD_INFRARED	0x1003	infrared
HI_CMD_VIDEO_PARAM	0x1004	video feature
HI_CMD_OSD_PARAM	0x1005	OSD video feature
HI_CMD_AUDIO_PARAM	0x1006	voice feature
HI_CMD_AUDIO_INPUT	0x1007	voice input
HI_CMD_RESOLUTION	0x1008	image resolution ratio
HI_CMD_FREQUENCY	0x1009	frequency
HI_CMD_PTZ_PARAM	0x1010	PTZ informatio
HI_CMD_MD_PARAM	0x1011	motion alarm information
HI_CMD_NET_INFO	0x1012	network configuration information
HI_CMD_HTTP_PORT	0x1013	webside port number
HI_CMD_DEVICE_INFO	0x1014	device information
HI_CMD_PRODUCTID	0x1015	products ID
HI_CMD_USERNUM	0x1016	user connect data
HI_CMD_SERVER_TIME	0x1017	get camera’s time
HI_CMD_NET_EXT	0x1020	get hiding area
HI_CMD_ATTR_EXT	0x1021	get OSD coordinate

pBuf

[OUT] GET DATE

u32BufLen

[IN] DATA length

Return Values

Success- HI_SUCCESS, fail- error code

Remarks

Each oder structure according with HI_SDK_SetConfig, inside HI_SDK_SetConfig have noas following :

1、HI_GET_PRODUCT_VENDOR

```
typedef struct HI_ProductVendor
{
      HI_CHAR      sProduct[32];      // products ID
      HI_CHAR      sVendor[32];       // suppliers ID
}HI_S_ProductVendor;
```

Example：

```
HI_S_ProductVendor sProduct;
HI_SDK_GetConfig (  lHandle,
                    HI_GET_PRODUCT_VENDOR,
                    &sProduct,
                    sizeof(HI_S_ProductVendor));
```

2、HI_CMD_DEVICE_INFO

```
typedef struct tagHI_DEVICE_INFO
{
      HI_CHAR aszServerSerialNumber[40 + 1];        // device order number
      HI_CHAR aszServerSoftVersion[64 + 1];      // software edition
      HI_CHAR aszServerName[40 + 1];              // server name
      HI_CHAR aszServerModel[40 + 1];             // type
      HI_CHAR aszStartDate[40 + 1];                    // System startup date
      HI_S32 s32ConnectState;                     // Internet Connect condition
}HI_DEVICE_INFO, *PHI_DEVICE_INFO;
```

Example：

```
HI_DEVICE_INFO sDeviceInfo;
HI_SDK_GetConfig (  lHandle,
                    HI_CMD_DEVICE_INFO,
                    &sDeviceInfo,
                    sizeof(HI_DEVICE_INFO));
```

3、HI_CMD_PRODUCTID

Products ID, use character string express

Example：

```
HI_CHAR sID[64] = {0};
HI_SDK_GetConfig(lHandle, HI_CMD_PRODUCTID, sID, sizeof(sID));
```

4、HI_CMD_USERNUM

Get users date use in input

Example：

```
int nNum = 0;
HI_SDK_GetConfig(lHandle, HI_CMD_USERNUM, &nNum, sizeof(int));
```

5、HI_CMD_SERVER_TIME

Get cameras fore timing

```
typedef struct hiSERVERTIME_INFO_S
{
    HI_CHAR sTime[32];          // Camera time, format 20110311091208
} HI_S_SERVERTIME;
```

sTime is camera's time, format is 20110311091208，the same 2011-3-11 09:12:08

Example：

```
HI_S_SERVERTIME sServerTime;
HI_SDK_GetConfig (  lHandle,
                    HI_CMD_SERVER_TIME,
                    &sServerTime,
                    sizeof(HI_S_SERVERTIME));
```

6、HI_CMD_COVER_PARAM

get hiding area

```
#define HI_NET_DEV_COVER_AREA_MAX 4
#define HI_NET_DEV_COVER_AREA_1 1
#define HI_NET_DEV_COVER_AREA_2 2
#define HI_NET_DEV_COVER_AREA_3 3
#define HI_NET_DEV_COVER_AREA_4 4
typedef struct HI_COVER_PARAM
{
HI_U32 u32Area; // Hiding area, best offer is setting 4 areas, must settin area
HI_BOOL bExpress; // Appear or not, HI_TRUE is appear. HI_FALSE is dispear
HI_U32 u32X; //X coordinate
HI_U32 u32Y; //Y coordinate
HI_U32 u32Width; // width
HI_U32 u32Height; // Height
HI_U32 u32Color; // color (decimal system)）
} HI_S_COVER_PARAM;
```

Example：

```
HI_S_COVER_PARAM sCover;
sCover.u32Area = HI_NET_DEV_COVER_AREA_1; // must setting area
HI_SDK_GetConfig ( lHandle,
HI_CMD_COVER_PARAM,
&sCover,
Sizeof(HI_S_COVER_PARAM));
```

7、HI_CMD_OSDEX_PARAM

Set OSD coordinates

```
#define HI_OSD_TIME 0 // time area
#define HI_OSD_NAME 1 // name area
typedef struct HI_OSD_EX
{
HI_U32 u32Area; // area type. Must setting area
HI_U32 u32X; //X coordinate
HI_U32 u32Y; //Y coordinate
} HI_S_OSD_EX;
Example：
HI_S_OSD_EX sOsdEx;
sOsdEx.u32Area = HI_OSD_TIME; // must setting area
HI_SDK_GetConfig ( lHandle,
HI_CMD_OSDEX_PARAM,
&sOsdEx,
Sizeof(sOsdEx));
```

1.5 preview decode quanlity control

HI_SDK_SetPlayerBufNumber

setting network delay and play reading fluency can through this port adjustment

```
HI_S32   HI_SDK_SetPlayerBufNumbe r(
        HI_HANDLE       lHandle,
        HI_S32          s32BufNum
);
```

Parameters

lHandle

[IN] Handle operate

s32BufNum

[IN] setting video play buffer zone largest frame number, short-cut process area(high definition (0-20) normal (0-50),SDK default frame buffer Is 0

Return Values

Success- HI_SUCCESS， Fail- error code

Remarks

s32RBNum greater the flow of play Chang, the better, the relative delay on the large; s32RBNum smaller the value, playback delay is small, but the time when the network is not smooth

Designate, will be dropped frames, smooth playback of effects.

1.6 PTZ control

Whether the camera supports PTZ properties, you can get HI_GET_PRODUCT_VENDOR in sProduct, PTZ control mode, the device containing the Z0 field is not supported.

HI_SDK_PTZControl

PTZ control mode, the device containing the Z0 field is not supported.

```
HI_S32    HI_SDK_PTZControl (
        HI_HANDLE        lHandle,
        HI_U32           u32Command,
        HI_U32           u32Speed
);
```

Parameters

lHandle

[IN] Handle operate

u32Command

[IN] PTZ control order

Macro definition	**Macro value**	**Definitionr**
HI_CTRL_PTZ_STOP	0x3000	stop PTZ
HI_CTRL_PTZ_UP	0x3001	PTZ up
HI_CTRL_PTZ_DOWN	0x3002	PTZ down
HI_CTRL_PTZ_LEFT	0x3003	PTZ left
HI_CTRL_PTZ_RIGHT	0x3004	ptz right
HI_CTRL_PTZ_ZOOMIN	0x3005	ptz zoom in
HI_CTRL_PTZ_ZOOMOUT	0x3006	ptz zoom out
HI_CTRL_PTZ_FOCUSIN	0x3007	focus in
HI_CTRL_PTZ_FOCUSOUT	0x3008	focus out
HI_CTRL_PTZ_APERTUREIN	0x3009	aperture in
HI_CTRL_PTZ_APERTUREOUT	0x3010	aperture out
HI_CTRL_PTZ_LIGHT_ON	0x3021	light on
HI_CTRL_PTZ_LIGHT_OFF	0x3022	light off
HI_CTRL_PTZ_WIPER_ON	0x3023	wiper on
HI_CTRL_PTZ_WIPER_OFF	0x3024	wiper out
HI_CTRL_PTZ_AUTO_ON	0x3025	auto on
HI_CTRL_PTZ_AUTO_OFF	0x3026	auto off
HI_CTRL_PTZ_HOME	0x3027	come back to home
HI_CTRL_PTZ_CRUISE_V	0x3028	PTZ cruise UP &DOWN
HI_CTRL_PTZ_CRUISE_H	0x3029	Ptz cruise right&left

u32Speed

[IN] 速度

```
#define HI_CTRL_PTZ_SPEED_MAX  0x3F // max speed
#define HI_CTRL_PTZ_SPEED_MIN   0x00 ///mini speed
```

Return Values

PTZ control to send commands no return value.

Remarks

Z by Vendor ID field to determine whether support of the property.

HI_S_ProductVendor in sProduct value.

HI_SDK_PTZControlEx

PTZ control extension, single-step execution.

```
HI_S32    HI_SDK_PTZControlEx (
        HI_HANDLE        lHandle,
        HI_U32           u32Command,
);
```

Parameters

lHandle

[IN] Handle operate

u32Command

[IN] PTZ control order

Macro definition	Macro value	Definitionr
HI_CTRL_PTZ_STOP	0x3000	stop PTZ
HI_CTRL_PTZ_UP	0x3001	PTZ up
HI_CTRL_PTZ_DOWN	0x3002	PTZ down
HI_CTRL_PTZ_LEFT	0x3003	PTZ left
HI_CTRL_PTZ_RIGHT	0x3004	ptz right
HI_CTRL_PTZ_ZOOMIN	0x3005	ptz zoom in
HI_CTRL_PTZ_ZOOMOUT	0x3006	ptz zoom out
HI_CTRL_PTZ_FOCUSIN	0x3007	focus in
HI_CTRL_PTZ_FOCUSOUT	0x3008	focus out
HI_CTRL_PTZ_APERTUREIN	0x3009	aperture in
HI_CTRL_PTZ_APERTUREOUT	0x3010	aperture out

Return Values

PTZ control to send commands no return value.

Remarks

PTZ control for single-step expansion of single-step move.

