SDK 程序开发手册
Developer’s Guide

Version:3.8.29.0

Date:2019/10

Write by TSW

本版本 SDK 开发包适用T100高清系列/标清系列采集卡

本SDK所有函数功能接口定义在DXMediaCap.h中，数据结构定义在datastru.h中。SDK动态库名为DXMediaCap.dl。调用编译本SDK VC demo为VC6需安装Direct9.0 SDK或以上。VB工程为VB6.0，delphi工程为DELPHI7，Csharp工程为vs2010.

本sdk适用同三维视频采集设备，程序开发中注意区分高清和标清调用参数会有些差异。调用要注 意的有下面几点。

1、 通过得到设备名称或是否是高清卡函数接口判断是否是高清卡。这里要设置一个bool值全局变量，以便后面区分高清和标清参考代码如下：调用函数接口为**DXGetDeviceName**或**DXDeviceIsHD**

```
//////////get device name 如果是高清设备,设备名称是HD开头或其他自己添加
CString  strDeviceName;
char* cDevName;
cDevName=DXGetDeviceName(DeviceInfo[i].dwChannelHandle);
strDeviceName=cDevName;
strDeviceName = strDeviceName.Left(2);
if(strDeviceName.Compare("HD") == 0 )
{
    DeviceInfo[i].dwHDdevice=true;//是否高清卡变量
}
```

2、 高清卡在opendevice后可以调用得到信号状态和设置视频输入端口。标清必须在DXDeviceRun函数后才可以调用。参考代码如下

```
DeviceInfo[i].dwChannelHandle =m_DeviceControl.OpenDevice (i,
&DeviceInfo[i].dwOpenDevState );
//////////get device name 如果是高清设备,设备名称是HD开头
CString  strDeviceName;
char* cDevName;
cDevName=DXGetDeviceName(DeviceInfo[i].dwChannelHandle);
strDeviceName=cDevName;
strDeviceName = strDeviceName.Left(2);
////高清设备得到设备的信号状态，此函数只适用高清
if(strDeviceName.Compare("HD") == 0 )
{
    DXGetVideoParaHD(DeviceInfo[i].dwChannelHandle,
                                    DeviceInfo[i].dwColorspace,
                                    DeviceInfo[i].dwWidth,
                                    DeviceInfo[i].dwHeight,
                                    DeviceInfo[i].dwFrameRate);
    DXSetVideoSourceEx(DeviceInfo[i].dwChannelHandle,GetVideoSource); //设置高
                                                                    清卡视频输入
    DeviceInfo[i].dwHDdevice=true;
}
```

3、 高清卡录像编码处理。目前不管是用gpu编码或软编码i5的机器都不能保证实时录像帧率到不了60帧，所以如果输入帧率大于或等于50帧的话录像帧率要丢一半处理。参考代码：

```
if(nCodecPara.fps>30)
{
     nCodecPara.fps=int( nCodecPara.fps/2);
     vidRatio=2; //降一半帧率
}
else
{
     vidRatio=1;
}

 DXSetVideoCodecPara(ChannelHandle, codec_x264, &nCodecPara);
```

4、高清和标清编码码流要单独处理。高清像素大必须保证数据码流到4mb以上，图像质量才是ok的

```
//如果是高清卡编码码流要设置大些
if(DeviceInfo[dwCard].dwHDdevice==true)
{
        DeviceInfo[dwCard].dwVideoCodecpara.Bitrate= 5000;  /
        DeviceInfo[dwCard].dwVideoCodecpara.Maxrbps=8000;
        DeviceInfo[dwCard].dwVideoCodecpara.Peekbps=5000;
 }
else
{
    DeviceInfo[dwCard].dwVideoCodecpara.Bitrate= 1200;
    DeviceInfo[dwCard].dwVideoCodecpara.Maxrbps=4000;
    DeviceInfo[dwCard].dwVideoCodecpara.Peekbps=2000;
}
```

