

Xtreme Imaging SDK v2.0

Developer's Guide

1. Introduction	4
2. DirectShow / MF	5
2.1 Introduction	5
2.2 Introduction of instance	5
3. Xtreme Imaging SDK	6
3.1 Introduction	6
3.2 Character set	7
3.3 Upgrade instructions of LibXiStream2	7
3.4 Examples	7
3.5 LibXiStream2	8
3.5.1 Open video device	8
3.5.2 Capture video device	9
3.5.3 Low latency callback function for video device	9
3.5.4 Video color space	9
3.5.5 Preview video	10
3.5.6 Display multiple-channel videos	10
3.5.7 Record to AVI file	11
3.5.8 Record to MP4 file	11
3.5.9 Low Latency mode of H.264	12
3.5.10 Snapshot	12
3.5.11 Audio capture	13
3.5.12 Audio Monitor	13
3.5.13 Thread Manager	13
3.6 LibXiProperty	14
3.6.1 Get the property handle of device	14
3.6.2 Get the property of device and signal	14
3.6.3 Set input signals of device	14
3.6.4 Firmware upgrade	15
3.7 Function of LibXiStream2 Description	15
3.7.1 XIS_Initialize	15
3.7.2 XIS_Uninitialize	15
3.7.3 XIS_RefreshDevices	16
3.7.4 XIS_GetVideoCaptureCount	16
3.7.5 XIS_GetVideoCaptureInfo	16
3.7.6 VIDEO_CAPTURE_INFO	16
3.7.7 XIS_GetVideoCaptureInfoEx	17
3.7.8 VIDEO_CAPTURE_INFO_EX	17
3.7.9 XIS_OpenVideoCapture	17
3.7.10 XIS_CloseVideoCapture	18
3.7.11 XIS_ShowVideoCapturePropertyDialog	18

3.7.12	XIS_OpenVideoCapturePropertyHandle	18
3.7.13	XIS_QueryInterface	18
3.7.14	XIS_TestVideoCaptureFormat.....	19
3.7.15	XIS_SetVideoCaptureFormat.....	19
3.7.16	XIS_SetVideoCaptureCallback	20
3.7.17	XIS_SetVideoCaptureCallbackEx	20
3.7.18	LPFN_VIDEO_CAPTURE_CALLBACK.....	20
3.7.19	LPFN_VIDEO_CAPTURE_CALLBACK_EX.....	21
3.7.20	XIS_StartVideoCapture.....	21
3.7.21	XIS_StopVideoCature	21
3.7.22	XIS_IsVideoCaptureStarted	22
3.7.23	XIS_CreateVideoRenderer	22
3.7.24	XIS_DestroyVideoRenderer	22
3.7.25	XIS_RsetVideoRenderer	23
3.7.26	XIS_SetVideoRendererPosition	23
3.7.27	XIS_VideoRendererDrawImage	23
3.7.28	XIS_VideoRendererRepaintRect	24
3.7.29	XIS_GetMoniterGUIDFromWindow	24
3.7.30	XIS_IsSupportH264HD	24
3.7.31	H264_ENCODER_PARAM	25
3.7.32	XIS_OpenVideoH264Encoder	25
3.7.33	XIS_CloseVideoH264Encoder	26
3.7.34	XIS_SetVideoCaptureEncoderOutput.....	26
3.7.35	XIS_CreatMP4Muxer	26
3.7.36	XIS_DestroyMP4Muxer	27
3.7.37	XIS_StartMP4Muxer	27
3.7.38	XIS_StopMP4Muxer	27
3.7.39	XIS_VideoCaptureCreateSnapShot.....	28
3.7.40	XIS_GetAudioCaptureCount.....	28
3.7.41	XIS_GetAudioCaptureInfoEx.....	28
3.7.42	XIS_OpenAudioCaptureEx	29
3.7.43	XIS_CloseAudioCapture.....	29
3.7.44	XIS_SetAudioCaptureFormat.....	29
3.7.45	LPFN_AUDIO_CAPTURE_CALLBACK	29
3.7.46	LPFN_AUDIO_CAPTURE_CALLBACK_EX	30
3.7.47	XIS_StartAudioCapture.....	30
3.7.48	XIS_StopAudioCapture	30
3.7.49	XIS_OpenAudioAACEncoder.....	31
3.7.50	XIS_StopAudioAACEncoder	31
3.7.51	XIS_VideoRendererBeginBatchDraw	31
3.7.52	XIS_VideoRendererBatchDrawRect.....	31
3.7.53	XIS_VideoRendererEndBatchDraw.....	32
3.7.54	XIS_CreateAVIMuxer	32
3.7.55	XIS_DestroyAVIMuxer	32

3.7.56	XIS_StartAVIMuxer	33
3.7.57	XIS_StopAVIMuxer.....	33
3.8	LibXIPProperty Function Description	33
3.8.1	XIP_OpenPropertyHandle	33
3.8.2	XIP_ClosePropertyHandle	33
3.8.3	XIP_GetDeviceType	34
3.8.4	XIP_GetHardwareInfo.....	34
3.8.5	XIPHD_IsSignalPresent/XIPCVBS_IsSignalPresent.....	34
3.8.6	XIPHD_IsSignalChanged/ XIPCVBS_IsSignalChanged	35
3.8.7	XIPHD_GetSignalFormat/XIPCVBS_GetSignalFormat.....	35
3.8.8	XIPCVBS_IsSquarePixelMode	35
3.8.9	XIPCVBS_SetSquarePixeMode.....	36
3.8.10	XIPCVBS_GetClip	36
3.8.11	XIPCVBS_SetClip	36
3.8.12	XIPHD_GetScaleType/ XIPCVBS_GetScaleType	37
3.8.13	XIPHD_SetScaleType/XIPCVBS_SetScaleType.....	37
3.8.14	XIPHD_GetDeinterlaceType/XIPCVBS_GetDeinterlaceType.....	37
3.8.15	XIPHD_SetDeinterlaceType/XIPCVBS_SetDinterlaceType	38
3.8.16	XIPHD_GetVideoProcParamRange	38
3.8.17	XIPCVBS_GetVideoProcParamRange.....	39
3.8.18	XIPHD_GetVideoProcParam/XIPCVBS_GetVideoProcParam	39
3.8.19	XIPHD_SetVideoProcParam/XIPCVBS_SetVideoProcParam.....	39
3.8.20	XIPHD_IsAutoDetectInputType	40
3.8.21	XIPHD_SetAutoDetectInputType.....	40
3.8.22	XIPHD_GetSupportedInputTypes	40
3.8.23	XIPHD_GetInputType.....	41
3.8.24	XIPHD_SetInputType	41
3.8.25	XIPHD_GetFlipMode	41
3.8.26	XIPHD_SetFlipMode	42
3.8.27	XIPHD_GetRGBComponentProcParam.....	42
3.8.28	XIPHD_SetRGBComponentProcParam	42
3.8.29	XIPHD_GetDefClip	43
3.8.30	XIPHD_GetClip.....	43
3.8.31	XIPHD_SetClip	44
3.8.32	XIPHD_GetSamplingPhaseRange	44
3.8.33	XIP_GetSamplingPhase	44
3.8.34	XIPHD_SetSamplingPhase	45
3.8.35	XIPHD_GetSamplingCountRange	45
3.8.36	XIPHD_GetSamplingCount	45
3.8.37	XIPHD_SetSamplingCount	46
3.8.38	XIP_GetSerialNo	46
3.8.39	XIP_GetFeatureData	46
3.8.40	XIPHD_IsHDMIPropertySupported.....	46
3.8.41	XIPHD_GetHDMIInfoFrame	47

1. Introduction

Xtreme Imaging SDK supports the capture devices whose drivers adopt Microsoft standard AVStream structure. On the Windows platform, developers can use following two methods to develop application software:

- Microsoft DirectShow SDK / Media Foundation (MF)
- Xtreme Imaging SDK

Here are the contrasts of these two ways:

Microsoft DirectShow / MF	● The software developed by Microsoft DirectShow / MF has a better compatibility and can support all capture devices that keep compatibility with Microsoft DirectShow / Media Foundation structure. ● DirectShow / MF needs higher requirement for developers. The developers must not only have enough background knowledge and a large number of development experiences, but also need to be familiar with the structure of the DirectShow / MF. ● The procedure of development is more complex. ● DirectShow / MF can support C++ and Microsoft Visual Studio perfectly, but it isn't friendly enough to support other development language and environment.
Xtreme Imaging SDK	● Software developed by Xtreme Imaging SDK only supports some special capture devices. ● It needs a lower requirement for developers. Developers only need to be familiar with the basic procedure of software development. ● The process of the development is very easy. Related features can be implemented by calling a few functions. ● Support standard C-Language API interface and support many other development languages and environment such as VB, C#, Delphi and C++ Builder.

2. DirectShow / MF

2.1 Introduction

On Windows platform, Microsoft DirectShow / MF is the best solution for development of streaming application. It has been integrated with Windows SDK and become a component of Windows.

The software developed by DirectShow / MF can keep compatibility with hardware that manufactured by many different companies. So it has enough generality and flexibility.