HI_SDK_PTZPreset

PTZ preset point operations

```
HI_S32    HI_SDK_PTZPreset (
        HI_HANDLE        lHandle,
        HI_U32           u32Command,
        HI_U32           u32Preset
);
```

Parameters

lHandle

[IN] Handle operate

u32Command

[IN] PTZ preset point operations

Macro definition	**Macro value**	**Definitionr**
HI_CTRL_PTZ_GOTO_PRESET	0x3015	Turn to Preset Points
HI_CTRL_PTZ_SET_PRESET	0x3016	Setting Preset Points
HI_CTRL_PTZ_CLE_PRESET	0x3017	Cancel Preset Points

u32Preset

[IN] present point

```
#define HI_CTRL_PTZ_PRESET_MAX  255
#define HI_CTRL_PTZ_PRESET_MIN   0
```

Return Values

PTZ control to send commands no return value.

Remarks

Z by Vendor ID field to determine whether support of the property. HI_S_ProductVendor in sProduct value.

HI_SDK_TransPTZ

Transparent PTZ operation

```
HI_S32   HI_SDK_TransPTZ (
    HI_HANDLE       lHandle,
    HI_CHAR*        psBuf,
    HI_U32          u32BufLen
);
```

Parameters

lHandle

[IN] Handle operate

psBuf

[IN] PTZ control command data, command data can only be 64 bytes string, such as ff01100800041d。

u32BufLen

[IN] PTZ control code length，

```
#define HI_CTRL_PTZ_FT_BUF_LEN   64
```

Return Values

PTZ control to send commands no return value。

Remarks

Pass-through function PTZ control through 845, only send data not receive data through the different PTZ control equipment is not the same pass code, access the equipment through the pass code to see the device-dependent instructions.

Z by Vendor ID field to determine whether support of the property. HI_S_ProductVendor in sProduct value.

1.7 Real-time preview callback data

HI_SDK_SetRealDataCallBack

Registration code stream data callback, registration will not decode display SDK

```
HI_S32   HI_SDK_SetRealDataCallBack (
        HI_HANDLE            lHandle,
        HI_U32               u32Chn,
        OnRealDataCallBack   streamCallBack,
        HI_VOID*             pUserData
);
```

Parameters

lHandle

[IN] Handle operate

u32Chn

[IN] Shaping parameters

streamCallBack

[IN] Data stream callback function

pUserData

[IN] user data

Callback Function

```
typedef   HI_S32   (*OnRealDataCallBack)(
                HI_U32           u32Chn,
                MEDIA_TYPE_E         eStreamType,
                HI_VOID*         pStreamData,
                HI_S32           s32DataNum,
                HI_U32           s32Pts,
                HI_S32           s32KeyFrame,
                HI_VOID*         pUserData
                );
```

Callback Function Parameters

u32Chn

Shaping parameters

eStreamType

Data type, audio and video data or header data

Macro definition	Macro value	Definitionr
HI_AV_DATA	0	voice data
HI_SYS_DATA	1	file data

pStreamData

Data contains header

s32DataNum

Data length

s32Pts

Time stamp

s32KeyFrame

Video key frame 1-I,2-P frame frame

pUserData

user data

Return Values

Success- HI_SUCCESS, Fail- error code

Remarks

1. connected to the first packet of HI_SYS_DATA type.
2. if pu8Buffer data HI_SYS_DATA, pu8Buffer structure is the structure by the HI_S_SysHeader

Composed of:

```
typedef struct
{
    HI_U32    u32Width;           // video Witch
    HI_U32    u32Height;          // video Height
} HI_S_VideoHeader;

typedef struct
{
     HI_U32  u32Format;           // voice format
} HI_S_AudioHeader;
```

Macro definition	Macro value	Definitionr
HI_AUDIO_TYPE_G711	0	G711
HI_AUDIO_TYPE_G726	1	G726
HI_AUDIO_TYPE_AMR	2	AMR

```
typedef struct
{
    HI_U32              u32SysFlag;
    HI_S_VideoHeader     struVHeader;
    HI_S_AudioHeader      struAHeader;
} HI_S_SysHeader;
```

Which defines a macro u32SysFlag #define HI_SYS_FLAG 0x53565848。

3.if pu8Buffer data HI_AV_DATA, pu8Buffer the header structure by HI_S_SysHeader Composed of:

```
typedef struct
{
HI_U32 u32AVFrameFlag; // Frame signs
HI_U32 u32AVFrameLen; // Frame length
HI_U32 u32AVFramePTS; // Time stamp
HI_U32 u32VFrameType; // Video type, I-frames or P frames
} HI_S_AVFrame;
```

u32AVFrameFlag, format as following excel

Macro definition	**Macro value**	**Definitionr**
HI_VIDEO_FRAME_FLAG	0x46565848	voice data
HI_AUDIO_FRAME_FLAG	0x46415848	file data

u32VFrameType format as following excel：

Macro definition	**Macro value**	**Definitionr**
HI_VIDEO_FRAME_I	1	I
HI_VIDEO_FRAME_P	2	P

HI_SDK_SetDecCallBack

Decode callback data registered

```
HI_S32    HI_SDK_SetDecCallBack (
        HI_HANDLE               lHandle,
        HI_U32                  u32Chn
        OnDecCallBack           CallBack,
        HI_VOID*                pUserData
);
```

Parameters

lHandle

[IN] Handle operate

u32Chn

[IN] Shaping parameters

CallBack

[IN] Callback function to decode the data

pUserData

[IN] user data

Callback Function

```
typedef   LONG   (*OnDecCallBack)(
            HI_U32                  u32Chn,
            const FRAME_INFO_S *pFrameInfo,
            HI_VOID *pUserData
            );
```

Callback Function Parameters

u32Chn

Shaping parameters

pFrameInfo

Frame Type

```
typedef struct hiFRAME_INFO_S
{
      HI_U8* pY;          // Y component video data decoded
      HI_U8* pU;          // U-component video data decoded
      HI_U8* pV;          // Decoded video data V components
      long nWidth;        // Video width
      long nHeight;       // Video High
      long nType;         //data type:YUV420
      long nYPich;
      long nUVPich;
      HI_U64 u64Pts;
}
FRAME_INFO_S;
```

pData

Decode the data callback

pUserData

user data

Return Values

Success- HI_SUCCESS, Fail-error code

HI_SDK_SetMessageCallBack

Callback data registered alarm information

```
HI_S32   HI_SDK_SetMessageCallBack (
      HI_HANDLE              lHandle,
      HI_U32                 u32Chn
      OnMessageCallBack  CallBack,
      HI_VOID *              pUserData
);
```

Parameters

lHandle

[IN] Handle operate

u32Chn

[IN] Shaping parameters

CallBack

[IN] Alarm information data callback function

pUserData

[IN] user data

Callback Function

```
typedef   LONG   (*OnMessageCallBack)(
        HI_U32            u32Chn,
        MD_TYPE_E         eDataType,
        HI_U8*            pu8Buffer,
        HI_U32            u32Length,
        HI_VOID*          pUserData
        );
```

Callback Function Parameters

u32Chn

Shaping parameters

eDataType

Data type

Macro definition	**Macro value**	**Definitionr**
HI_MOTION_DETECTION	0	Motion detection alarm
HI_INPUT_ALARM	1	Input Alarm

pu8Buffer

The data. If HI_MOTION_DETECTION， data storage structure will HI_S_ALARM_MD:

```
typedef struct
{
    HI_U32        u32Area;                // area
    HI_U32        u32X;            //x coordinate
    HI_U32        u32Y;            //y coordinate
    HI_U32        u32Width;        // rectangle width
    HI_U32        u32Height;       // rectangle high
} HI_S_ALARM_MD;
```

u32Area up to 4, as follows:

Macro definition	**Macro value**	**Definitionr**
HI_MOTION_AREA_1	1	Area1
HI_MOTION_AREA_2	2	Area2
HI_MOTION_AREA_3	3	Area3

HI_MOTION_AREA_4	4	Area4

u32Length

Data length，HI_MOTION_DETECTION，while there are two regions:

u32Length = 2*sizeof(HI_S_ALARM_MD)

u32DataType

user data

Return Values

Success- HI_SUCCESS, Fail- error code

HI_SDK_SetEventCallBack

Callback event data

```
HI_S32   HI_SDK_SetEventCallBack (
        HI_HANDLE               lHandle,
        HI_U32                  u32Chn,
        OnEventCallBack                 eventCallBack,
        HI_VOID*                pUserData
);
```

Parameters

lHandle

[IN] Handle operate

u32Chn

[IN] Shaping parameters

eventCallBack

[IN] Callback event data

pUserData

[IN] user data

Callback Function

```
typedef   LONG   (*OnEventCallBack) (
                HI_U32                  u32Chn,
                EVENT_TYPE_E                    eEventType,
                HI_VOID*                pEventData,
                HI_S32                  s32DataNum,
                HI_VOID*                pUserData
                );
```

Callback Function Parameters

u32Chn

Shaping parameters

eEventType

Event Type

Macro definition	Macro	Definitionr

	value	
EVENT_LIVE_STOP	0	stop timing preview
EVENT_LIVE_PAUSE	1	time-out timing preview
EVENT_LIVE_PLAY	2	timing preview
EVENT_TALK_STOP	3	stop talk-back
EVENT_TALK_PLAY	4	start talk-back
EVENT_TALK_ABNORM	5	talkback unusual
EVENT_REC_STOP	6	stop video
EVENT_REC_PLAY	7	start video
EVENT_REC_ABNORM	8	video unusual
EVENT_PLAYBACK_READ	9	Ready for playback
EVENT_PLAYBACK_PLAY	10	Start playback
EVENT_PLAYBACK_PAUSE	11	Pause playback
EVENT_PLAYBACK_STOP	12	To stop playback
EVENT_NET_CONNECTING	13	Connecting
EVENT_NET_CONNECTED	14	Successful connection
EVENT_NET_DISCONNECT	15	Connection failed
EVENT_NET_ABNORMAL	16	Abnormal disconnect
EVENT_NET_RECONNECT	17	Reconnect
EVENT_NET_CONNECTFAIL	18	Connection failed
EVENT_REALDATA_STOP	19	Real-time data capture
EVENT_REALDATA_PLAY	20	Stop the capture data

pEventData

Event Type

s32DataNum

Length of event data

pUserData

User data

Return Values

Successful return HI_SUCCESS, failure to return an error code.

Remarks

Event callback thread in a thread with the network which, if the process window messages to the message window with WINDOWS
Mechanisms, such as POSTMESSAGE etc.; other threads can handle.