5、图像算法设置差异

说明：DXEnableDeinterlace，DXEnableDenoise，DXEnableSharpen，DXFlipVideo，DXMirrorVideo，DXEnDataRaterPara这个几个函数都是通过软件算法实现，高清卡调用像素太高，处理不了那么高的数据会造成帧率丢失。DXSetFilpHD，DXSetMirrorHD，DXSetGainHD这个是对应高清卡的设置函数接口，算法直接在fpga里面实现不占用cpu资源。

SDK函数升级说明

//////////////////////

增加对Nvidia显卡gpu编码功能：编码算法定义为#define NVIDIACODEC_FILTER "nvidia Codec\0"

2019/10升级说明

1、 增加osd画矩形框的函数接口

```
unsigned __stdcall DXSetRectOSD( device_handle device,
  int x,
  int y,
  int width,
  int height,
  int RectNO,
  int lineSize,
```

```
        int lineStyle = style_solid,
        COLORREF color  = RGB(0,0,255),
        unsigned char alpha = 255);
```

说明：设置矩形框 OSD 叠加参数

参数：device - [in] 设备句柄

x - [in] 指定矩形区域的 X 标

y - [in] 指定矩形区域的 Y 坐标

width - [in] 指定矩形区域的宽度

height - [in] 指定矩形区域的高度

int RectNO：叠加矩形框序号

int lineSize - [in]矩形框的线宽

int lineStyle = style_solid：矩形框的线类型

COLORREF color = RGB(0,0,255)：矩形框的颜色

unsigned char alpha = 255：透明度值：0-255 255 不透明

返回值：0 - 成功；失败则返回错误代码

2、 增加osd画线的函数接口。可通过本函数画十字线

```
unsigned __stdcall DXSetLineOSD(device_handle device,
                                int x0,
                                int y0,
                               int x1,
                                int y1,
                                int LineNO,
                                int lineSize,
                                int lineStyle = style_solid,
                                COLORREF color  = RGB(0,0,255),
                                unsigned char alpha = 255)
```

说明：设置画线 OSD 叠加参数

参数：device - [in] 设备句柄

X0 - [in] 指定画线的开始端点 X 标

Y0 - [in] 指定画线的开始端点 Y 坐标

X1 - [in] 指定画线的结束端点 X 标

Y1 - [in] 指定画线的结束端点 Y 坐标

int LineNO：叠加直线线序号

int lineSize - [in]：线宽

int lineStyle = style_solid：画线类型

COLORREF color = RGB(0,0,255)：画线的颜色

unsigned char alpha = 255：透明度值：0-255 255 不透明

返回值：0 - 成功；失败则返回错误代码

3、 BOOL __stdcall DXVideoCodecFilterIsSupport(const char* codecFilter);

说明：检查视频编码器

参数：codecFilter--[in]编码器的名称,如:INTEL264CODEC_FILTER

返回值：TRUE - 成功（支持）；FALSE 失败

//////////////////

2015/10升级说明

1、升级视频编码器，将原来内嵌的开源x264编码器取消。新sdk内嵌XVID和intel编码器，intel编码器支持硬件GPU编码，如系统中cpu支持硬件GPU功能sdk会自动启用GPU编码不占系统资源，目前不支持xp系统下INTEL编码器编码。

1、unsigned DXSetVideoCodecPara(device_handle device, unsigned& codecType, void* pPara);

说明：设置视频 sdk 内嵌编码器属性

参数：device - [in] 设备句柄

codecType--[in]编码器的类型,H264 名称为 codec_x264，XVID 编码器为 codec_xvid

pPara - [in]编码器的具体属性，NULL - 表示使用默认属性，传入结构体 VidCodecX264Para

```
typedef struct{int fps;     视频帧率
        int keyframeinterval; 关键帧间隔
        int rcMode;     //码率控制:codec_CBR(恒定码率)/codec_VBR(平均码率);
        int Quality; 编码质量， 1 ~ 31 (1 为最小量度，压缩质量最好)
        int Bitrate      编码码率，单位：kbps）默认值 256，x264 中为 0 时，编码器内
                         部自己计算
        int Maxrbps;   //最大码流 VBR 有效位率范围:56bps ~ 10Mbps; 单位 Kbps
        int Peekbps;   //编码峰值 BR 有效，位率范围:56bps ~ 10Mbps; 单位 Kbps
        }VidCodecX264Para;
```