Users are convenient to develop software of capture device by DirectShow / MF interface. Developers also can achieve many kinds of customized services by extend interface of DirectShow / MF.

- Support up to six kinds of original image data format such as YUYV, I420 and Nv12.
- Test if current device can capture video signal.
- Support capturing images of multiple sizes from QCIF to 1080P.
- Support automatic image scaling followed to the size of original video signal.

If you want to get more information about DirectShow / MF, please refer to the documents of Microsoft Windows SDK.

2.2 Introduction of instance

There are instances developed by DirectShow / MF in SDK which can be referred by developers.

Instance Name	Development environment	Introduction
DShowSet	VC2008 / MFC / DirectX9	● Call DirectShow interface to preview the video device. ● Call extend interface of DirectShow to access advanced properties, including testing device signals and adjusting the scale method of capture device.
DShowPreview	VC2008 / MFC / DirectX9 / WM9	● Call DirectShow interface to implement the capture and preview function of video and audio device. ● Call Windows Media SDK to generate

		compressed ASF file.
DShowProperty	VC2008 / MFC / DirectX9	● Call DirectShow interface to implement the capture and preview function of video device. ● Call LibXIProperty API to access advanced properties of video device including status of the signal, unique serial number of the hard device and so on.
DShowUpdate	VC2008	● An instance based on DirectShow and LibXIProperty. ● Implement the firmware upgrade of the capture device.
Amcap	VC2008 / MFC / DirectX9	● A classic DirectShow instance released by Microsoft. ● Call DirectShow interface to implement the capture and preview function of video and audio device.

3. Xtreme Imaging SDK

3.1 Introduction

On Windows platform, Xtreme Imaging SDK is a set of C-LANGUAGE API interface which is based on the encapsulation of the Microsoft DirectShow. It simplifies the complex development procedure of the DirectShow and shields the implement details of the DirectShow. So users can effectively build streaming application software.

SDK provide so many interfaces, that developers can control media flexibly and can be deal with all details in the development of capture device. SDK is developed by Microsoft Visual Studio 2010 and Microsoft Visual C++. So it is better to use the same development language and environment.

SDK mainly include two core components. They are both support the mode of dynamic link which provide header files (.h), link library (.lib) and dynamic link library (.DLL). The developers need to load the DLL of SDK by the mode of dynamic link. Following are the introduction to the core components.

Component Name	Introduction about Component
LibXiStream2	Provide many features to video and audio device such as data capture, preview, data compression and special effect processing.
LibXiProperty	Provide the functions that can access the advanced properties of the capture device and can adjust the luminance, contrast ratio, and color of the video screen.

3.2 Character set

Xtreme Imaging SDK supports Unicode and multi-byte character sets. Different applications should use different libraries that use the same character set. Otherwise, exceptions may occur.

Character set	Debug version	Release version
Unicode	LibXiStream2d.lib LibXiPropertyd.lib	LibXiStream2.lib LibXiProperty.lib
Multi-byte	LibXiStreamA2d.lib LibXiPropertyAd.lib	LibXiStreamA2.lib LibXiPropertyA.lib

The concept of character set and the way of setting character set in compiler can be found in associated documents on MSDN.

3.3 Upgrade instructions of LibXiStream2

LibXiStream2 is an upgrade version of LibXiStream. Following are the upgrades based on the previous version.

- Simplified the interface of encoding the video signal by H.264 and the interface of writing data to MP4 file. So they are more convenient to use than before.
- Add the function that supports capturing and encoding 4K HD video (i.e. 4096 x 2160, 3840 x 2160).
- Add the low delay mode for H.264 encoding.
- Add the interface that can generate key frame mandatorily in H.264 encoding.
- Add the callback function of video capture with a lower delay.
- Add the interface based on Direct3D to combine the multiple-channel video.
- Adjust the audio capture interface to decrease the delay when capturing the audio.
- Add high precision timestamp for original video and audio data.

3.4 Examples

Examples with various development languages can be found in SDK and can be referred by

developers.

Name	Development Environment	Introduction
XICapture	VC2008 / MFC	● A single example base on LibXISStream2 and LibXIProperty. ● Support capturing and previewing single-channel video and audio device. ● Support screen shots. ● Record the signal and generate MP4 file (H.264/ACC) and AVI file without compression.
XICaptureQuad	VC2008 / MFC	● An example base on LibXISStream2 and LibXIProperty. ● Display the function that capture and preview quad-channel video device.
XICaptureCSharp	VC2008 / C#	● An example base on LibXISStream2 and LibXIProperty. ● The function is similar to XICapture but it is developed by C#.
XICaptureVB	VB2008	● An example base on LibXISStream2 and LibXIProperty. ● The function is similar to XICapture but it is developed by VB.

3.5 LibXISStream2

LibXISStream2 implement the capture, preview, shots and record function of video and audio device. Its structure is so flexible that it is convenient to extend its interfaces for developers.

3.5.1 Open video device

If you want to do any operation on the video device, you must open the video device first. The general calling procedure is as follows:

- Initial the library, **XIS_Initialize**.
- Get the number of current video device, **XIS_GetVideoCaptureCount**.
- Get the properties of the specified video device (mainly to acquire DshowID of the device), **XIS_GetVideoCaptureInfo**.

- Open the specified video device, **XIS_OpenVideoCapture**.

Following are the procedures for closing the video device:

- Close the device that has been opened, **XIS_CloseVideoCapture**.
- Quit the library, **XIS_Uninitialize**.

It is general to call initial and quit interface when opening and closing the application.

Sample code can be found in the instance “XICapture” and “XICaptureQuad”.

3.5.2 Capture video device

When opening the video device, the device can capture video signals. The procedure of starting the capture is as follows:

- Set callback function for captured image data, **XIS_SetVideoCaptureCallbackEx**.
- Set format for captured image, **XIS_SetVideoCaptureFormat**.
- Start to capture video signals, **XIS_StartVideoCapture**.

When starting to capture video, the original video data will be translated frame by frame by the callback function that you set.

When you need to stop the capture, call **XIS_StopVideoCapture** directly.

Note 1: The function accepts video image cannot contain time-consuming operation. Otherwise the function will not return the image data with the frame rate that you set.

Note 2: If you want to change the format of the captured image, you should stop capture first. Then you can reset the format before restarting the capture again.

Sample code can be found in the instance “XICapture” and “XICaptureQuad”.

3.5.3 Low latency callback function for video device

If you set **XIS_SetVideoCaptureCallbackEx** interface as the accept function of the captured data, the low delay mode will be used after capturing. The difference with normal mode is as follows:

- On low latency mode, the captured data will be returned directly by the callback function and there is no any buffer.
- On low latency mode, there is no mechanism for frame buffer.
- To avoid losing the frame, you should return the callback function in the shortest time.

3.5.4 Video color space

XI series of capture device all support follow six original image color space (data space).

Color Space	Introduction
YUYV	Each pixel point takes 2 bytes (16 bits), 4:2:2 sampling, it is equal to YUY2.
UYVY	Each pixel point takes 2 bytes (16 bits), 4:2:2 sampling, the sequence of bytes is just the opposite against YUYV.
I420	Each pixel points occupies 12 bits, 4:2:0 sampling, also called IYUV.

NV12	Each pixel points occupies 12 bits, 4:2:0 sampling, The way of sampling is the same as I420, But in the arrangement, U plane and V plane data order staggered storage.
RGB24	Each pixel point is 3 bytes (24 bits), the image data in accordance with the “RGBRGBRGB ...” arrangement, the order of the points of pixels from left to right and from bottom to top.
RGB32	Each pixel point is 4 bytes (32 bits), compared with RGB24,each pixel add a 0 byte (i.e., the Alpha channel 0), the image data in accordance RGB0RGB0 ... arranged in the order of the pixel point is from left to right, from bottom to top.

Because the captured image is original image data, the amount of data is a litter larger. But there is a restriction with band width when capture device transport the data. So it is better to avoid using RGB24 format and RGB32 format when developers are capturing 1080p HD video. This may avoid the phenomenon of losing frame caused by the shortage of band width.

3.5.5 Preview video

When the customer wants to preview the data that captured currently, the procedure is as follows:

- Get the GUID of the display device by the handler of this device, **XIS_GetMonitorGUIDFromWindow**.
- Create the display device, **XIS_CreateVideoRenderer**.
- Set the domain that displays the image, **XIS_SetVideoRendererPosition**.
- Draw the image on the display device, **XIS_VideoRendererDrawImage**.
- Repaint the image in the response function of WM_PAINT event of display device, **XIS_VideoRendererRepaintRect**.
- Reset the domain that displays the video in the response function of WM_MOVE and WM_SIZE event of display device, **XIS_SetVideoRendererPosition**.

Directly calling **XIS_DestroyVideoRenderer** can close the preview.

Notes1: if you want to preview the original image that you capture, you can call **XIS_VideoRendererDrawimage** directly in the function that accept data. The function can be set by **XIS_SetVideoCaptureCallback**.