1.8 preview voice control

HI_SDK_SetVolume

Set the volume size

```
HI_S32   HI_SDK_SetVolume (
        HI_HANDLE            lHandle,
        AUDIO_DIRECT_E       eDir,
        HI_S32               s32Volume
);
```

Parameters

u32Handle

[IN] Handle operate

eDir

[IN] AUDIO_OUT output audio，AUDIO_IN audio input (MIC)

s32Volume

[IN] Audio size range [0,100]

Return Values

Success- HI_SUCCESS,Fail- error code

HI_SDK_GetVolume

Get the current volume

```
HI_S32   HI_SDK_GetVolume (
        HI_HANDLE            lHandle,
        AUDIO_DIRECT_E       eDir,
        HI_S32*              pVolume
);
```

Parameters

u32Handle

[IN] Handle operate

eDir

[IN] AUDIO_OUT output audio，AUDIO_IN audio input (MIC)

pVolume

[OUT] Audio size range [0,100]

Return Values

Success- HI_SUCCESS,Fail- error code

HI_SDK_SetMute

Set Mute / monitor mode

```
HI_S32   HI_SDK_SetMute (
        HI_HANDLE            lHandle,
        AUDIO_DIRECT_E    eDir,
        AUDIO_MUTE_E      eMute
);
```

Parameters

u32Handle

[IN] Handle operate

eDir

[IN] AUDIO_OUT output audio，AUDIO_IN audio input (MIC)

eMute

[IN] AUDIO_MUTE_ON muted，AUDIO_MUTE_OFF listening state

Return Values

Success- HI_SUCCESS,Fail- error code

HI_SDK_GetMute

Get Mute / monitor mode

```
HI_S32    HI_SDK_GetMute (
        HI_HANDLE         lHandle,
        AUDIO_DIRECT_E    eDir,
        AUDIO_MUTE_E*     pMute
);
```

Parameters

u32Handle

[IN] Handle operate

eDir

[IN] AUDIO_OUT output audio，AUDIO_IN audio input (MIC)

pMute

[OUT] AUDIO_MUTE_ON muted，AUDIO_MUTE_OFF listening state

Return Values

Success- HI_SUCCESS,Fail- error code

1.9 Record

HI_SDK_StartRecord

Start recording, video supports two formats: ASF, and custom composite video stream through the interface parameters eFileFormat Control record type.

```
HI_S32    HI_SDK_StartRecord (
        HI_HANDLE         lHandle,
        HI_CHAR *         pFilePath,
        FILE_FORMAT_E     eFileFormat,
        MEDIA_TYPE_E          eFlag,
        HI_S32                  s32FileTime
    );
```

Parameters

lHandle

[IN] Handle operate

pFilePath

[IN] record file path

eFileFormat

[IN] File format, currently supports AVI (FILE_FORMAT_AVI) video format, SF (FILE_FORMAT_ASF) video format and composite flow (FILE_FORMAT_NUDE_STREAM) video format.

eFlag

[IN] The form of video, audio, video, audio and video, reference enumeration MEDIA_TYPE_E

s32FileTime

[IN] Recording the length of time, in seconds, default is 0, 0 for no limit.

Return Values

Success- HI_SUCCESS,Fail- error code

Remarks

Composite stream video: real-time data capture, in order to save the file, the file format in the previous section contains a HI_S_SysHeader structure of the file header, followed by HI_S_AVFrame structure, save the data block size, type and other information, and is data block, the length of the value defined by HI_S_AVFrame data block size. Structured as follows:

HI_S_SysHeader
HI_S_AVFrame
Block
HI_S_AVFrame
Block
....
HI_S_AVFrame
Block
HI_S_AVFrame
Block

The data can be saved in the SDK or player library function HI_SDK_Playback provided HI_PLAYER_OpenFile interface to play.

HI_SDK_StopRecord

stop recording

```
    HI_S32    HI_SDK_StopRecord (
        HI_HANDLE               lHandle
    );
```

Parameters

lHandle

[IN] Handle operate

Return Values

Success- HI_SUCCESS,Fail- error code

1.10 Snapshot

HI_SDK_CapturePicture

Capture BMP plans, including real-time preview and playback files

```
    HI_S32    HI_SDK_CapturePicture (
        HI_U32          u32Handle,
        HI_CHAR*     pszFilePath
    );
```

Parameters

u32Handle

[IN] Handle operate

pszFilePath

[IN] snapshot path

Return Values

Successful return HI_SUCCESS, failure to return an error code

HI_SDK_CaptureJPEGPicture

Capture JPG map, including real-time preview and playback files

```
    HI_S32    HI_SDK_CaptureJPEGPicture (
        HI_HANDLE        lHandle,
        HI_CHAR*            sFilePath
    );
```

Parameters

lHandle

[IN] Handle operate

sFilePath

[IN] snapshot path

Return Values

Successful return HI_SUCCESS, failure to return an error code

HI_SDK_SnapJpeg

network snapshot

```
HI_S32    HI_SDK_SnapJpeg (
HI_HANDLE        lHandle,
HI_U8* pu8Data,
HI_S32 s32BufLen,
HI_S32 *pSize
);
```

Parameters

lHandle

[IN] Handle operate

pu8Data

[IN] Memory data, JPG format

s32BufLen

[IN] Application memory data length, not less than 1,024 bytes

pSize

[IN] Return data size

Return Values

Successful return HI_SUCCESS, failure to return an error code.

Remarks

Network to achieve crawl the web to capture images in JPG format to save the data into memory, interface and log (HI_SDK_Login) can be successfully used in the external memory for the application, the application memory size can not be less than:

```
#define HI_SDK_SNAP_BUF_LEN_MIN 1024
```

Used as follows:

```
char *sData = (char*)malloc(1024*1024);
int nSize = 0;
s32Ret = HI_SDK_SnapJpeg(m_lHandle, (HI_U8*)sData, 1024*1024, &nSize);
if(s32Ret == HI_SUCCESS)
{
FILE *fp = fopen("D:\\photo.jpg", "wb+");
if( !fp )
free(sData);
fwrite((const char*)sData, 1, nSize, fp);
fclose( fp );
}
free(sData);
sData = NULL;
```

1.11 image overlay display

HI_SDK_InputDrawData

Add to overlay the image information, type

```
HI_S32   HI_SDK_InputDrawData (
         HI_HANDLE            lHandle,
         DRAW_INFO_S*              pstrDrawData,
         HI_S32               s32StrSize,
         HI_S32               s32DrawState
);
```

Parameters

lHandle

[IN] Handle operate

pstrDrawData

[IN] information buffer

s32StrSize

[IN] Buffer size information

s32DrawState

[IN] Display type, DRAW_STATE and EVENT_STATE two types

Return Values

Successful return HI_SUCCESS, failure to return an error code.

HI_SDK_ClearDrawData

Clear overlay image information specified

```
HI_S32   HI_SDK_ClearDrawData (
         HI_HANDLE       lHandle,
         HI_CHAR*        pDrawData,
         HI_S32          s32DrawState
);
```

Parameters

lHandle

[IN] Handle operate

pDrawData

[IN] information buffer

s32DrawState

[IN] Display type, DRAW_STATE and EVENT_STATE two types

Return Values

Successful return HI_SUCCESS, failure to return an error code.

HI_SDK_SelectPic

Set the location of the mouse focus, the callback function call DRAW superimposed image processing

```
HI_S32   HI_SDK_SelectPic (
         HI_HANDLE       lHandle,
        CPoint              point
    );
```

Parameters

lHandle

[IN] Handle operate

point

[IN] The current coordinates of the mouse

Return Values

Successful return HI_SUCCESS, failure to return an error code.

HI_SDK_MouseMove

This function is called when the mouse moves, DRAW callback function call update MD area coordinates

```
    HI_S32   HI_SDK_MouseMove (
        HI_HANDLE           lHandle,
        UINT                nFlags,
        CPoint              point,
        CRect               rcRect
    );
```

Parameters

lHandle

[IN] Handle operate

nFlags

[IN] Buttons mark

point

[IN] The current coordinates of the mouse

rcRect

[IN] window coordinates

Return Values

Successful return HI_SUCCESS, failure to return an error code.

HI_SDK_SetDrawCallBack

Registered drawing callback, when the mouse to modify the MD coordinate information, call the callback function to update the MD properties

```
    HI_S32   HI_SDK_SetDrawCallBack (
        HI_HANDLE               lHandle,
        HI_U32                  u32Chn,
        OnDrawCallBack              callBack,
        HI_VOID*                pUserData
    );
```

Parameters

lHandle

[IN] Handle operate

u32Chn

[IN] Shaping parameters

OnDrawCallBack

[IN] Callback event data

pUserData

[IN] user data

Callback Function

```
typedef   LONG   (*OnDrawCallBack) (
          HI_U32        u32Chn,
          RECT          rcDrawRect,
          HI_CHAR*      pszName,
          HI_VOID*      pUserData
          );
```

Callback Function Parameters

u32Chn

Shaping parameters

rcDrawRect

Icon on the new coordinate

pszName

The name of the icon

pUserData

user data

Return Values

Successful return HI_SUCCESS, failure to return an error code.

HI_SDK_EnablePic

Superimposed images express hidden

```
HI_S32   HI_SDK_EnablePic (
     HI_HANDLE        lHandle,
     HI_CHAR*         pszName,
     HI_S32           s32EnableValue,
     HI_S32           s32DrawState
);
```

Parameters

lHandle

[IN] Handle operate

pszName

[IN] name

s32EnableValue

[IN] Express Hidden, 0 hidden and 1 to express

s32DrawState

[IN] Display type, DRAW_STATE and EVENT_STATE two types

Return Values

Successful return HI_SUCCESS, failure to return an error code.

HI_SDK_GetPicInfo

Get image height and width

```
HI_S32    HI_SDK_GetPicInfo (
        HI_HANDLE        lHandle,
        HI_S32*          pHeight,
        HI_S32*          pWidth
);
```

Parameters

lHandle

[IN] Handle operate

pHeight

[OUT] Height

pWidth

[OUT] width

Return Values

Successful return HI_SUCCESS, failure to return an error code.