返回值：0 - 成功；失败则返回错误代码

//

DLL函数说明

一：错误代码及说明

1、RETERRNO：错误类型定义

```
typedef enum
{
 RET_NOERROR = 0,          /* no error */
 RET_EXCEPTION,            /* sdk exception error */
 RET_ERR_INIT,             /* do not initialize or initialize failed */
 RET_INVALID_DEV,          /* do not find valid media device */
 RET_INVALID_CHANNEL,      /* invalid channel */
 RET_BAD_POINTER,          /* invalid or Null pointer param */
 RET_NO_MATCH,             /* do not match the compatiple object  */
 RET_DX_FAILED,            /* fail from direct show */
 RET_INVALID_HANDLE,       /* find null handle */
 RET_INVALID_PARAM,        /* invalid function param */
 RET_GDI_ERROR,            /* GDI error */
 RET_IO_ERROR,             /* common io read or write error */
 RET_FAILED,               /* common error */
 RET_VIDFMT_ERROR,         /* Video format selected do not support */
}RETERRNO;
```

二：数据类型及结构体定义

2.1 state 设备运行状态

enum

{state_stopped, 停止状态

state_paused, 暂停状态

state_running}; 运行状态
2.1 去隔行算法
enum
{ di_none, 取消隔行算法
 di_blend, 设置 bob 隔行算法
 di_bob, }; 设置 tom 隔行算法
2.2 设置降噪算法
enum {dn_none, dn_b};