Notes2: Image format (i.e. color space, width and height of the image) that input for creating the display device must be the same as the image format that drawing on the device. If the image data drawing on the display device is the original data, the format must keep same with the one set by the capture device.

Sample code can be found in the instance “XICapture”, “XICaptureQuad”.

3.5.6 Display multiple-channel videos

To display multiple-channel videos, users can create several display devices and each one displays the video as the single-channel. But the efficiency of this way is very low and it cost many system resources.

SDK provide optimized function to solve this problem. This function improves efficiency greatly when displaying the video of several devices on one window. The procedure is similar as single-channel, the only difference is on the part of drawing the image. The detail step is as follows:

- Get the GUID of the display device by the handler of this device, **XIS_GetMonitorGUIDFromWindow**.
- Create the display device, **XIS_CreateVideoRenderer**.
- Set the domain that displays the image, **XIS_SetVideoRendererPosition**.
- Open the multiple drawing modes before drawing the first frame data if there are more than one capture devices, **XIS_VideoRendererBeginBatchDraw**.
- Drawing one frame data in turn on all video devices, **XIS_VideoRendererBatchDrawRect**.
- Close the multiple channel drawing modes after drawing on the devices, **XIS_VideoRendererEndBatchDraw**.
- Draw the image on the display device, **XIS_VideoRendererDrawImage**.
- Repaint the image in the response function of WM_PAINT event of display device, **XIS_VideoRendererRepaintRect**.
- Reset the domain that displays the video in the response function of WM_MOVE and WM_SIZE event of display device, **XIS_SetVideoRendererPosition**.

Note1: In multiple-channel preview, the capture argument of capture device (i.e. color space, frame rate) excluding size of video must keep the same.

Note2: You can set several video on different position of one display device by **XIS_SetVideoRendererPosition**. This way can implement special effect of splitting window and PIP.

Sample code can be found in the instance "XICaptureQuad".

3.5.7 Record to AVI file

If you want to record nondestructive AVI file, you can do as follow procedures:

- Create AVI file muxer, **XIS_CreateAVIMuxer**.
- Start recording, **XIS_StartAVIMuxer**.

Following are the procedures of stopping the recording:

- Stop recording, **XIS_StopAVIMuxer**.
- Destroy AVI file muxer, **XIS_DestroyAVIMuxer**.

Note1: The video and audio device must be in capture status before recording.

Note2: An AVI file can contain one channel video and one channel audio or contain only one channel video without audio. Other situations are not permitted.

Sample code can be found in the instance "XICaptureQuad".

3.5.8 Record to MP4 file

If you want to record MP4 file compressed by H.264/AAC, following procedure can be used:

- Text whether the local hard environment support hard compression or not, **XIS_IsSupportH264HD**.

- Open the video and audio device and begin to capture and preview the device.
- Open the H.264 encoder on the video device, **XIS_OpenVideoH264Encoder**.
- Open the ACC encoder on the audio device, **XIS_OpenAudioACCEncoder**.
- Create MP4 muxer, **XIS_CreateMP4Muxer**.
- Start MP4 muxer, **XIS_StartMP4Muxer**.

Follow are the procedures of stopping the recording:

- Stop MP4 muxer, **XIS_StopMP4Muxer**.
- Destroy MP4 muxer, **XIS_DestroyMP3Muxer**.
- Close the H.264 encoder on the video device, **XIS_CloseVideoH264Encoder**.
- Close the ACC encoder on the audio device, **XIS_CloseAudioAACEncoder**.

Note1: The video and audio device must be in capture status before recording.

Note2: An MP4 file can contain one channel video and one channel audio or contain only one channel video without audio. Other situations are not permitted.

Note3: Because H.264 uses the hardware acceleration, the soft and hard environment must meet some conditions for recording. Following are the detail conditions:

- The operation system must be Windows 7 or Windows 8 and Windows XP is not permitted.
- Use Intel's CPU that has Sandy Bridge structure or Ivy Bridge structure.
- Use Intel HD Graphics and the driver must update to the latest version.
- The customer can text whether the local environment support or not by calling **XIS_IsSupportH264HD**.

Sample code can be found in the instance "XICapture".

3.5.9 Low Latency mode of H.264

In normal H.264 encoder has eight frame delays. If encoding has a higher real-time request, low latency mode can be used. Uses only need to set the value of nLowLatency to 1 in **H264_ENCODER_PARAM**.

On low latency mode, encoder delay will decrease to one frame.

Note: The encoding quality of video image will be reduced on low latency mode.

3.5.10 Snapshot

When video device begin to capture data, we can generate a screen shot of current video device by calling **XIS_VideoCaptureCreateSnapShot**. The picture is saved as JPG file. This interface can be called when the video is recoding.

Note: This function is carried out by an internal thread, so there is a litter delay when generating the picture. So you must not close the video device immediately after calling this function. Otherwise, the operation would be failed.

Sample code can be found in the instance "XICapture".

3.5.11 Audio capture

If the customer wants to capture the audio device, following are the procedures:

- Get the amount of audio device in current system, **XIS_GetAudioCaptureCount**.
- Get the name and ID of audio device, **XIS_GetAudioCaptureInfoEx**.
- Open the audio device, **XIS_OpenAudioCaptureEx**.
- If necessary, set the callback function for the audio data, **XIS_SetAudioCaptureCallbackEx**.
- Start audio capture, **XIS_StartAudioCapture**.

If you want to record into MP4 file, following are the procedures:

- Set the linker between audio device and AAC encoder, **XIS_SetAudioAACDataOutput**.
- Add the audio to the MP4 muxer, **XIS_MP4MuxerAddAudioStream**.
- Push the data to MP4 muxer, **XIS_MP4MuxerPutFrame**.

After using the audio device, you can stop audio capturing by calling **XIS_StopAudioCapture** and close audio device by calling **XIS_CloseAudioCapture**.

In this SDK version, some audio capture interfaces have important upgrades; here are the functions that have changed.

- **XIS_OpenAudioCapture**, this interface has been obsolete and use **XIS_OpenAudioCaptureEx** instead. This interface is modified to open the specified device by the sequential number.
- **XIS_SetAudioCaptureCallback** can be used continually. But we suggest upgrading to **XIS_SetAudioCaptureCallbackEx** which add the timestamp about the audio data.
- **XIS_SetAudioCaptureFormat**, If the function don't call, the audio device will use the format in sample rate 48KH, double-channel and 16-bit sample. If the function had been called to set audio device in the other format, the audio data timestamp will invalid.

Sample code can be found in the instance "XICapture".

3.5.12 Audio Monitor

When audio device begin to capture data, you can monitor the captured voice by calling **XIS_SetAudioMonitor** and also adjust the volume by calling **XIS_SetAudioMeterCallBack**. Audio monitor will not have any impact on audio recording.

Sample code can be found in the instance "XICapture".

3.5.13 Thread Manager

In SDK thread will be created in device capture and video encoding:

- Each video devices and audio devices will create thread to capture data. The capture callback functions will be called in the threads.
- Each video encoder will create a thread and the encoder's callback function will be called in the thread.

3.6 LibXIPProperty

LibXIPProperty is mainly used to get the status of the capture device and set the property of the capture device.

3.6.1 Get the property handle of device

Getting the handle of device property is the precondition of doing all operations on capture device. There are two ways to accept device handle:

- If you develop with DirectShow, you can translate pointer **ICaptureFilter** to property handle directly by calling **XIP_OpenPropertyHandle**. Sample code can be found in the instance "DShowProperty".
- If you develop with XI SDK, use **XIS_OpenVideoCapturePropertyHandle** to get the property handle.
- If you want to close the property handle, use **XIP_ClosePropertyHandle**. Sample code can be found in the instance "XICapture".

3.6.2 Get the property of device and signal

After getting property handle of the device, we can query property of device:

- Get device type by **XIP_GetDeviceType**.
- Get firmware information of current device by **XIP_GetFirmwareVerInfo**.
- Get hardware information of device by **XIP_GetHardwareInfo**.
- Get the unique hardware serial number of device by **XIP_GetSerialNo**.
- Test whether there are signals now or not by **XIPHD_IsSignalPresent** / **XIPCVBS_IsSignalPresent**.
- Test whether the signal is changed or not by **XIPHD_IsSignalChanged** / **XIPCVBS_IsSignalChanged**.
- Get the signal format of current device by **XIPHD_GetSignalFormat** / **XIPCVBS_GetSignalFormat**.

Sample code can be found in the instance "DShowProperty" and "XICapture".

3.6.3 Set input signals of device

After getting property handle of the device, we can query and set input signal property of device.

- Get whether current device choose input signal automatically or not by **XIPHD_IsAutoDetectInputType**.
- Modify whether current device choose input signal automatically or not by **XIPHD_SetAutoDetectInputType**.

- Get the signal types that device support by **XIPHD_GetSupportedInputTypes**.
- Get the signal type that current device input by **XIPHD_GetInputType**.
- Modify the signal type that current device input by **XIPHD_SetInputType**.

Signal types that support auto choose include: XIPHD_INPUT_DVI_HDMI, XIPHD_INPUT_VGA, and XIPHD_INPUT_Component.