1.12 voice talkback transmit

HI_SDK_StartVoiceCom

Open voice talkback

```
HI_S32    HI_SDK_StartVoiceCom (
        HI_HANDLE              lHandle,
        HI_U32                 u32Chn,
        OnVoiceDataCallBack       callback,
        HI_VOID *              pUserData
);
```

Parameters

lHandle

[IN] Handle operate

u32Chn

[IN] INT parameter

Callback

[IN] Voice talkback，the default value is NULL

pUserData

[IN] user data, the default value is NULL

Callback Function

```
typedef   LONG   (*OnVoiceDataCallBack) (
                HI_U32 u32Chn,
                HI_U8* pBuf,
                HI_S32 s32Size,
                HI_U32 u32TimeStamp,
                HI_VOID *pUserData
                );
```

Callback Function Parameters

u32Chn

Int parameter

pBuf

voice data

s32Size

voice data size

u32TimeStamp

timestamp

pUserData

user data

Return Values

Return HI_SUCCESS if sucess，or else return error code.

Remarks

If Callback is not null, SDK cannot send voice data to camera,the voice data can be sent through HI_SDK_VoiceComSendData function.

HI_SDK_StopVoiceCom

Close voice talkback

```
HI_S32   HI_SDK_StopVoiceCom (
        HI_HANDLE        lHandle,
);
```

Parameters

lHandle

[IN] Handle operate

Return Values

Return HI_SUCCESS if success, or else return error code.

HI_SDK_VoiceComSendData

Send the collected data back to each other

```
HI_S32   HI_SDK_VoiceComSendData (
        HI_HANDLE        lHandle,
        HI_CHAR*         psBuf,
        HI_U32           u32BufLen,
        HI_U64           u64Pts
);
```

Parameters

lHandle

[IN] Handle operate

psBuf

[IN] send data

u32BufLen

[IN] data size

U64Pts

[IN] timestamp

Return Values

Return HI_SUCCESS if success, or else return error code.

Remarks

The collected data of voice talk back must be 8K, 16 bit, mono channel G726 compressed data, pls read the usage of Demo.about details.

1.12 Record playback

HI_SDK_Playback

Playback

```
HI_HANDLE   HI_SDK_Playback (
        HI_CHAR*         psFilePath,
        HI_VOID*         pWnd
);
```

Parameters

psFilePath

[IN] File path

pWnd

[IN] playback window handle

Return Values

Return Handle operateHI_HANDLE if success, or else return zero.

HI_SDK_StopPlayback

Close playback

```
HI_S32    HI_SDK_StopPlayback (
        HI_HANDLE        lPlayHandle
);
```

Parameters

lPlayHandle

[IN] Return Handle operate of HI_SDK_Playback

Return Values

Return HI_SUCCESS if success, or else return error code.

HI_SDK_PlayBackControl

Playback control

```
HI_S32    HI_SDK_PlayBackControl (
        HI_HANDLE            lPlayHandle,
        PBCTRL_TYPE_E        s32Command,
        HI_S32               s32Value,
        HI_S32               *s32OutValue
);
```

Parameters

lPlayHandle

[IN]   Handle operate of HI_SDK_Playback

s32Command

[IN] Command operation

Definition	**Value**	**Definition**
PB_CTRL_PLAY	0	play
PB_CTRL_STOP	1	stop
PB_CTRL_PAUSE	2	pause
PB_CTRL_RATE	3	Adjust speed
PB_CTRL_FRAME	4	Single frame
PB_CTRL_SETPOS	5	Locate play
PB_CTRL_GETPOS	6	Get play position
PB_CTRL_MUTE	7	Mute/monitor
PB_CTRL_VOLUME	8	Set volume
PB_CTRL_GETTIME	9	Get play time

s32Value

[IN] set operation value

s32OutValue

[OUT] obtain the value of operation

Return Values

Return HI_SUCCESS if success, or else return error code.

1.13 Decoding operation

HI_SDK_PauseDecode

Pause decoding, video cannot be display

```
HI_S32    HI_SDK_PauseDecode (
          HI_HANDLE       lHandle
);
```

Parameters

lHandle

[IN] Handle operate

Return Values

Return HI_SUCCESS if success, or else return error code.

HI_SDK_ResumeDecode

Resume decoding from the first one frame

```
HI_S32    HI_SDK_ResumeDecode (
          HI_HANDLE       lHandle
);
```

Parameters

lHandle

[IN] Handle operate

Return Values

Return HI_SUCCESS if success, or else return error code.

1.14 Other

HI_SDK_GetSDKVersion

Get the version of SDK

```
HI_S32    HI_SDK_GetSDKVersion (
          HI_CHAR*    pVersion
);
```

Parameters

pVersion

[OUT] SDK version

Return Values

Return HI_SUCCESS if success, or else return error code.

HI_SDK_GetPlayRate

Get average bit rateof preview play

```
HI_S32   HI_SDK_GetPlayRate (
        HI_HANDLE        lHandle,
        HI_S32           *pFrameRate,
        HI_S32           *pBitRate
);
```

Parameters

lHandle

[IN] Handle operate

pFrameRate

[OUT] FrameData

pBitRate

[OUT] BitRate

Return Values

Return HI_SUCCESS if success, or else return error code.

HI_SDK_GetState

Get play, voice talkback, record state

```
HI_S32   HI_SDK_GetState (
        HI_HANDLE        lHandle,
        STATE_ID_E       eStateID,
        HI_S32 *                 pState
);
```

Parameters

lHandle Get average bit stream frame of preview play

[IN] Handle operate

eStateID

[IN] Type

```
typedef enum hiSTATE_ID_E
{
        STATE_ID_PLAY = 0,                //File or stream playmark
        STATE_ID_REC,                     // Record mark
        STATE_ID_TALK,                    // Voice playback mark
        STATE_ID_SERVER_USERNUM,          // User connection number
        STATE_ID_BUTT
} STATE_ID_E;
```

pState

[OUT] State

STATE_ID_E as following：

1、STATE_ID_PLAY

```
typedef enum hiPLAY_STATE_E
{
    PLAY_STATE_PAUSE = 0,           //Pause
    PLAY_STATE_PLAY,                //Play
    PLAY_STATE_AUDIO,                   //Audio
    PLAY_STATE_VIDEO,                   //Video
    PLAY_STATE_STOP,                //Stop
    PLAY_STATE_BUTT
} PLAY_STATE_E;
```

2、STATE_ID_REC

```
typedef enum hiREC_STATE_E
{
    REC_STATE_RUN    = 0,           //Recording
    REC_STATE_STOP,                 //Stop recording
    REC_STATE_BUTT
} REC_STATE_E;
```

3、STATE_ID_TALK

```
typedef enum hiTALK_STATE_E
{
    TALK_STATE_RUN = 0,             //Begin talking
    TALK_STATE_STOP,                //Stop talking
    TALK_STATE_BUTT
} TALK_STATE_E;
```

Return Values

Return HI_SUCCESS if success, or else return error code.

HI_SDK_GetPlayerHandle

Get playerhandle which contains real-time preview and file playback

```
HI_S32   HI_SDK_GetPlayerHandle (
    HI_HANDLE        lHandle,
    HI_VOID**        ppPlayerHandle
);
```

Parameters

lHandle

[IN] Handle operate

ppPlayerHandle

[OUT] Play library handle

Return Values

Return HI_SUCCESS if success, or else return error code.

HI_SDK_SetDrawWnd

It will display window if you change playing.

```
HI_S32    HI_SDK_SetDrawWnd (
        HI_HANDLE        lHandle,
        HI_VOID*         pWnd
);
```

Parameters

lHandle

[IN] Handle operate

pWnd

[IN] Window handle

Return Values

Return HI_SUCCESS if success, or else return error code.

Remark

If you want to change the current play window to another when playing, you can change port directly, that means connect the display window handle with correspounding Handle operate. If pWnd is null, DDRAW will be destroyed, that means no display video; only when pWnd is not null agagin, the video will ba display agagin.

HI_SDK_GetSupportAttr

Get camera'[s support attribute

```
HI_S32    HI_SDK_GetSupportAttr (
        HI_HANDLE        lHandle,
        HI_S_SUPPORT*pSupport
);
```

Parameters

lHandle

[IN] Handle operate

pSupport

[OUT] HI_S_SUPPORT struct

```
typedef struct tagHI_SUPPORT
{
  HI_U32 u32Operation;          // operation attribute, such as night
  vision effect ,whitebalance
    HI_U32 u32Reslution;        //Main stream suppors resolution
    HI_U32 u32Reslution1;       //Substream supports resolution
      HI_U32 u32FrameMax;           //Max frame
      HI_U32 u32BitRateMin;     //Main stream bit rate min
      HI_U32 u32BitRateMax;     // Main stream bit rate max
HI_U32 u32BitRateMin1;   // Substream bit rate min
```

```
            HI_U32 u32BitRateMax1;  // Substream bit rate max

        }HI_S_SUPPORT;
```

Return Values

Return HI_SUCCESS if success, or else return error code.

Remarks

```
SUPPORTATTR_NIGHTVISION_SET_FLAG      (0x000000001<<1) //Night vision
SUPPORTATTR_WHITEBALANCE_FLAG     (0x000000001<<3) //White balance
SUPPORTATTR_FLIP_FLAG             (0x000000001<<4) //Flip
SUPPORTATTR_MIRROR_FLAG           (0x000000001<<5) //Mirror
SUPPORTATTR_BRIGHTNESS_FLAG       (0x000000001<<6) //Brightness
SUPPORTATTR_SATURATION_FLAG       (0x000000001<<7) //Saturation
SUPPORTATTR_CONTRAST_FLAG             (0x000000001<<8) //Contrast
SUPPORTATTR_HUE_FLAG              (0x000000001<<9) //Hue
SUPPORTATTR_SUBSTREAM_FLAG        (0x000000001<<10) //Substream
SUPPORTATTR_POWERFREQ_FLAG        (0x000000001<<11) //Framerate
```

Example：

```
HI_S_SUPPORT sSupport;
HI_SDK_GetSupportAttr( lHandle, &sSupport );

if( sSupport.u32Operation |= SUPPORTATTR_SUBSTREAM_FLAG )
    // support night vision
if( sSupport.u32Operation |= SUPPORTATTR_FLIP_FLAG )
    //support flip
if( sSupport.u32Operation |= SUPPORTATTR_POWERFREQ_FLAG )
    //support frame setting
… …
if( sSupport.u32Reslution |= (0x00000001<<HI_RESOLUTION_VGA) )
    // mainstream supports VGA
if( sSupport.u32Reslution |= (0x00000001<<HI_RESOLUTION_CIF) )
    //mainstream supports CIF
```

HI_SDK_SetAutoAdjust

Set the display proportion of video

```
HI_S32    HI_SDK_SetAutoAdjust (
    HI_HANDLE        lHandle,
);
```

Parameters

lHandle

[IN] Handle operate

Return Values

Return HI_SUCCESS if success, or else return error code.