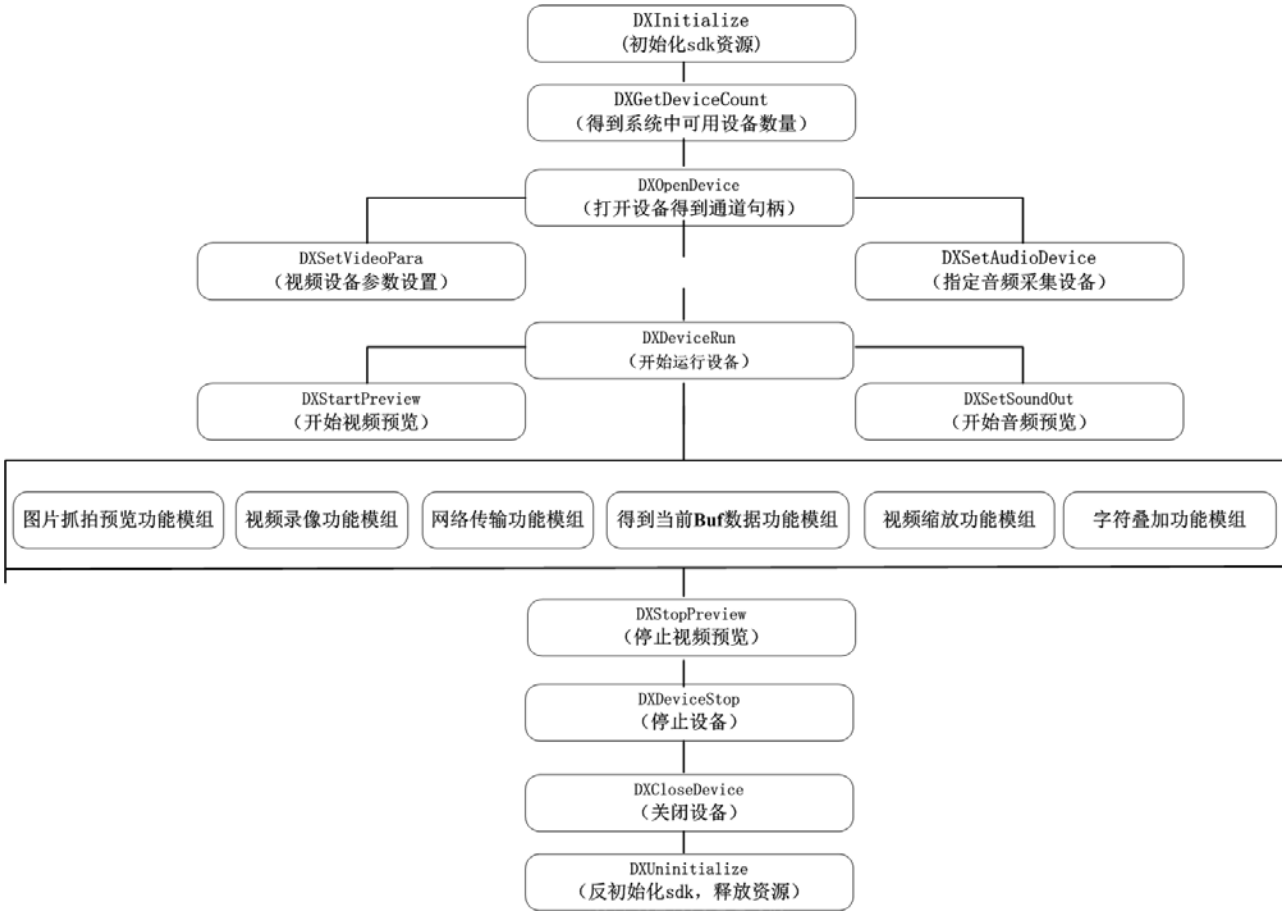
2.3 视频颜色空间
enum {cs_rgb24, RGB24 视频流格式
      cs_rgb32, RGB32 视频流格式
      cs_yuy2}; yuv2 视频流格式

2.4 设备属性页
enum {attr_video, attr_audio, attr_encode};