Following types do not support auto choose: XIPHD_INPUT_SVideo, XIPHD_INPUT_CVBS.

Sample code about relative functions can be found in instance “XICapture”.

3.6.4 Firmware upgrade

After getting the property handle of device, we also can upgrade firmware of the device. Upgrading firmware is a dangerous operation. If the electricity is off on upgrading, this may cause the device unavailable. So please take this operation carefully. Before upgrade, we suggest to backup the firmware. After upgrade we need to turn off the electricity to restart the device.

- Backup the firmware by **XIP_BackupFirmware**.
- Upgrade the firmware to specified version by **XIP_UpgradeFirmware**.
- Terminate the upgrade temporarily by **XIP_FinishFirmwareOperations**.

3.7 Function of LibXIStream2 Description

3.7.1 XIS_Initialize

Function Name	XIS_Initialize		
Parameter	None		
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Initialize XIStream library. Call it before other XIStream functions are called. Usually it is placed at the beginning of the program, and is only usable together with XIS_Uninitialize .		

3.7.2 XIS_Uninitialize

Function Name	XIS_Uninitialize		
Parameter	None		
Return Value	None		
Function Description	Release the resource quoted by XIStream library. Usually it is called when the application is about to exit. It is only useable together with XIS_Initialize .		

3.7.3 XIS_RefreshDevices

Function Name	XIS_RefreshDevices		
Parameter	None		
Return Value	BOOL	TRUE	Success
		FALSE	Fail
Function Description	Double check the capture device in the system, if the system supports PCI Express hot-plug (such as a laptop), this function can be called before XIS_GetVideoCaptureCount to get the latest device number. Generally this function will not be called.		

3.7.4 XIS_GetVideoCaptureCount

Function Name	XIS_GetVideoCaptureCount	
Parameter	None	
Return Value	int	Return the number of video capture devices.
Function Description	Since a video capture card is usually equipped with multiple video capture devices, the returned value may be different from the number of video capture cards.	

3.7.5 XIS_GetVideoCaptureInfo

Function Name	XIS_GetVideoCaptureInfo		
Parameter	iDevice	Device serial number, starting from 0.	
	pVideoCaptureInfo	Returned device information.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	For details, please refer to the structure description about VIDEO_CAPTURE_INFO.		

3.7.6 VIDEO_CAPTURE_INFO

Struction Name	VIDEO_CAPTURE_INFO	
Parameter	adapterModel	Capture card model
	deviceType	Capture device type
	szName	Name of capture device
	szDShowID	DirectShow ID of capture device, which is used to create DirectShow Capture Filter

3.7.7 XIS_GetVideoCaptureInfoEx

Function Name	XIS_GetVideoCaptureInfoEx		
Parameter	iDevice	Device serial number, starting from 0.	
	pVideoCaptureInfoEx	Return to device information.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	For details, please refer to the structure description about VIDEO_CAPTURE_INFO_EX.		

3.7.8 VIDEO_CAPTURE_INFO_EX

Struction Name	VIDEO_CAPTURE_INFO_EX	
Parameter	adapterModel	Capture card Model
	deviceType	Capture device type
	szName	Capture device name
	szDShowID	DirectShow name of capture device could be used to create DirectShow Capture Filter.
	cxMin/cxMax	Min/Max width of output image
	cyMin/cyMax	Min/Max height of output image
	nFrameDurationMin / nFrameDurationMax	Min/Max value of the frame duration of the output image (100ns as one unit)
	cxGranularity/ cyGranularity	Granularity of output image's width / height
	dwColorFormatMasks	Support the color format bitmask: Like dwColorFormatMasks & (1<< XI_COLOR_RGB24) not equal to zero means support RGB24 Color format.

3.7.9 XIS_OpenVideoCapture

Function Name	XIS_OpenVideoCapture		
Parameter	lpszDShowID	DirectShow name of capture device.	
Return Value	HANDLE	NULL	Failure
		Other Values	Video capture device handle.
Function Description	Through video capture device handle, you can do every operation on the video capture device. If you no longer use video capture devices, you should call		

	XIS_CloseVideoCapture to release resources.
--	--

3.7.10 XIS_CloseVideoCapture

Function Name	XIS_CloseVideoCapture		
Parameter	hVideoCapture	The video capture device handle returned by XIS_OpenVideoCapture.	
Return Value	None		
Function Description	It is only usable together with XIS_OpenVideoCapture .		

3.7.11 XIS_ShowVideoCapturePropertyDialog

Function Name	XIS_ShowVideoCapturePropertyDialog		
Parameter	hVideoCapture	The video capture device handle returned by XIS_OpenVideoCapture.	
	hWndParent	Parent window of properties dialog	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Display video capture device properties dialog. For properties dialog windows, please check appendix 5.1, 5.2.		

3.7.12 XIS_OpenVideoCapturePropertyHandle

Function Name	XIS_OpenVideoCapturePropertyHandle		
Parameter	hVideoCapture	The video capture device handle returned by XIS_OpenVideoCapture.	
Return Value	HANDLE	NULL	Failure
		Other Value	Video capture device property handle
Function Description	The return value of this function can be used to call XIProperty series functions. Please use XIP_ClosePropertyHandle to close the property handle returned by this function.		

3.7.13 XIS_QueryInterface

Function Name	XIS_QueryInterface		
Parameter	hVideoCapture	The video capture device handle returned by	

		XIS_OpenVideoCapture.	
	iid	The interface ID that need to check.	
	ppvInterface	Returned interface hand.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	This function can be used to check the interface hand of expansion properties COM object.		

3.7.14 XIS_TestVideoCaptureFormat

Function Name	XIS_TestVideoCaptureFormat		
Parameter	hVideoCapture	Video capture device handle.	
	colorFormat	Video capture color format.	
	cx	The width of the captured video must be integer multiple of 2. For CVBS device, Valid range: 40-768; For HD device, Valid range: 40 -4000.	
	cy	The height of the captured video must be integer multiple of 2. For CVBS device, Valid range: 30-576; for HD device, Valid range: 30-4000.	
	nFrameDuration	Use100ns as unit for frame duration.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Detect whether the device supports capture format settings.		

3.7.15 XIS_SetVideoCaptureFormat

Function Name	XIS_SetVideoCaptureFormat		
Parameter	hVideoCapture	Video capture device handle.	
	colorFormat	Video Capture color format.	
	cx	The width of the captured video must be integer multiple of 2. For CVBS device, Valid range: 40-768; For HD device, Valid range: 40-4000.	
	cy	The height of the captured video must be integer multiple of 2. For CVBS device, Valid range: 30-576; for HD device, Valid range: 30-4000.	
	nFrameDuration	Use 100ns as unit for frame duration.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function	Set the video capture format. If it is in the state of capture (have called XIS_StartVideoCapture), calling		

Description	this function is invalid.
-------------	---------------------------

3.7.16 XIS_SetVideoCaptureCallback

Function Name	XIS_SetVideoCaptureCallback		
Parameter	hVideoCapture	Video capture device handle.	
	lpfnVideoCaptureCallback	Video frame callback function.	
	pvParam	Parameters passed to callback function.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Set video capture callback function, when capture is started, the image data will be returned by video frame callback function. If it is in the state of capture (have called XIS_StartVideoCapture), calling this function is invalid.		

3.7.17 XIS_SetVideoCaptureCallbackEx

Function Name	XIS_SetVideoCaptureCallbackEx		
Parameter	hVideoCapture	Video capture device handle.	
	lpfnVideoCaptureCallbackEx	Video frame callback function which can return the timestamp of current frame.	
	pvParam	Parameters passed to callback function.	
	bLatencyMode	Whether adopt low delay mode.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Set video capture callback function, when capture is started, the image data will be returned by video frame callback function. If low delay mode is enabled, the internal buffer and frame compensation mechanism would be canceled when capturing the video. If it is in the state of capture (have called XIS_StartVideoCapture), calling this function is invalid.		

3.7.18 LPFN_VIDEO_CAPTURE_CALLBACK

Function Name	LPFN_VIDEO_CAPTURE_CALLBACK	
Parameter	pbyImage	Image data.
	cblImageStride	The number of bytes of each line of the image.
	pvParam	Parameter when setting callback function.
Return Value	None	

Function Description	The function will be called in frame duration which had been set in <code>XIS_SetVideoCaptureFormat</code>. Every be called, the function return image data in a frame. The image data will be saved in the first param (pbyImage). This point was allocated by SDK and the user only can be allowed to read it. When the function returned, the point will be invalid. The image data length = <code>cbImageStride</code> * the image height. The function will be called in video capture thread. So it need to be returned as soon as possible.
----------------------	--

3.7.19 LPFN_VIDEO_CAPTURE_CALLBACK_EX

Function Name	LPFN_VIDEO_CAPTURE_CALLBACK	
Parameter	pbyImage	Image data.
	cbImageStride	The number of bytes of each line of the image.
	pvParam	Parameter when setting callback function.
	u64TimeStamp	High precision timestamp with unit 1/100000000s
Return Value	None	
Function Description	Please see "LPFN_VIDEO_CAPTURE_CALLBACK" function description.	