HI_SDK_GetAutoAdjust

Get the display proportion of video

```
HI_S32    HI_SDK_GetAutoAdjust (
        HI_HANDLE         lHandle,
);
```

Parameters

lHandle

[IN] Handle operate

Return Values

HI_SUCCESS expresss that the current displaying is auto adjust state, HI_ FAILURE expresss non-auto adjust state

HI_SDK_GetMediaAttr

Get attribute parameter of audio and video

```
HI_S32    HI_SDK_GetMediaAttr (
        HI_HANDLE         lHandle,
        STREAM_ATTR_S *pStreamInfo
);
```

Parameters

lHandle

[IN] Handle operate

pStreamInfo

[OUT] STREAM_ATTR_S struct

```
typedef struct tagPLAYERSDK_ATTR_VIDEO_STREAM_S
{
        PLAYERSDK_VIDEO_FORMAT_E eVEncode;    //video format
        long lHeight;              //video height
        long lWidth;               //video width
        long lBitRate;             //video bit rate
        long lFrameRate;           //video frame rate
}PLAYERSDK_ATTR_VIDEO_STREAM_S;
//audio attr
typedef struct tagPLAYERSDK_ATTR_AUDIO_S
{
        PLAYERSDK_AUDIO_FORMAT_E    eAEncode;    //audio    encode
format
        long lSamplesPerSec;               //audio's samples per second
        long lBitsPerSample;               //bits per sample
```

```
    long lBitRate;              //audio's bit rate
    long lBlockAlign;           //if block align
    long lChannels;             //audio's channels
    long lFrameFlag;            //audio's frame flag
    long length;                //audio's size
    void *pReserved;
}PLAYERSDK_ATTR_AUDIO_S;
//frame image info

typedef struct hiSTREAM_ATTR_S
{
    PLAYERSDK_ATTR_VIDEO_STREAM_S struVAttr;
    PLAYERSDK_ATTR_AUDIO_S        struAAttr;
} STREAM_ATTR_S;
```

Return Values

Return HI_SUCCESS if success, or else return error code.

Remarks

Example：

```
STREAM_ATTR_S struStreamInfo;
HI_SDK_GetMediaAttr(.lHandle, &struStreamInfo);
```

HI_SDK_DisplayAll

Display area will be electronic amplified

```
HI_S32  HI_SDK_DisplayAll (
    HI_HANDLE    lHandle,
    HI_S32       s32Left,
    HI_S32       s32Top,
    HI_S32       s32Right,
    HI_S32       s32Bottom,
    HI_BOOL      bDisplayAll
);
```

Parameters

lHandle

[IN] Handle operate

s32Left

[IN] top-left coordinate（x）

s32Top

[IN] top-left coordinate（y）

s32Right

[IN] bottom-right coordinate（x）

s32Bottom

[IN] bottom-right coordinate（y）

bDisplayAll

[IN] whether to display the entire image, HI_TRUE-display all, HI_FALSE-use area amplification function

Default value is HI_TRUE，you must use HI_FALSE to set display area；

Return Values

Return HI_SUCCESS if success, or else return error code.

Remark

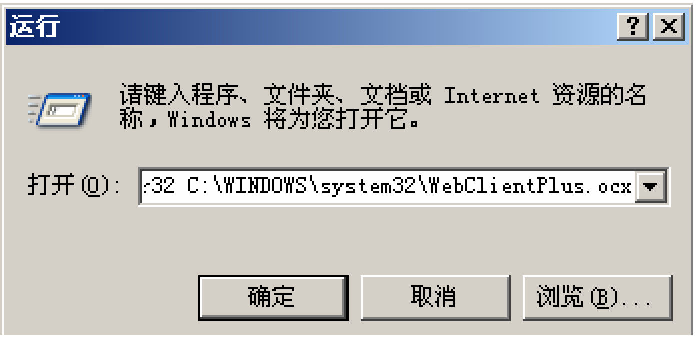
Function within the SDK functions to display dynamic electronic amplification, the input coordinates are relative to window coordinates.

Second part OCX activex port

2.1 Function brief introduction

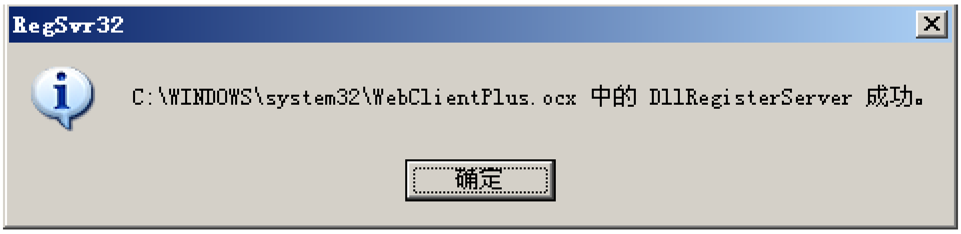
Client OCX provides real-time preview, client snapshot, client record, video parameter display and configuration, movement parameter display and configuration, taikback, PTZ control, local playback and so on.
IE main UI, image parameter configuration and movement parameter UI use the same OCX.activex. Movement parameter configuration UI will callback port SetUseMDPage. You can refer to IE web code callback OCX activex.

Usage:
Register the activex before using, if the activex is download from web, after installed, it have installed. If it is not manually register, you can register by using command regsvr32+, Download area from web and install activex, there is WebClientPlus.OCX area and related library file under C:\WINDOWS\system32. The method of callback activex is insert to enginner contents in the form of groupware(development environment is different, the callback way is different), and then you can use the related port.After engineer development finished, package need choose OCX activex and auto register.
Register OCX manually:



请键入程序、文件夹、文档或 Internet 资源的名称，Windows 将为您打开它。

打开(O)：:32 C:\WINDOWS\system32\WebClientPlus.ocx



C:\WINDOWS\system32\WebClientPlus.ocx 中的 DllRegisterServer 成功。

1、The method of callback OCX in the web:

```
<SCRIPT type=text/JavaScript>
if (navigator.appName.indexOf("Microsoft Internet Explorer") != -1)
{
        document.open();
        document.write('<object                    classid="clsid:42B182F9-3F08-484E-9913-07193A5D36A5"
codebase="WebClientPlus.OCX#version=3,0,1,1" id="DHiMPlayer" align="absbottom" viewastext>');
        document.write('<p align="left" style="font-size:14px">');
        document.write('&nbsp;&nbsp;&nbsp;&nbsp;<span id="t5"> alarm information：</span><br>');
       document.write(' <span id="t6">1. your PC have not installed video activex <br>2. The activex is
    notlatest, pls re-install again. <br><br> Pls click </span><a href="/web/ClientOCXPlus_Setup.exe" id="t7">download activex </a>');
        document.write(' <span id="t8">然后点击</span> <b id="t9"> 运行 </b> <span id="t10"> install acticex，refresh web,browser video。</span></p>');
        document.write('<param name="_Version" value="65536"> <param name="_ExtentX" value="10954"> <param name="_ExtentY" value="6826">');
        document.write('<param name="_StockProps" value="0">');
        document.write('<embed src="65536" _version="65536" _extentx="10954" _extenty="6826" _stockprops="0" align="center" height="0" width="0"> </object>');
        document.close();
}
</SCRIPT>
```

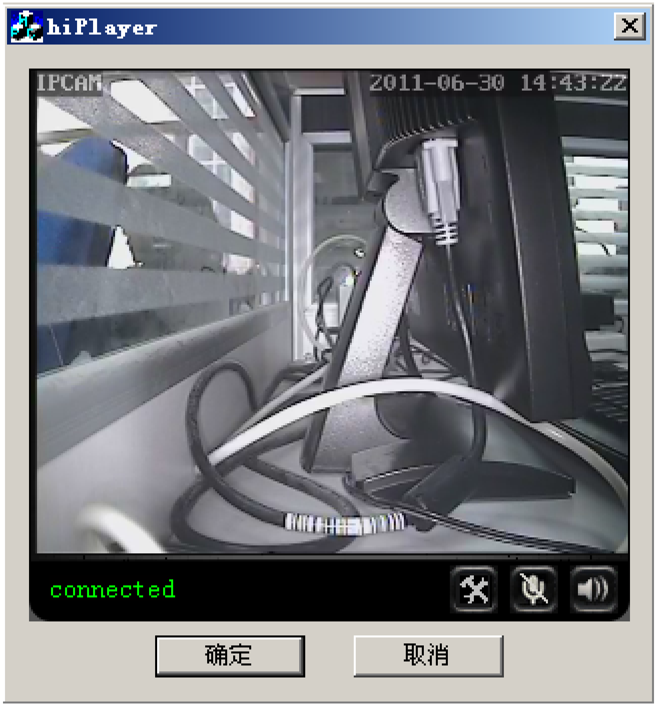
clsid:42B182F9-3F08-484E-9913-07193A5D36A5 is Clsid of OCX;
codebase="WebClientPlus.OCX is the name of OCX;
version=3,0,1,1 OCX version

revoke port:

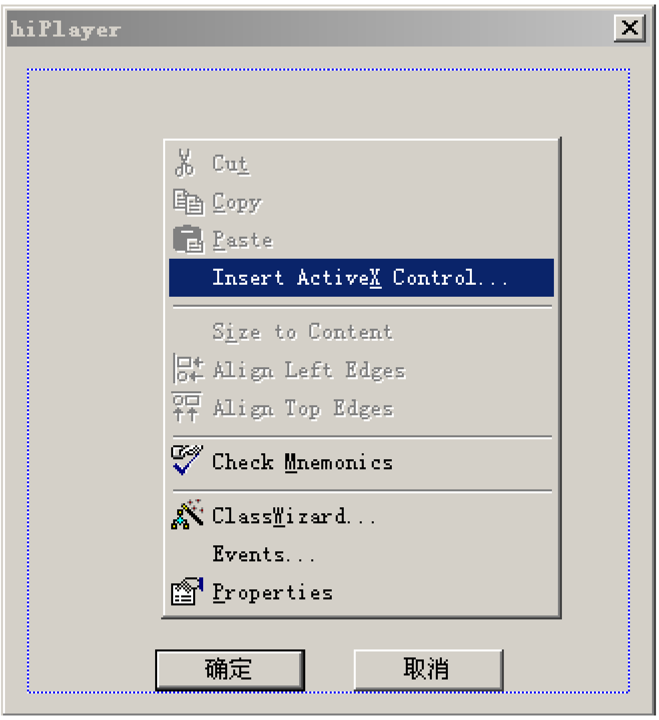
```
    DHiMPlayer.SetUrl(url,80,streamnum,name0,password0);
    DHiMPlayer.SetWndPos(0, 0, w, h);
    DHiMPlayer.Play();
```