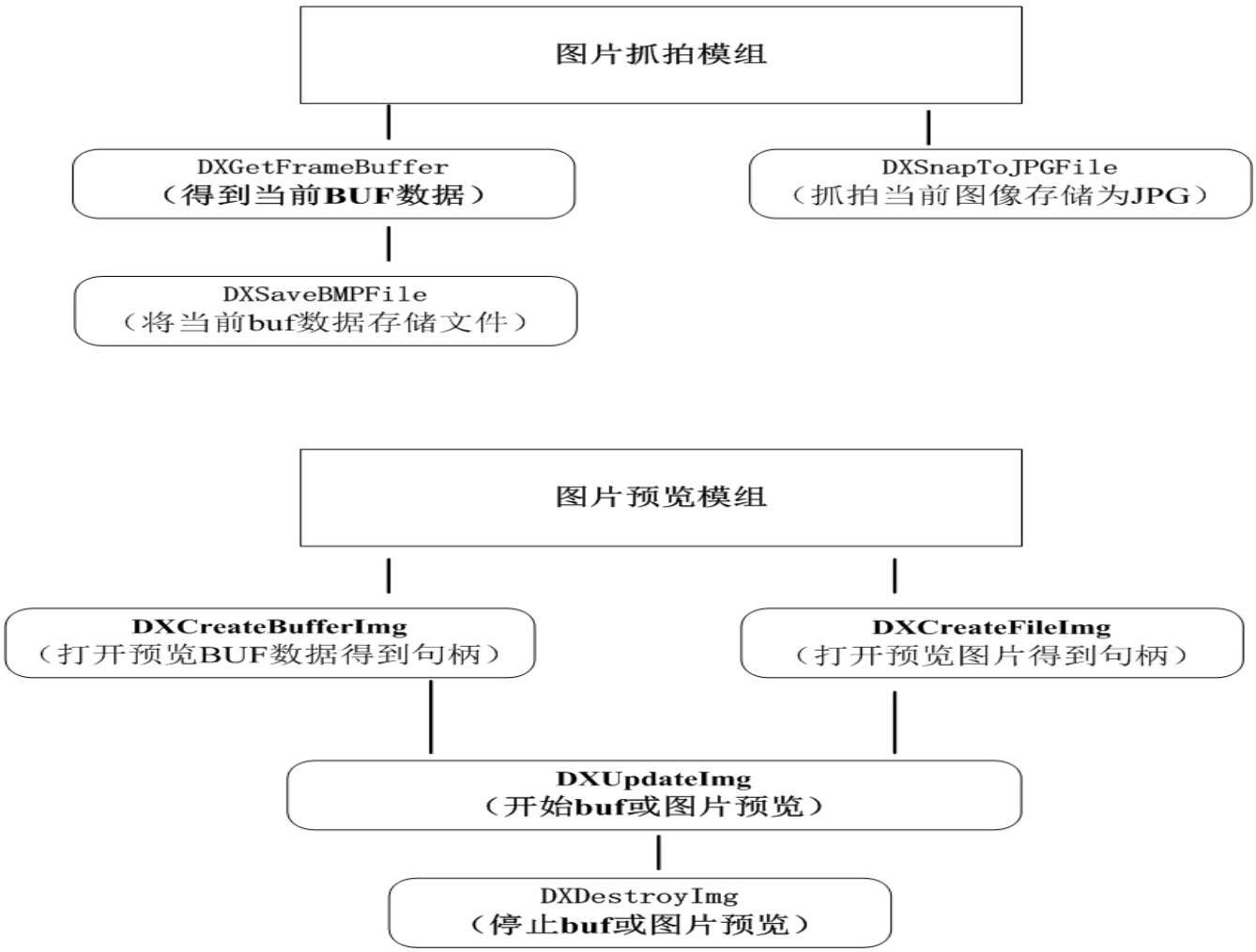
2.5 设备信息
typedef struct _device_tag
{
    unsigned idx;   // 过滤器在枚举过程中的序号
    TCHAR   deviceName[MAX_DEVICE_NAME];// 过滤器的名称
} DEVICE_TAG, *PDEVICE_TAG;