3.7.20 XIS_StartVideoCapture

Function Name	XIS_StartVideoCapture		
Parameter	hVideoCapture	Video capture device handle.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Start video capture. It is only usable together with XIS_StopVideoCapture.		

3.7.21 XIS_StopVideoCature

Function Name	XIS_StopVideoCapture		
Parameter	hVideoCapture	Video capture device handle.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Stop video capture. It is only usable together with XIS_StartVideoCapture.		

3.7.22 XIS_IsVideoCaptureStarted

Function Name	XIS_IsVideoCaptureStarted		
Parameter	hVideoCapture	Video capture device handle.	
Return Value	BOOL	TRUE	In the capture state
		FALSE	In a stopped state
Function Description	Get the current capture state.		

3.7.23 XIS_CreateVideoRenderer

Function Name	XIS_CreateVideoRenderer		
Parameter	pGUID	GUID of the Video display device	
	hWnd	Video display window.	
	colorFormat	Video color format	
	nWidth	Width of the video.	
	nHeight	Height of the video.	
	bUseOverlay	Whether suit for Overlay, set as TRUE, when the system supports Overlay, it will use Overlay to display video, display a higher performance.	
	bWaitforVerticalBlank	Whether start to draw when vertical retrace appear. Set as TRUE could prevent screen tearing.	
Return Value	HANDLE	NULL	Failure
		Other Value	Video display handle.
Function Description	By video capture device handle, you can perform operations regarding to video display. If you no longer use the video capture device, you should call XIS_DestroyVideoRenderer to release resources.		

3.7.24 XIS_DestroyVideoRenderer

Function Name	XIS_DestroyVideoRenderer	
Parameter	hVideoRenderer	Video display handle
Return Value	None	
Function Description	Turn off the video display handle, and release resources.	

3.7.25 XIS_RsetVideoRenderer

Function Name	XIS_ResetVideoRenderer		
Parameter	hVideoRenderer	Video display handle	
	pGUID	GUID of Video display device	
	hWnd	Video display window	
	colorFormat	Video color format	
	nWidth	Width of video	
	nHeight	Height of video	
	bUseOverlay	Whether suit for Overlay, set as TRUE, when the system supports Overlay, it will use Overlay to display video, display a higher performance.	
	bWaitForVerticalBlank	Whether start to draw when vertical retrace appear. Set as TRUE could prevent screen tearing.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Modify the properties of video display when you haven't turned off the video display handle.		

3.7.26 XIS_SetVideoRendererPosition

Function Name	XIS_SetVideoRendererPosition		
Parameter	hVideoRenderer	Video display handle	
	lprcSource	Set the coordinates of the display domain in the original video, if it is NULL, it means displaying the complete video.	
	lprcDest	Set the coordinates that video displayed in the window.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	By setting the goal rectangle, it could display video in a partial region of the window. By specifying a source rectangle, you can cut the video before display it.		

3.7.27 XIS_VideoRendererDrawImage

Function Name	XIS_VideoRendererDrawImage		
Parameter	hVideoRenderer	Video display handle	
	pblImage	A pointer to the video image data	
	cbStride	The number of bytes of the video image data of each line	
	bSrcTopDown	The video image whether stored by a top-down format	
Return Value	BOOL	TRUE	Success

		FALSE	Failure
Function Description	In order to draw video image, video image data must comply with the specified picture format which is set by calling XIS_CreateVideoRenderer or XIS_ResetVideoRenderer last time. Video will display in the specified location according to XIS_SetVideoRendererPosition. This function is generally called when the new video image data is received.		

3.7.28 XIS_VideoRendererRepaintRect

Function Name	XIS_VideoRendererRepaintRect		
Parameter	hVideoRenderer	Video display handle	
	lprcSource	A pointer to the region where need to redraw	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Redraw the window region that specified. Call this function when the windows receive WM_PAINT. If XIS_VideoRendererDrawImage is called periodically, invalid window will be updated, and this function needs not to be called.		

3.7.29 XIS_GetMonitorGUIDFromWindow

Function Name	XIS_GetMonitorGUIDFromWindow		
Parameter	hWnd	Video display handle	
	pGuidMonitor	A pointer to GUID variable returned by DirectDraw device.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Get the relative display DirectDraw GUID for hWnd which can input into XIS_CreateVideoRenderer or XIS_RsetVideoRenderer.		

3.7.30 XIS_IsSupportH264HD

Function Name	XIS_IsSupportH264HD		
Parameter	None		
Return Value	BOOL	TRUE	Support
		FALSE	Not Support
Function Description	Check whether the local machine support hardware H.264 coding. If the return value is FALSE, it means it does not support hardware H.264 coding. If you call XIS_CreateH264Encoder, it will create software encoder of H.264.		

3.7.31 H264_ENCODER_PARAM

Name	H264_ENCODER_PARAM	
Member	cx	Width of the original video image
	cy	Height of the original video image
	nFrameDuration	Frame duration of original video
	nKbps	Bit rate of the encoding video
	usage	Quality of the encoding video
	profile	Algorithm of the encoding video
	nGopPicSize	Interval of the key frame, default is 0, means disable.
	nGopRefSize	Interval of the reference frame, default 0, means disable
	nDstX	Width of video after encoding, default is 0, means disable.
	nDstY	Height of video after encoding, default is 0, means disable.
	nLowLatency	Option for low delay mode, default is 0, means disable.
Description	This parameter is used to configure H.264 encoder. There are three kinds of encoding video quality. ● XI_TARGETUSAGE_BEST_QUALITY has the highest encoding quality. ● XI_TARGETUSAGE_BALANCED has a middle quality. ● XI_TARGETUSAG_BEST_SPEED has a fastest speed and a common quality There are also three kinds of algorithm level: ● XI_PROFILE_BASELINE is a baseline algorithm. ● XI_PROFILE_MAIN is a main profile algorithm. ● XI_PROFILE_HIGH is a high profile algorithm. It is not suitable to set the interval of key frame too small. If set the value of reference frame to 1, the B frame would not be used. If set the value of law delay option to 1, the low delay encoding mode would be opened.	

3.7.32 XIS_OpenVideoH264Encoder

Function Name	XIS_OpenVideoH264Encoder	
Parameter	HANDLE	Handle of video capture device
	const H264_ENCODER_PARAM*	H.264 encoder parameter
	LPFN_H264_ENCODER_CALLBACK	Callback function of H.264 encoder, the compressed image data will be returned by this function when starting to

		compress.	
	void *	The parameter is customized by H.264 encoder's callback function.	
Return Value	BOOL	TRUE	Calling successfully
		FALSE	Calling failed
Function Description	Open the H.264 encoder on the video capture device. If local environment support the hardware coding, an encoder will be created. Otherwise, return false.		

3.7.33 XIS_CloseVideoH264Encoder

Function Name	XIS_CloseVideoH264Encoder	
Parameter	HANDLE	Handle of video capture device
Return Value	None	
Function Description	Close the H.264 encoder on the video capture device. It is only usable together with XIS_OpenVideoH264Encoder .	

3.7.34 XIS_SetVideoCaptureEncoderOutput

Function Name	XIS_SetVideoCaptureEncoderOutput		
Parameter	hVideoCapture	Handle of video capture device returned by XIS_OpenVideoCapture	
	hH264Encoder	H264 encoder handle returned by XIS_OpenVideoH264Encoder.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	When starting to compress, it is used to connect video capture device and H.264 encoder. It will be called when H. 264 is successfully created. When the compression stops, it is used to disconnect the video capture device and H.264 encoder by setting the hH264encoder as NULL. Use it before XIS_DestroyH264Encoder.		

3.7.35 XIS_CreatMP4Muxer

Function Name	XIS_CreateMP4Muxer		
Parameter	lpszPath	Path for the generated MP4 file.	
	nTimeScale	Audio sampling rate, if there is no audio, it can send 0.	
	hVideoCapture	Video capture device handle that has opened the H.264 encoder.	
	hAudioCapture	Audio capture device handle that has opened the ACC	

		encoder.	
Return Value	HANDLE	NULL	Calling error
		Other Values	Handle of MP4 file muxer
Function Description	Create MP4 muxer. Please be sure to open the corresponding video device and audio device. Otherwise it may cause failure when calling this function.		

3.7.36 XIS_DestroyMP4Muxer

Function Name	XIS_DestroyMP4Muxer		
Parameter	hMP4Muxer	MP4 muxer handle returned by XIS_OpenVideoH264Encoder	
Return Value	None		
Function Description	Destroy the MP4 muxer. It is only usable together with XIS_CreateMP4Muxer .		

3.7.37 XIS_StartMP4Muxer

Function Name	XIS_StartMP4Muxer		
Parameter	hMP4Muxer	MP4 muxer handle returned by XIS_OpenVideoH264Encoder	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Begin to input data into MP4 file.		

3.7.38 XIS_StopMP4Muxer

Function Name	XIS_StopMP4Muxer		
Parameter	hMP4Muxer	MP4 muxer handle returned by XIS_OpenVideoH264Encoder	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Stop to input data into MP4 file.		