2、OCX is used in the develop environment（ take VC++ 6.0 for example）

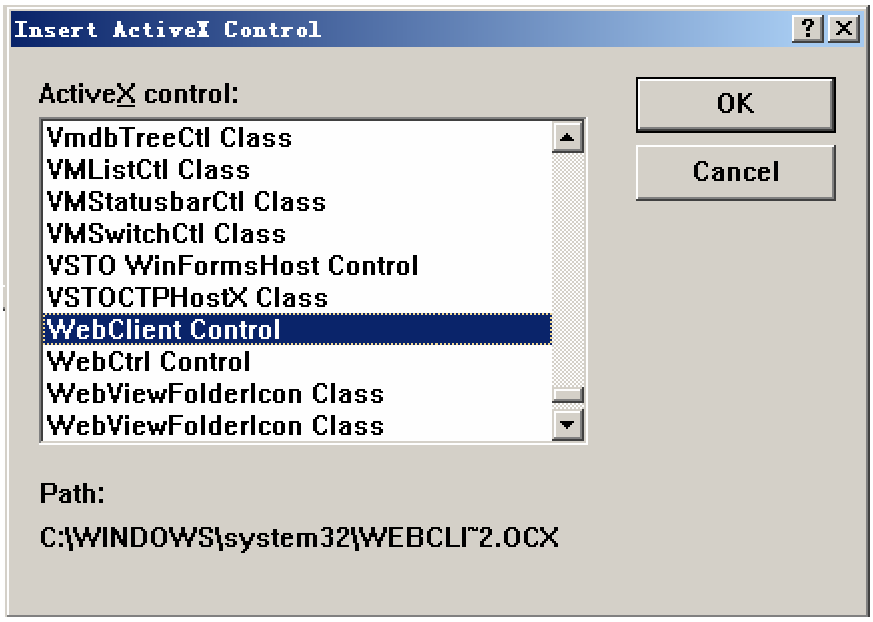
Effect graph:



1) Set a new mfc engineer based on dialog box, named hiPlayer;



2) Right click and choose "Insert ActiveX Control…", pop-up the following secreen Insert ActiveX Control;



Path:
C:\WINDOWS\system32\WEBCLI~2.OCX

3) Choose the registered OCX（ you must register OCX）,OCX will be displayed in the dialog box;

4) Add member var for OCX :CwebClient m_hiPlayer;

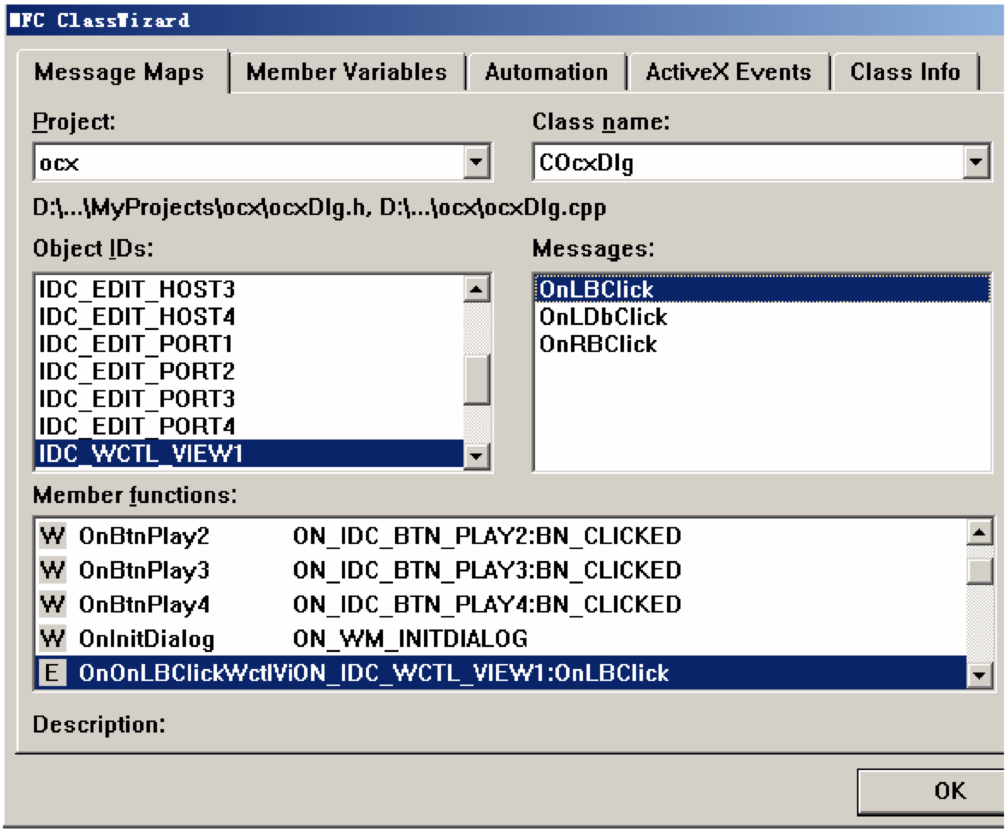
5) Enter the following information in the OnOK button:

```
void CHiPlayerDlg::OnOK()
{
    m_hiPlayer.SetUrl("192.168.1.22", 80, 11, "admin", "admin");
    //Mainstream-11 Substream-12
    m_hiPlayer.Play();
    //CDialog::OnOK();
    }
```

6) Compile and run。

Note: The method of revoking OCX is different in the different development environment,.

7) Click OCX message,choose OCX in MFC ClassWizard，there are three events: OnLBClick（left cick）、OnLDbClick（double click）和 OnRBClick（right click ）, double click Add event and add code in the event.（the version must be above3.0.2.2）



Project: ocx

Class name: COcxDlg

D:\...\MyProjects\ocx\ocxDlg.h, D:\...\ocx\ocxDlg.cpp

Object IDs:

IDC_EDIT_HOST3
IDC_EDIT_HOST4
IDC_EDIT_PORT1
IDC_EDIT_PORT2
IDC_EDIT_PORT3
IDC_EDIT_PORT4
IDC_WCTL_VIEW1

Messages:

OnLBClick
OnLDbClick
OnRBClick

Member functions:

```
W  OnBtnPlay2      ON_IDC_BTN_PLAY2:BN_CLICKED
W  OnBtnPlay3      ON_IDC_BTN_PLAY3:BN_CLICKED
W  OnBtnPlay4      ON_IDC_BTN_PLAY4:BN_CLICKED
W  OnInitDialog    ON_WM_INITDIALOG
E  OnOnLBClickWctlViON_IDC_WCTL_VIEW1:OnLBClick
```

Description:

2.2、revoke sequence

SetWndPos
SetUrl
Play

2.3、Port instruction

2.3.1 Set window positon

Set window position

```
    long    SetWndPos (
        long        lLeft,
        long        lTop,
        long        lRight,
        long        lBottom
    );
```

Parameters

lLeft

[IN] left coordinate

lTop

[IN] top coordinate

lRight

[IN] right coordinate

lBottom

[IN] bottom coordinate

Return Values

HI_SUCCESS expresss success，HI_ FAILURE expresss failure。

2.3.2 SET URL

Set URL

```
long   SetUrl (
        LPCTSTR     sHost,
        long        lPort,
        long        lChn,
        LPCTSTR     sUser,
        LPCTSTR     sPwd
);
```

Parameters

sHost

[IN] Host address

lPort

[IN] Port number

lChn

[IN] stream（11-main stream，12-substream）

sUser

[IN] username

sPwd

[IN] password

Return Values

HI_SUCCESS expresss success，HI_ FAILURE expresss failure.

2.3.3 Connect preview image

Connect preview image

```
long   Play(
);
```

Return Values

HI_SUCCESS expresss success，HI_ FAILURE expresss failure.

2.3.4 Get connection state

Get connection state

```
long    GetPlayState (
);
```

Return Values

Return 3 expresss no audio and video, that is no connection, 2 expresss only have audio but no video, 1 expresss only have video but no audio.

2.3.5 Stop preview

Stop preview

```
long    Stop (
);
```

Return Values

HI_SUCCESS expresss success，HI_ FAILURE expresss failure.

2.3.6 Set mute/monitor

Set mute/ monitor

```
long    Mute (
);
```

Return Values

HI_SUCCESS expresss success，HI_ FAILURE expresss failure.

2.3.7 Get video state

Get video state

```
BOOL    GetMuteState (
);
```

Return Values

HI_SUCCESS expresss mute，HI_ FAILURE expresss monitor.

2.3.8 Start/stop recording

Start/stop recording

```
long    Record (
        long            lMode,
);
```

Parameters

lMode

[IN] unused

Return Values

HI_SUCCESS expresss success，HI_ FAILURE expresss failure.

2.3.9 Get record state

Get record state

```
BOOL   GetRecState (
);
```

Return Values

HI_SUCCESS expresss recording，HI_ FAILURE expresss no recording

2.3.10 Snapshot

Snapshot

```
long   Snapshot (
);
```

Return Values

HI_SUCCESS expresss success, HI_ FAILURE expresss failure.

2.3.11 Set the storage path of record and snapshot

Set the storage path of record and snapshot, recoke port and it will pop-up dialog box.

```
long   SetRecordPath (
);// pop-up path window

long   SetRecordPathEx (
        LPCTSTR     lpStrPath
);// tranmit path
```

Return Values

HI_SUCCESS expresss success, HI_ FAILURE expresss failure.

Note:注：SetRecordPath will pop up path select dialog box,，SetRecordPathEx（OCX mustbe above 3.0.2.2）is the method of transmit path.

2.3.12 Open/close talk

Open/close voice talkback

```
long   Talk (
);
```

Return Values

HI_SUCCESS expresss success, HI_ FAILURE expresss failure.

2.3.13 Get talk state

Get talk state

```
BOOL   GetTalkState (
);
```

Return Values

HI_SUCCESS expresss is talking，HI_ FAILURE expresss stop talking.

2.3.14 Open player

Open player

```
long   PlayBack (
);
```

Return Values

HI_SUCCESS expresss success, HI_ FAILURE expresss failure.

2.3.15 PTZ control

PTZ control

```
long   PtzControl (
       long      lType,
       long      lSpeed
);
```

Parameters

lType

[IN] operation type

value	Definitionr
0	Stop PTZ
1	PTZ upward
2	PTZ downward
3	PTZ turn left
4	PTZ turn right
5	Zoom in
6	Zoom out
7	Open light
8	Close light
9	Open windshield wiper
10	Close windshield wiper

11	Auto open
12	Auto close
13	Zoom in
14	Zoom out
15	Enlarger aperture
16	Smaller aperture

lSpeed

[IN] parameter

Return Values

HI_SUCCESS expresss success, HI_ FAILURE expresss failure.

2.3.16 PTZ preset

PTZ preset

```
long   PTZPreset (
       long      lType,
       long      lPreset
);
```

Parameters

lType

[IN] preset type（0-go to preset, 1-set preset，2-delete preset）

lPreset

[IN] parameter, range [0，255]

Return Values

HI_SUCCESS expresss success, HI_ FAILURE expresss failure.

2.3.17 PTZ transparent transmission

PTZ transparent transmission

```
long   PtzControl (
       LPCTSTR     sCode,
       long        lSize
);
```

Parameters

sCode

[IN] Control PTZ command data, the command data must be consisted by 64bit such as: ff01100800041d.

lSize

[IN] Control PTZ command data size

Return Values

HI_SUCCESS expresss success, HI_ FAILURE expresss failure.

2.3.18 Mouse operate PTZ

Enable/disable OCX mouse operate PTZ function

```
long   SetUsePtzCtrl (
       long      lEnable
);
```

Parameters

lEnable

[IN] Enable/disable OCX mouse operate PTZ function:0- disable,1-enable

Return Values

HI_SUCCESS expresss success, HI_ FAILURE expresss failure.

2.3.19 Open/close motion detection area

Open/close motion detection area

```
long   OpenMDSetPage (
       long      lFlag
);
```

Parameters

lFlag

[IN] 0:normal play,1:motion detection editor state

Return Values

HI_SUCCESS expresss success, HI_ FAILURE expresss failure.

2.3.20 Display/hide edit area

Display/hide edit area

```
long   EnablePic (
       long      s32MDNum,
       long      s32EnableValue,
       long      s32Width,
       long      s32Height,
       long      s32X,
       long      s32Y
);
```

Parameters

s32MDNum

[IN] MD area（1~4）

s32EnableValue

[IN] display hide flag（1-display，2-hide）

s32Width

[IN] MD width

s32Height

[IN] MD heigh

s32X

[IN] MD x coordinate

s32Y

[IN] MD y coordinate

Return Values

HI_SUCCESS expresss success, HI_ FAILURE expresss failure.

Remark

This function takes effect only under this condition:open motion detection area settings.

2.3.21 Get edit area attribute

Get edit area attribute

```
long   GetPic (
       long      s32MDNum,
       long      s32Flag,
);
```

Parameters

s32MDNum

[IN] MD area（1~4）

s32Flag

[IN] get coordinate flag（0-width，1-height，2-x，3-y）

Return Values

Return coordinate value.

2.3.22 Save video stream attribute

Save video stream attribute to configuration file.

```
long   SetStreamNum (
       long      lStreamNum
);
```

Parameters

lStreamNum

[IN] video stream attribute

Return Values

HI_SUCCESS express success, HI_ FAILURE express failure.

2.3.23 Get video stream attribute

Get video stream attribute form configuration file

```
long   GetStreamNum (
);
```

Return Values

11 express main stream, 12 express substream.

2.3.24 Request video stream

Request video stream, camera cannot send video data when playing(it will teke effect after reconnecting thedevice again)

```
long   PauseVideo (
        long      lVideoTag
);
```

Parameters

lVideoTag

[IN] flag，0-request video,1- not request video

Return Values

HI_SUCCESS express success,   HI_ FAILURE express failure.

2.3.25 Request audio stream

Requset audio stream, camera cannot send video data when playing(it will teke effect after reconnecting thedevice again).

```
long   PauseAudio (
        long      lAudioTag
);
```

Parameters

lAudioTag

[IN]   flag, 0-request audio，1-not request audio

Return Values

HI_SUCCESS express success, HI_ FAILURE express failure.

2.3.26 Get display proportion

Get display proportion, 0 express tensile mode,1 express auto adjust proportion.

```
long    GetAutoAdjust (
);
```

Return Values

0 express tensile mode,1 express auto adjust proportion.

2.3.27 Set auto adjust mode

Set display proportion

```
long    SetAutoAdjust (
        long        lType
);
```

Parameters

lType

[IN] proportion mode 0-tensile，1-auto adjust

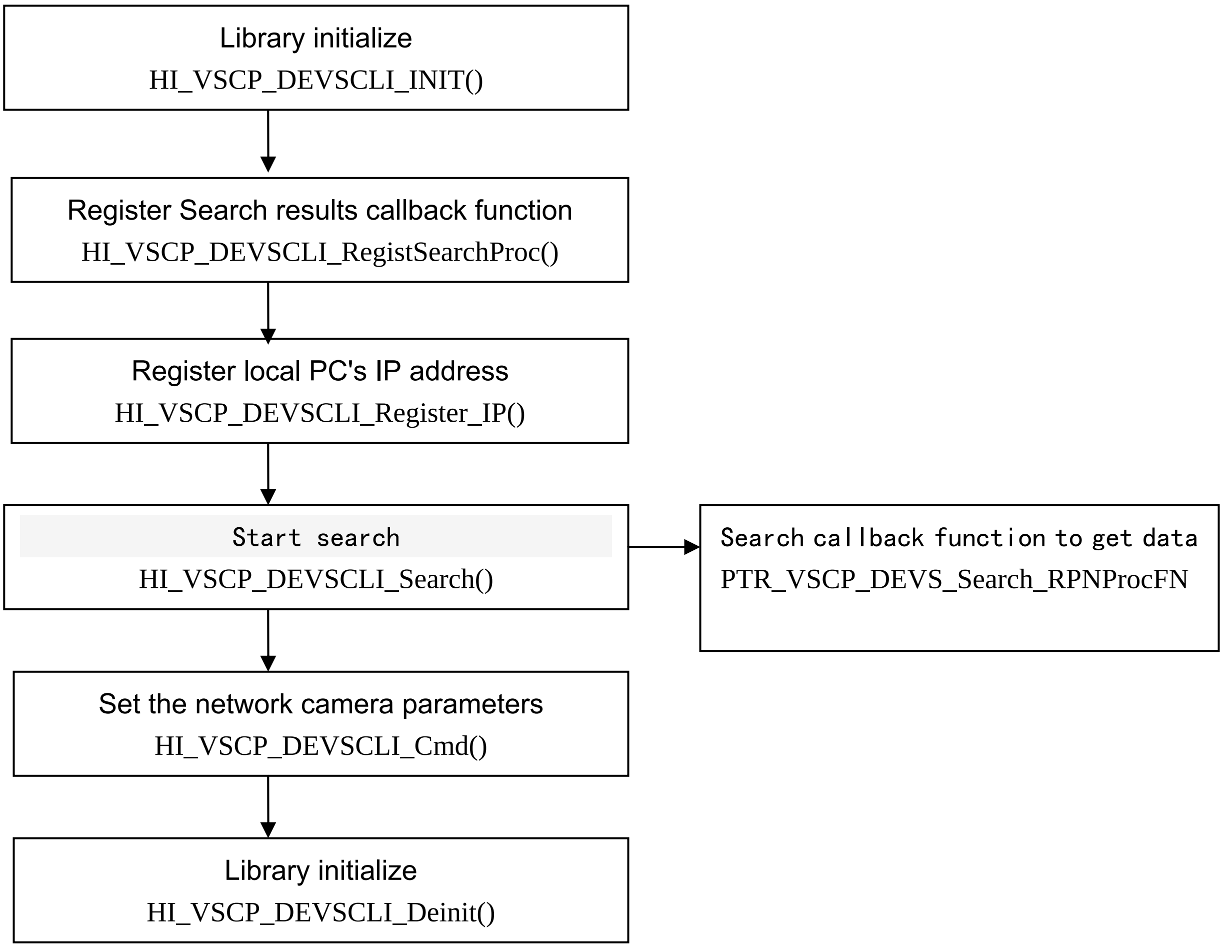
Return Values

HI_SUCCESS expresss success, HI_ FAILURE expresss failure.

Third part Searching SDK instruction

Version：1.0.0.2

3.1、programming guide


Library initialize
HI_VSCP_DEVSCLI_INIT()
Register Search results callback function
HI_VSCP_DEVSCLI_RegistSearchProc()
Register local PC's IP address
HI_VSCP_DEVSCLI_Register_IP()
Start search
HI_VSCP_DEVSCLI_Search()
Search callback function to get data
PTR_VSCP_DEVS_Search_RPNProcFN
Set the network camera parameters
HI_VSCP_DEVSCLI_Cmd()
Library initialize
HI_VSCP_DEVSCLI_Deinit()

3.2、Data structure

Device stream information:

```
typedef struct {
      HI_CHAR aszIP[HI_VSCP_IP_STRSIZE + 1];                    /*IP address*/
      HI_CHAR aszMASK[HI_VSCP_IP_STRSIZE + 1];                  /*subnet mask*/
      HI_CHAR aszMAC[HI_VSCP_MAC_STRSIZE + 1];                  /*MAC address*/
      HI_CHAR aszGTW[HI_VSCP_IP_STRSIZE + 1];                   /*gateway address*/
      HI_S32      s32Dhcp;       /* DHCP, 1 :open，0:close */
      HI_S32      s32DnsFlag; /* DNS set flag，1 :auto，0:manual*/
      HI_CHAR aszFdns[HI_VSCP_IP_STRSIZE + 1];       /* first choice DNS */
      HI_CHAR aszSdns[HI_VSCP_IP_STRSIZE + 1];      /* sencondary DNS */
} HI_S_VSCP_NETINFO;

typedef struct {
      HI_CHAR aszDevID[HI_VSCP_DEVID_STRSIZE + 1];   //Device ID，random get
HI_CHAR aszDevMDL[HI_VSCP_DEVNAME_STRSIZE + 1];   // Device mode
      HI_CHAR aszSwVersion[HI_VSCP_SWVER_STRSIZE + 1]; //Software version
      HI_CHAR aszDevName[HI_VSCP_DEVNAME_STRSIZE + 1]; //Device name
      HI_CHAR aszHttpPort[HI_VSCP_IP_STRSIZE + 1];        //HTTP monitor port
      HI_S_VSCP_NETINFO struNetInfo;
} HI_S_VSCP_DEVINFO;
```