3.7.39 XIS_VideoCaptureCreateSnapshot

Function Name	XIS_VideoCaptureCreateSnapshot		
Parameter	hVideoCapture	Handle of video capture device returned by XIS_OpenVideoCapture	
	lpszFilePath	File path of snapshot	
	nQuality	The quality of snapshot, limited in 50 – 90. The higher the number, the higher the quality of snapshot, the larger the files.	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Generate the current snapshot		

3.7.40 XIS_GetAudioCaptureCount

Function Name	XIS_GetAudioCaptureCount	
Parameter	None	
Return Value	int	Return the number of audio capture devices
Function Description	Since an audio capture card is usually equipped with multiple audio capture devices, the returned value may be different from the number of audio capture cards.	

3.7.41 XIS_GetAudioCaptureInfoEx

Function Name	XIS_GetAudioCaptureInfo		
Parameter	iDevice	device serial number, starting from 0	
	pAudioCaptureInfoEx	Returned device information	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Get device information. AUDIO_CAPTURE_INFO_EX: szName : audio device name; guid: audio device's GUID; id: audio device's ID, It's used by XIS_OpenAudioCaptureEx to open audio device;		

3.7.42 XIS_OpenAudioCaptureEx

Function Name	XIS_OpenAudioCaptureEx		
Parameter	pszID	ID of audio capture device	
Return Value	HANDLE	NULL	Failure
		Other Value	Audio capture device handle
Function Description	Through audio capture device handle, you can operate everything on the audio capture device. If you no longer use audio capture devices, you should call XIS_CloseAudioCapture to release resources.		

3.7.43 XIS_CloseAudioCapture

Function Name	XIS_CloseAudioCapture		
Parameter	hAudioCapture	Handle of audio capture device returned by XIS_OpenAudioCapture2	
Return Value	None		
Function Description	It is only usable together with XIS_OpenAudioCapture2 .		

3.7.44 XIS_SetAudioCaptureFormat

Function Name	XIS_SetAudioCaptureFormat		
Parameter	hAudioCapture	Audio capture device handle	
	cSamplesPersec	Audio samples per second	
	cChannels	Audio channels	
	cSamplesPerFrame	Audio samples per frame	
	cBufferedFrames	Audio frame's buffer count	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Set audio device capture format. If it is in the state of capture (have called XIS_StartAudioCapture), this function call is invalid.		

3.7.45 LPFN_AUDIO_CAPTURE_CALLBACK

Function Name	LPFN_AUDIO_CAPTURE_CALLBACK	
Parameter	pbFrame	audio data

	cbFrame	audio data length in byte
	pvParam	The parameter set for callback function
Return Value	None	
Function Description	If this function does too many time-consuming operations, it could cause that the actual capture frame rate is less than user-set frame rate.	

3.7.46 LPFN_AUDIO_CAPTURE_CALLBACK_EX

Function Name	LPFN_AUDIO_CAPTURE_CALLBACK	
Parameter	pbFrame	audio data
	cbFrame	audio data length in byte
	pvParam	The parameter set for callback function
	U64TimeStamp	High precision timestamp with unit 1/10000000s
Return Value	None	
Function Description	If this function does too many time-consuming operations, it could cause that the actual capture frame rate is less than user-set frame rate.	

3.7.47 XIS_StartAudioCapture

Function Name	XIS_StartAudioCapture		
Parameter	hAudioCapture	Audio capture device handle	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Start audio capture. It is only usable together with XIS_StopAudioCapture .		

3.7.48 XIS_StopAudioCapture

Function Name	XIS_StopAudioCapture		
Parameter	hAudioCapture	Audio capture device handle	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Stop audio capture It is only usable together with XIS_StartAudioCapture .		

3.7.49 XIS_OpenAudioAACEncoder

Function Name	XIS_OpenAudioAACEncoder		
Parameter	hAudioCapture	Audio capture device handle	
	lpCallback	Audio compress data callback function	
	pvParam	Audio compress data callback function's parameters	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Used to start or stop AAC audio compressor		

3.7.50 XIS_StopAudioAACEncoder

Function Name	XIS_CloseAudioAACEncoder		
Parameter	hAudioCapture	Audio capture device handle	
Return Value	BOOL	TRUE	Success
		FALSE	Failure
Function Description	Used to stop or stop AAC audio compressor. It is only usable together with XIS_StopAudioCapture .		

3.7.51 XIS_VideoRendererBeginBatchDraw

Function Name	XIS_VideoRendererBeginBatchDraw		
Parameter	hVideoRenderer	Handle of display device	
Return Value	BOOL	TRUE	success
		FALSE	fail
Function Description	Start multiple-channel drawing		

3.7.52 XIS_VideoRendererBatchDrawRect

Function Name	XIS_VideoRendererBatchDrawRect		
Parameter	hVideoRenderer	Handle of display device	
	pblImage	The pointer to the video image data	
	cxSrc	The width of image	
	cySrc	The height of image	
	bSrcTopDown	The video image whether stored by a top-down format	
	xDest	The x coordinate of drawing domain	

	yDest	The y coordinate of drawing domain	
Return Value	BOOL	TRUE	success
		FALSE	fail
Function Description	Begin to draw the image on the render in turn		

3.7.53 XIS_VideoRendererEndBatchDraw

Function Name	XIS_VideoRendererEndBatchDraw		
Parameter	hVideoRenderer	Handle of display device	
	bPresent		
Return Value	BOOL	TRUE	success
		FALSE	fail
Function Description	Start multiple-channel drawing		

3.7.54 XIS_CreateAVIMuxer

Function Name	XIS_CreateAVIMuxer		
Parameter	lpszPath	Path for the generated MP4 file.	
	hVideoCapture	The capture device handle returned by XIS_OpenVideoCapture	
	hAudioCapture	The audio device handle returned by XIS_OpenAudioCapture2	
Return Value	HANDLE	NULL	Fail
		Other Value	Handle of AVI muxer
Function Description	Create handle of AVI muxer. Please be sure to open the corresponding video device and audio device. Otherwise it may cause failure when calling this function.		

3.7.55 XIS_DestroAVIMuxer

Function Name	XIS_DestroAVIMuxer	
Parameter	hAVIMuxer	Handle of AVI muxer
Return Value	None	
Function Description	Destroy handle of AVI muxer	

3.7.56 XIS_StartAVIMuxer

Function Name	XIS_StartAVIMuxer		
Parameter	hAVIMuxer	Handle of AVI muxer	
Return Value	BOOL	True	Success
		False	Fail
Function Description	Begin to record		

3.7.57 XIS_StopAVIMuxer

Function Name	XIS_StopAVIMuxer	
Parameter	hAVIMuxer	Handle of AVI muxer
Return Value	None	
Function Description	Stop recording	

3.8 LibXIProperty Function Description

3.8.1 XIP_OpenPropertyHandle

Function Name	XIP_OpenPropertyHandle		
Parameter	pCaptureFilter	DirectShow capture filter hand	
Return Value	HANDLE	NULL	Failure
		Other Value	property handle of video capture device
Function Description	Open the property handle of capture filter. It needs to call XIP_ClosePropertyHandle to close this handle when using it up.		

3.8.2 XIP_ClosePropertyHandle

Function Name	XIP_ClosePropertyHandle	
Parameter	hXIProperty	Property handle
Return Value	None	

Function Description	Close the opened property handle of the capture filter.
----------------------	---

3.8.3 XIP_GetDeviceType

Function Name	XIP_GetDeviceType	
Parameter	hXIProperty	Property handle
Return Value	XI_DEVICE_TYPE	Returned to the types of video capture device
Function Description	Get the types of video capture device via property handle.	

3.8.4 XIP_GetHardwareInfo

Function Name	XIP_GetHardwareInfo		
Parameter	hXIProperty	Property handle	
	pInfo	Returned to the information of hardware device	
		szDeviceID	Device ID, including bus types, manufacturer's number and device number string.
		szInstanceID	Instance ID of the device in the OS
		ulBusNumber	Bus number
		usDeviceNumber	Device number
		usFunctionNumber	Function number
Return Value	HRESULT	S_OK	Success
		Other Value	Failure
Function Description	Get the types of video capture device via property handle.		

3.8.5 XIPHD_IsSignalPresent/XIPCVBS_IsSignalPresent

Function Name	XIPHD_IsSignalPresent/XIPCVBS_IsSignalPresent		
Parameter	hXIProperty	Property handle	
Return Value	HRESULT	S_OK	Have video signal
		S_FALSE	Have no video signal
		Other Value	Calling error
Function Description	Get to know whether there is signal connected to the video capture device.		

3.8.6 XIPHD_IsSignalChanged/ XIPCVBS_IsSignalChanged

Function Name	XIPHD_IsSignalChanged/XIPCVBS_IsSignalChanged		
Parameter	hXIProperty	Property handle	
Return Value	HRESULT	S_OK	The input video signal has already changed
		S_FALSE	The input video signal haven't changed
		Other Value	Calling error
Function Description	Get the information that whether the input signal of the video capture device has changed or not since the last calling of this function.		