Send command end device information

```
typedef   struct{
    HI_CHAR*   pszDevID;     //Device identify,the unique identify,the parameter can be
    obtained from devce searching
     HI_CHAR*   pszUserName;   //username
    HI_CHAR*   pszPasswd;         //pasword
}   HI_S_VSCP_DEVSCLI_DevInfo;
```

3.3、Port instruction

3.3.1 Initialize deviceserach

Initialize

```
    HI_S32   HI_VSCP_DEVSCLI_INIT (
        const HI_CHAR*         pszListenIP,
        HI_U16                 u16Port,
        HI_U32                 u32TimeOut,
        HI_VOID**          ppvHandle
    );
```

Parameters

pszListenIP

[IN] used to deal with multicast IP of searching answer. Fixed value is "239.255.255.250"

u16Port

[IN] used to deal with multicast port of searching answer.. Fixed value is "8002"

u32TimeOut

[IN] Search timeout. unit：second

ppvHandle

[IN] Enter search object handle

Return Values

HI_SUCCESS express success, HI_ FAILURE express failure.

3.3.2 To initialize device search

To initialize

```
    HI_S32   HI_VSCP_DEVSCLI_Deinit (
        HI_VOID*     pvHandle
    );
```

Parameters

pvHandle

[IN] Output search object handle

Return Values

HI_SUCCESS express success, HI_ FAILURE express failure.

3.3.3 Register search answer deal function

Register search answer deal function

```
    HI_S32    HI_VSCP_DEVSCLI_Deinit (
        HI_VOID*                    pvHandle,
        PTR_VSCP_DEVS_Search_RPNProcFN    pfunSearchRProc,
        HI_VOID*                    pvUserData
    );
```

Parameters

pvHandle

[IN] search object handle

pfunSearchRProc

[IN] Search answer deal callback function

pvUserData

[IN] user data, this parameter can be sent through search answer deal answer callback function

Callback Function

```
    typedef    HI_S32    (*PTR_VSCP_DEVS_Search_RPNProcFN) (
            const    HI_VOID*            pvHandle,
            HI_CHAR*                     pszRNPCode,
            HI_S_VSCP_DEVINFO*    pstruDevInfo,
            HI_VOID*                     pvUserData
            }
```

Callback Function Parameters

pvHandle

Unused

pszRNPCode

Return value

pstruDevInfo

Device information

pvUserData

User data

Return Values

HI_SUCCESS express success, HI_ FAILURE express failure.

Remarks

The channel and stream attribute of pstruDevinfo can be distribute area by SDK, the upper application program callback free function to free area after using the pstruDevinfo parameter.

3.3.4 Register command answer deal function

Register command answer deal function

```
HI_S32    HI_VSCP_DEVSCLI_RegistCmdProc (
      HI_VOID*                    pvHandle,
      PTR_VSCP_DEVS_Cmd_RPNProcFN    pfunCmdRProc,
      HI_VOID*                    pvUserData
);
```

Parameters

pvHandle

[IN] search object handle

pfunCmdRProc

[IN] Command answer deal function

pvUserData

[IN] User data. This parameter can be sent through command answer deal callback function..

Callback Function

```
typedef    HI_S32    (*PTR_VSCP_DEVS_Cmd_RPNProcFN) (
            const    HI_VOID*              pvHandle,
            HI_CHAR*                       pszRNPCode,
            HI_S_VSCP_DEVSCLI_Cmd_ResponsInfo*    pstruResponseInfo,
            HI_VOID*                       pvUserData
            }
```

Callback Function Parameters

pvHandle

Unused

pszRNPCode

Return value. It is success when the value contains 200, or else failure.

pstruResponseInfo

Unused

pvUserData

User data

Return Values

HI_SUCCESS express success, HI_ FAILURE express failure.

3.3.5 Register accept search answer local’s IP

Register accept search answer local’s IP

```
HI_S32   HI_VSCP_DEVSCLI_Register_IP (
        HI_CHAR      aaszIP[][HI_VSCP_IP_STRSIZE+1],
        HI_U32       u32Num
);
```

Parameters

aaszIP

[OUT] local IP address list

u32Num

[OUT] Local IP address number

Return Values

HI_SUCCESS express success, HI_ FAILURE express failure.

3.3.6 Send search command

Send search command

```
HI_S32   HI_VSCP_DEVSCLI_Search (
        HI_VOID*     pvHandle
);
```

Parameters

pvHandle

[IN] Search object handle

Return Values

HI_SUCCESS express success, HI_ FAILURE express failure.

3.3.7 Send set command

Send configuration command

```
HI_S32   HI_VSCP_DEVSCLI_Cmd (
        HI_VOID*                            pvHandle,
        const   HI_S_VSCP_DEVSCLI_DevInfo   *pstruDEV,
        HI_S32                              s32Cmd,
        const HI_VOID*                      pData
);
```

Parameters

pvHandle

[IN] Search object handle

PstruDEV

[IN] Device information

s32Cmd

[IN] Device type

#define HI_VSCP_CMD_NET 0x01 // Network basic parameter configuration

#define HI_VSCP_CMD_PORT 0x02 //port number

pData

[IN] set parameter

1、HI_VSCP_CMD_NET：HI_S_VSCP_NETINFO structure

```
typedef struct {
    HI_CHAR aszIP[HI_VSCP_IP_STRSIZE + 1];      //IP address
    HI_CHAR aszMASK[HI_VSCP_IP_STRSIZE + 1];     //subnet mask
    HI_CHAR aszMAC[HI_VSCP_MAC_STRSIZE + 1];         //MAC address
    HI_CHAR aszGTW[HI_VSCP_IP_STRSIZE + 1];      //gateway address
    HI_S32    s32Dhcp;                  //DHCP, 1 : open，0: close
    HI_S32    s32DnsFlag;          //DNS configure flag，1 : auto，0: manual
    HI_CHAR aszFdns[HI_VSCP_IP_STRSIZE + 1];      // first choiceDNS
    HI_CHAR aszSdns[HI_VSCP_IP_STRSIZE + 1];      // sencondary DNS
} HI_S_VSCP_NETINFO;
```

2、HI_VSCP_CMD_PORT：char str[16]

Return Values

HI_SUCCESS express success, HI_ FAILURE express failure.

Appendix

Ⅰ、File list

Lib	store library file, it has three files: libNetLib.so， NetLib.lib， NetLib.dll.
Include	store head file;
VC_demo	store mfc Demo；
Bin	the storage path of executive file.

Ⅱ、Factory code and device type definition

1. Factory code:

   Be used to identify produce factory;
   Can be alter by the special tool. Users can read, but cannot alter.
   ACSII code，32 byte size..

2. Device type

   Be used to identify device type, different device has different function.
   Can be alter by the special tool. Users can read, but cannot alter.
   ACSII code，32 byte size..

   Each field has 2 byte,the first byte express field type, the second field express sub type.

Field 1	Field 2	Field 3	Field 4	Field 5	Field 6	Field 7	Reserved fields
Chip	NTSC	LEN	PTZ type	Network type	Platform type	Language type	
‘C’	‘F’	‘S’	‘Z’	‘N’	‘P’	‘L’	

1). Chip field ’C’

Chip type:

‘0’	Hi3510
‘1’	Hi3512

2). NTSC field: ’F’

Video NTSC, the current value:

‘0’	PALand NTS support
‘1’	PAL(704x576, 352x288, 176x144) MAX:25 farme
‘2’	NTSC(704x480, 352x240, 176x120) MAX:30farme

3). LEN field: ‘S’

Photosensitive len type:

‘0’	OV7725	Brightness,contrast,saturation,hue

	LED control	Indoor,outdoor,open led, flip,mirror. Main stream：VGA, QVGA, QQVGA substream：QVGA, QQVGA
‘1’	CCDOSP	Brightness,contrast,saturation,hue Main stream：D1,CIF,QCIF substream：CIF,QCIF
‘2’	CCD	Brightness,contrast,saturation,hue Main stream：D1,CIF,QCIF substream：CIF,QCIF
‘3’	MT9D131	brightness，contrast(1-7)，saturation, flip, mirror, main stream：720P(max 30frame) sub stream：QVGA
‘4’	HDCCD	Main stream：720P(max:30 frame) substream：QVGA
‘5’	630D	Brightness(0-6)，contrast(0-8)，saturation,(0-6)。 Main stream：720P(max:30 frame) substream：Q720P
‘6’	630C	brightness(0-4)，contrast(0-4)，saturation (0-2)。 Main stream：720P(max 30frame) substream：Q720P
‘7’	CMOS 720P	brightness，contrast(1-7)，saturation，flip,mirror main stream：720P(max 30frame)，Q720P substream：Q720P，QQ720P
‘8’	633	brightness (0-6)，contrast (0-8)，saturation (0-6)。 Main stream：720P(max 30frame)，Q720P substream：Q720P，QQ720P

4). PTZ field: ‘Z’

PTZ type:

‘0’	Small ball	Up,down,left,right, down-up cruise,left-right cruise, back to the center position, preposition(max:8),no serial port configuration.
‘1’	White ball	Up,down,left,right, preposition(max:8).fixed serial port configuration.
‘2’	Zoom ball	Up,down,left,right,,zoom, preposition(max:8), fixed serial port configuration.
‘3’	Standard ball	Up,down,left,right, windshield wiper，light，preposition. You can configure serial port.
‘4’	Variable power ball	Up,down,left,right, down-up cruise,left-right cruise, back to the center position, zoom in ,zoom out

5). Network field: ‘N’

Network type:

‘0’	Support wird
‘1’	Support WIFI
‘2’	Support EVDO
‘3’	Support TD
‘4’	Support WCDMA

6). Platform field: ‘P’

Platform type

‘0’	No PTZ

7). Language field: ‘L’

Language type

‘0’	Chinese
‘1’	English

8). Reserved field is used for extension later