3.8.7 XIPHD_GetSignalFormat/XIPCVBS_GetSignalFormat

Function Name	XIPHD_GetSignalFormat/XIPCVBS_GetSignalFormat		
Parameter	hXIProperty	Property handle	
	pnSignalWidth	Returned the original width of signal (pixel number).	
	pnSignalHeight	Returned the original height of signal (pixel number).	
	pnSignalFrameDuration	Returned the original frame interval (100ns per unit)	
Return Value	HRESULT	S_OK	Calling Successfully
		Other Value	Calling failed
Function Description	Get the original format of the input signal of the video capture device.		

3.8.8 XIPCVBS_IsSquarePixelFormat

Function Name	XIPCVBS_IsSquarePixelFormat		
Parameter	hXIProperty	Property handle	
Return Value	HRESULT	S_OK	square pixel mode
		S_FALSE	non-square pixel mode
		Other Value	Calling Error
Function Description	Set it to be square pixel mode. For NTSC signal, it will be digitalized to be 640x480, and for PAL signal, it will be digitalized to be 768x576. In this mode, the pixel width height rate is 1:1 which is suitable for PC display. In the non-square mode, for NTSC signal, it will be digitalized to be 720x480, and for PAL, it will be digitalized to be 720x576		

3.8.9 XIPCVBS_SetSquarePixelMode

Function Name	XIPCVBS_SetSquarePixelMode		
Parameter	hXIProperty	Property handle	
	bSquarePixelMode	TRUE means that the mode is set to be square pixels	
Return Value	HRESULT	S_Ok	Calling successfully
		Other Value	Calling Error
Function Description	Set digital methods for video capturing. Please refer to XIPCVBS_IsSquarePixelMode for details.		

3.8.10 XIPCVBS_GetClip

Function Name	XIPCVBS_GetClip		
Parameter	hXIProperty	property handle	
	pnLeft	Return to the removed pixel number on the left side of the image	
	pnTop	Return to the removed pixel number on the top of the image	
	pnWidth	Return to the clipped width of the image	
	pnHeight	Return to the clipped height of the image	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get the current clipped area. For NTSC and PAL, the capture card has different settings for clipping area. When the square pixel mode has changed, the clipping area will also change.		

3.8.11 XIPCVBS_SetClip

Function Name	XIPCVBS_SetClip		
Parameter	hXIProperty	Property handle	
	nLeft	The starting position of image relative to the left of original signal (0-128).	
	nTop	The starting position of image relative to the top of original signal (0-128).	
	nWidth	The clipped width of the image (512-original width)	
	nHeight	The clipped height of the image(320-original height)	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error

Function Description	When set the current clipping area, for NTSC and PAL, the capture card may have different settings for clipping area. When Square Pixel Mode is changed, the clipping area will change, too. nLeft, nTop, nWidth, nHeight must be the integer times of 2.
----------------------	--

3.8.12 XIPHD_GetScaleType/ XIPCVBS_GetScaleType

Function Name	XIPHD_GetScaleType/XIPCVBS_GetScaleType		
Parameter	hXIProperty	Property handle	
	pnScaleType	Return to the current image scale method	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get the current image scale mode, when the captured size and recommend size are different, the capture device will scale the image. XICVBS (HD) _SCALE_DO_NOT_KEEP_ASPECT: To be fully filled the screen but not guarantee the rate of the original image. XICVBS(HD)_SCALE_FILL_TO_KEEP_ASPECT: Keep the rate of original image and black edges will be filled at the top, bottom, left and right of the screen when it's necessary. XICVBS (HD) _SCALE_ZOOM_TO_KEEP_ASPECT :Keep the rate of original image and the pixel at the top, bottom, left and right of the screen will be clipped when it's necessary.		

3.8.13 XIPHD_SetScaleType/XIPCVBS_SetScaleType

Function Name	XIPHD_SetScaleType/XIPCVBS_SetScaleType		
Parameter	hXIProperty	Property handle	
	nScaleType	image scale method	
Return Value	HRESULT	S_OK	Calling Successfully
		Other Value	Calling Error
Function Description	Set the current image scale method. The capture device will scale the image when the captured size and recommended size are different. Please refer to the instructions of XIPCVBS (HD)_GetScaleType for details.		

3.8.14 XIPHD_GetDeinterlaceType/XIPCVBS_GetDeinterlaceType

Function Name	XIPHD_GetDeinterlaceType/XIPCVBS_GetDeinterlaceType	
Parameter	hXIProperty	Property handle
	pnDeinterlaceType	Return to the current method from interlacing to

		non-interlacing.	
Return Value	HRESULT	S_OK	Calling Successfully
		Other Value	Calling Error
Function Description	Get the current mode from interlacing to non-interlacing. When the input signal is interlacing signal, the capture card can automatically process the image according to this setting. XIPCVBS(HD)_DEINTERLACE_NODE: Do nothing. XIPCVBS(HD)_DEINTERLACE_BLEND: Use the Means algorithm to transfer the interlacing signal to non-interlacing signal. XIPCVBS(HD)_DEINTERLACE_MOTION_ADAPTIVE: Use the 3D motion-adaptive algorithm to transfer the interlacing signal to non-interlacing signal.		

3.8.15 XIPHD_SetDeinterlaceType/XIPCVBS_SetDeinterlaceType

Function Name	XIPHD_SetDeinterlaceType/XIPCVBS_SetDeinterlaceType		
Parameter	hXIProperty	Property handle	
	nDeinterlaceType	Method from interlacing to non-interlacing.	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Set the method from interlacing to non-interlacing. For details, refer to the instructions of XIPCVBS (HD) _GetDeinterlaceType.		

3.8.16 XIPHD_GetVideoProcParamRange

Function Name	XIPHD_GetVideoProcParamRange		
Parameter	hXIProperty	Property handle	
	videoProcParamType	Parameter type of video processing (like brightness, contrast, hue, saturation).	
	pIParamValueMin	Return to the minimum value of this parameter.	
	pIParamValueMax	Return to the maximum value of this parameter.	
	pIparamValueDef	Return to the default value of this parameter.	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get the maximum, minimum area and default value of the video processing parameter.		

3.8.17 XIPCVBS_GetVideoProcParamRange

Function Name	XIPCVBS_GetVideoProcParamRange		
Parameter	hXIProperty	Property handle	
	videoProcParamType	Parameter type of video processing (like brightness, contrast, hue, saturation).	
	pIParamValueMin	Return to the minimum value of this parameter.	
	pIParamValueMax	Return to the maximum value of this parameter.	
	pIParamValueDef	Return to the default value of this parameter.	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get the maximum, minimum area and default value of the video processing parameter.		

3.8.18 XIPHD_GetVideoProcParam/XIPCVBS_GetVideoProcParam

Function Name	XIPHD_GetVideoProcParam /XIPCVBS_GetVideoProcParam		
Parameter	hXIProperty	Property handle	
	videoProcParamType	Parameter type of video processing (like brightness, contrast, hue, saturation).	
	pIParamValue	Return to the current value of this parameter.	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get the current value of the video processing parameter.		

3.8.19 XIPHD_SetVideoProcParam/XIPCVBS_SetVideoProcParam

Function Name	XIPHD_SetVideoProcParam/XIPCVBS_SetVideoProcParam		
Parameter	hXIProperty	Property handle	
	videoProcParamType	Parameter type of video processing (like brightness, contrast, hue, saturation).	
	pIParamValue	Parameters should be within the returned values of XIPCVBS(HD)_GetVideoProcParamRange.	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error

Function Description	Set the current value of the video processing parameter.
----------------------	--

3.8.20 XIPHD_IsAutoDetectInputType

Function Name	XIPHD_IsAutoDetectInputType		
Parameter	hXIProperty	Property handle	
Return Value	HRESULT	S_OK	Detect the type of the input signal automatically.
		S_FALSE	Not detect the type of the input signal automatically.
		Other Value	Calling Error
Function Description	Query whether to use the automatic detection of the type of input signal If using the automatic detection of the type of input signal, Capture card automatically determine whether the input signal is DVI/HDMI, VGA or Y/Pb/Pr. And automatically set the current input signal type.		

3.8.21 XIPHD_SetAutoDetectInputType

Function Name	XIPHD_SetAudioDetectInputType		
Parameter	bXIProperty	Property handle	
	bAutoDetectInputType	Whether automatically detect the input type	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Setting automatic detection of the type of input signal See XIPHD_IsAutoDetectInputType		

3.8.22 XIPHD_GetSupportedInputTypes

Function Name	XIPHD_GetSupportedInputTypes		
Parameter	hXIProperty	Property handle	
	pnSupportedInputTypesMasks	Return the input signal type that device support.	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get the input signal type that device support (bitmask). For example, if the returned value is the same as 1<<XIPHD_INPUT_DVI_HDMI, then set DVI or HDMI signal supported.		

3.8.23 XIPHD_GetInputType

Function Name	XIPHD_GetInputType		
Parameter	hXIProperty	Property handle	
	pnInputType	Return the current setting of the input signal type.	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Return the current setting of the input signal type. XIPHD_INPUT_DVI_HDMI: DVI or HDMI signal XIPHD_INPUT_VGA: VGA signal XIPHD_INPUT_Component: Y/Pb/Pr Component signal XIPHD_INPUT_CVBS: Composite video signal XIPHD_INPUT_Svideo: S-terminal signal		

3.8.24 XIPHD_SetInputType

Function Name	XIPHD_SetInputType		
Parameter	hXIProperty	Property handle	
	nInputType	Signal type to be set.	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Set the current input signal type. Refer to XIPHD_GetInputType .		

3.8.25 XIPHD_GetFlipMode

Function Name	XIPCVBS_GetFlipMode		
Parameter	hXIProperty	Property handle	
	pbFlipHorz	Whether the returned screen is horizontal rotation or not (Mirror)	
	pbFlipVert	Whether the returned screen is vertical rotation or not (Reverse)	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get the current output of the screen horizontally and vertically reversed state. Through the reversal settings of the screen, you can flip reversed camera screen to the normal screen.		

3.8.26 XIPHD_SetFlipMode

Function Name	XIPCVBS_SetFlipMode		
Parameter	hXIProperty	Property handle	
	pbFlipHorz	Whether the returned screen is horizontal rotation or not (Mirror)	
	pbFlipVert	Whether the returned screen is vertical rotation or not (Reverse)	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Set the current output of the screen horizontally and vertically reversed state. Refer to XIPHD_GetFlipMode .		

3.8.27 XIPHD_GetRGBComponentProcParam

Function Name	XIPHD_GetRGBComponentProcParam		
Parameter	hXIProperty	Property handle	
	pnRedBrightness	Return the red luminance offset which ranges from -100 to 100. The default is 0.	
	pnRedContrast	Return the red contrast offset which ranges from -100 to 100. The default is 0.	
	pnGreenBrightness	Return the green luminance offset which ranges from -100 to 100. The default is 0.	
	pnGreenContrast	Return the green contrast offset which ranges from -100 to 100. The default is 0.	
	pnBlueBrightness	Return the blue luminance offset which ranges from -100 to 100. The default is 0.	
	pnBlueContrast	Return the blue contrast offset which ranges from -100 to 100. The default is 0.	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get the current color adjustment settings.		

3.8.28 XIPHD_SetRGBComponentProcParam

Function Name	XIPHD_SetRGBComponentProcParam	
Parameter	hXIProperty	Property handle
	pnRedBrightness	Red luminance, -100 – 100, 0 means not change

	pnRedContrast	Red contrast, -100 – 100, 0 means not change	
	pnGreenBrightness	Green luminance, -100 – 100, 0 means not change	
	pnGreenContrast	Green contrast , -100 – 100, 0 means not change	
	pnBlueBrightness	Blue luminance , -100 – 100, 0 means not change	
	pnBlueContrast	Blue contrast , -100 – 100, 0 means not change	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Set the color adjustment options.		

3.8.29 XIPHD_GetDefClip

Function Name	XIPHD_GetDefClip		
Parameter	hXIProperty	Property handle	
	pnLeft	Return the number of pixels that need resection to the left side of the screen.	
	pnTop	Return the number of pixels that need resection to the top side of the screen.	
	pnWidth	Return the width of the screen after resection	
	pnHeight	Return the height of the screen after resection	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get input signal's default clipping region. The capture card cut locale with different input signal formats (such as different resolution VGA signal).		

3.8.30 XIPHD_GetClip

Function Name	XIPHD_GetClip		
Parameter	hXIProperty	Property handle	
	pnLeft	Return the number of pixels that need resection to the left side of the screen.	
	pnTop	Return the number of pixels that need resection to the top side of the screen.	
	pnWidth	Return the width of the screen after resection	
	pnHeight	Return the height of the screen after resection	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get input signal's clipping region. The capture card cut locale with different input signal formats (such as different resolution VGA signal).		

3.8.31 XIPHD_SetClip

Function Name	XIPHD_SetClip		
Parameter	hXIProperty	Property handle	
	nLeft	Starting position relative to the left of original signal.	
	nTop	Starting position relative to the top of original signal.	
	nWidth	Return the width of the screen after resection	
	nHeight	Return the height of the screen after resection	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Set clipping region. The capture card cut locale with different input signal formats (such as different resolution VGA signal). Effectively cut regional is 160x120 —the original signal size returned by XIPHD_GetSignalFormat.		

3.8.32 XIPHD_GetSamplingPhaseRange

Function Name	XIPHD_GetSamplingPhaseRange		
Parameter	hXIProperty	Property handle	
	pbyPhaseMin	The minimum value of the sampling phase	
	pbyPhaseMax	The maximum value of the sampling phase	
	pbyPhaseDef	The default value of the sampling phase	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get the effective ranges and default values of sampling phase of the current A / D converter. Only for VGA / RGB signal acquisition.		

3.8.33 XIP_GetSamplingPhase

Function Name	XIPHD_GetSamplingPhase		
Parameter	hXIProperty	Property handle	
	pbyPhase	Return to the current sampling phase	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get the current A / D converter sampling phase. Only for VGA / RGB signal acquisition.		

3.8.34 XIPHD_SetSamplingPhase

Function Name	XIPHD_SetSamplingPhase		
Parameter	bXIProperty	Property handle	
	bAutoDetectPhase	Whether to trigger phase detection function automatically. If it is TRUE, ignore the parameter byPhase. Otherwise, set the parameters byPhase manually.	
	byPhase	Sampling phase which should be within the effective range returned by XIPHD_GetSamplingPhaseRange.	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Set the current A/D converter sampling phase Only for VGA/RGB signal acquisition		

3.8.35 XIPHD_GetSamplingCountRange

Function Name	XIPHD_GetSamplingCountRange		
Parameter	hXIProperty	Property handle	
	pcxSamplingCountMin	The minimum number of sampling points on each line	
	pcxSamplingCountMax	The maximum number of sampling points on each line	
	pcxSamplingCountDef	The default number of sampling points on each line	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get the effective ranges and default numbers of sampling points of the current A / D converter. Only for VGA / RGB signal acquisition.		

3.8.36 XIPHD_GetSamplingCount

Function Name	XIPHD_GetSamplingCount		
Parameter	hXIProperty	Property handle	
	pcxSamplingCount	Return the number of sampling points on each line	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get the number of current A/D converter's sampling points of each line. Only for VGA/RGB signal acquisition		

3.8.37 XIPHD_SetSamplingCount

Function Name	XIPHD_SetSamplingCount		
Parameter	hXIProperty	Property handle	
	cxSamplingCount	Number of sampling points on each line which should be within the effective range returned by XIPHD_GetSamplingCountRange.	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Set the number of current A/D converter's sampling points of each line. Only for VGA/RGB signal acquisition.		

3.8.38 XIP_GetSerialNo

Function Name	XIP_GetSerialNo		
Parameter	hXIProperty	Property handle	
	pchSerialNo	Pointer to the unique hardware serial	
Return Value	HRESULT	S_OK	Calling successfully
		Other Value	Calling Error
Function Description	Get the unique hardware serial of capture device		

3.8.39 XIP_GetFeatureData

Function Name	XIP_GetFeatureData		
Parameter	hXIProperty	Property handle	
	wFeatureID	Feature data's ID	
	pbyFeatureData	Point of the feature data, at least 8 bytes	
Return Value	HRESULT	S_OK	Successfully
		Other value	error
Function Description	Get the feature data of capture device.		

3.8.40 XIPHD_IsHDMIPropertySupported

Function Name	XIPHD_IsHDMIPropertySupported		
Parameter	hXIProperty	Property handle	
Return Value	HRESULT	S_OK	HDMI property is supported

		S_FALSE	HDMI property is not supported
		Other value	error
Function Description	Get whether support HDMI property.		

3.8.41 XIPHD_GetHDMIInfoFrame

Function Name	XIPHD_GetHDMIInfoFrame		
Parameter	hXIProperty	Property handle	
	XIPHD_HDMI_INFO_FRAME_TYPE	Input the type of HDMI info frame	
	XIPHD_HDMI_INFO_FRAME	Return HDMI info frame	
Return Value	HRESULT	S_OK	Successfully
		Other value	error
Function Description	Get HDMI information frame of the specified type.		

3.9 Differences between using the 1st generation card and the 2nd generation card

SDK version 2.1 is compatible with the 1st generation card and the 2nd generation card. However, due to the differences of realizing, invoking some interfaces may lead to different results. The following lists them.

XIP_GetFeatureData	Not supported by the 2 nd generation cards. Return E_FAIL.
XIPHD_GetClip	The channels are independent with each other by using the 2 nd generation cards. There is no clip process in a newly opened channel. So all parameters return 0. We will not get valid clip params until setting clip. Different channels have their own clip parameters.
XIPHD_SetDeinterlaceType	XIPHD_DEINTERLACE_MOTION_ADAPTIVE is not supported by the 2 nd generation cards.
XIPHD_SetFlipMode	Not impemented with the 2 nd generation cards. Return E_NOTIMPL.
XIPHD_SetRGBComponentProcParam	Not impemented with the 2 nd generation cards. Return E_NOTIMPL.