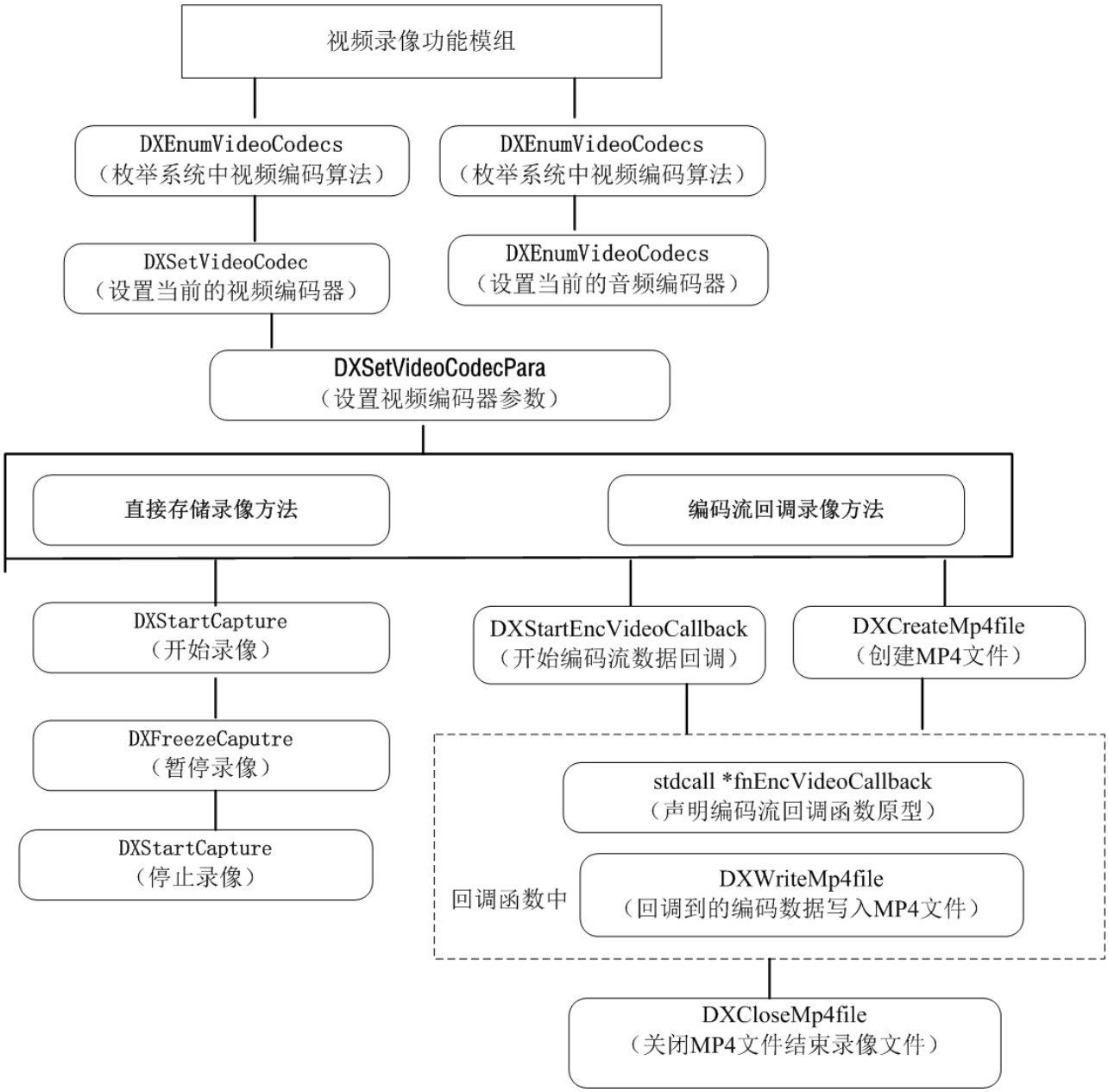
三：函数说明调用顺序：

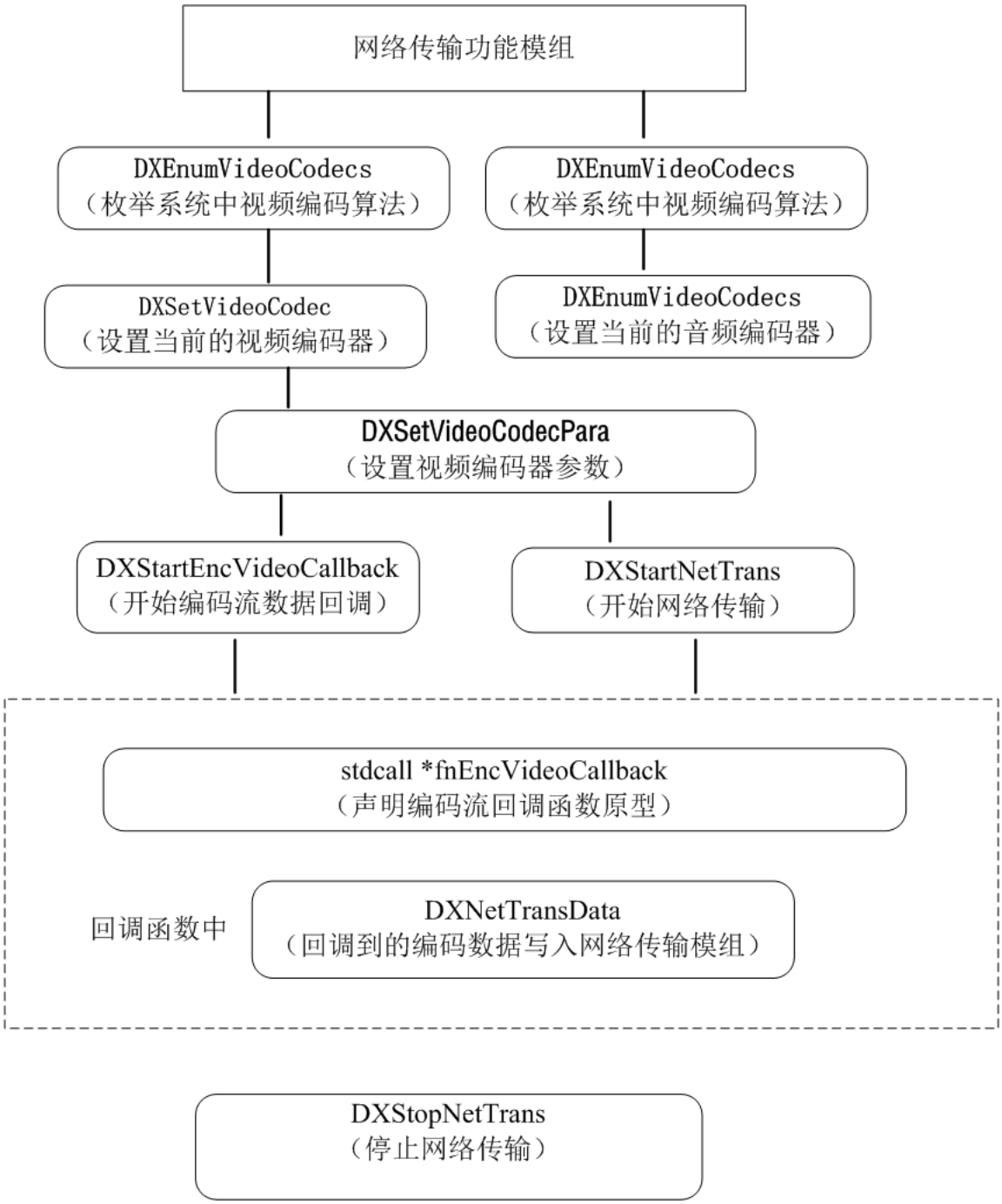
主函数调用流程



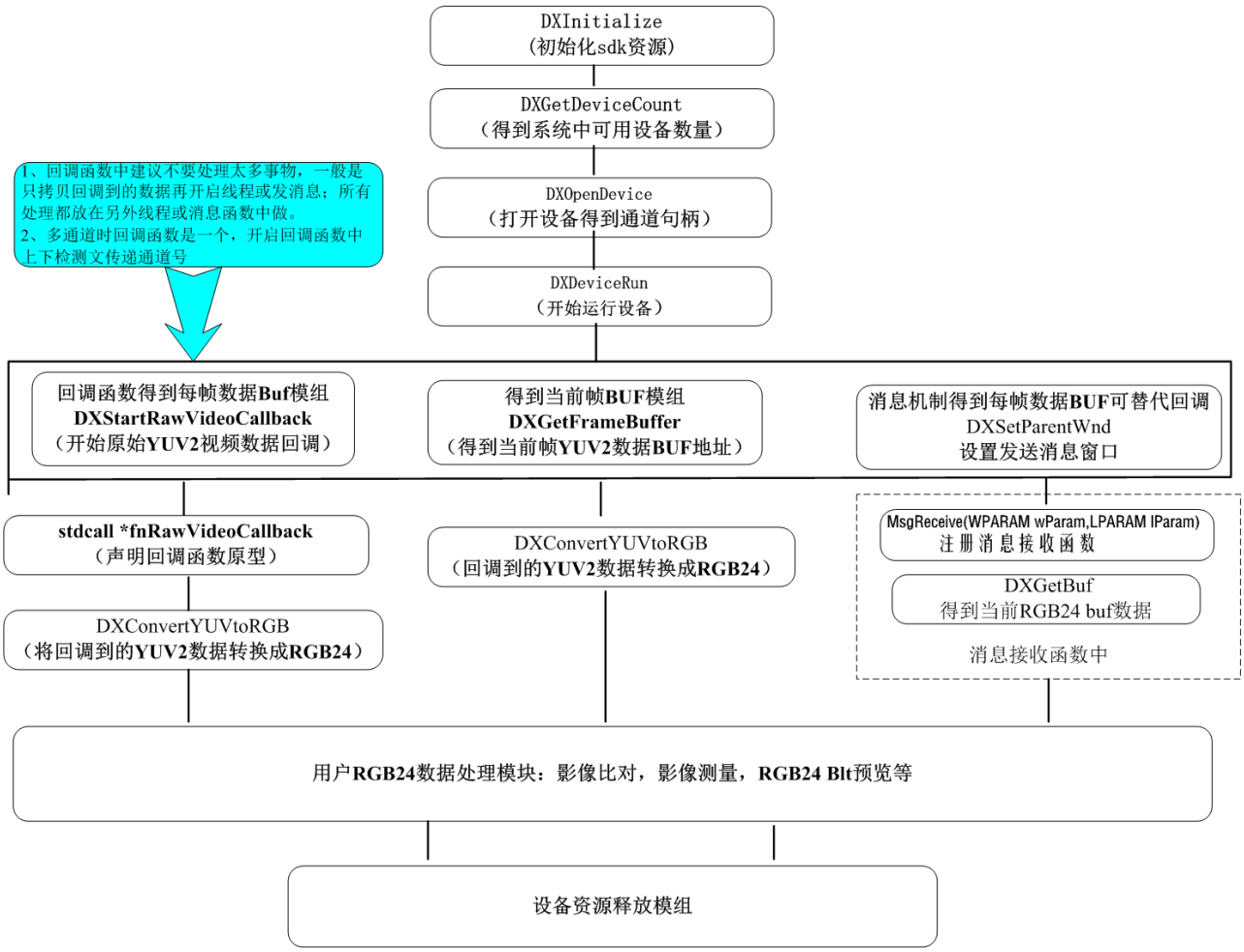
功能模组调用流程：


图片抓拍模组
DXGetFrameBuffer
（得到当前BUF数据）
DXSnapToJPGFile
（抓拍当前图像存储为JPG）
DXSaveBMPFile
（将当前buf数据存储文件）
图片预览模组
DXCreateBufferImg
（打开预览BUF数据得到句柄）
DXCreateFileImg
（打开预览图片得到句柄）
DXUpdateImg
（开始buf或图片预览）
DXDestroyImg
（停止buf或图片预览）


视频录像功能模组
DXEnumVideoCodecs
（枚举系统中视频编码算法）
DXEnumVideoCodecs
（枚举系统中视频编码算法）
DXSetVideoCodec
（设置当前的视频编码器）
DXEnumVideoCodecs
（设置当前的音频编码器）
DXSetVideoCodecPara
（设置视频编码器参数）
直接存储录像方法
编码流回调录像方法
DXStartCapture
（开始录像）
DXStartEncVideoCallback
（开始编码流数据回调）
DXCreateMp4file
（创建MP4文件）
DXFreezeCaputre
（暂停录像）
stdcall *fnEncVideoCallback
（声明编码流回调函数原型）
回调函数中
DXWriteMp4file
（回调到的编码数据写入MP4文件）
DXStartCapture
（停止录像）
DXCloseMp4file
（关闭MP4文件结束录像文件）



不开始视频预览调用函数流程



四：函数说明

4.1设备的初始化及卸载

4.1.1 unsigned __stdcall DXInitialize()

说明：初始化SDK资源

参数：无

返回值：正常为0

4.1.2 void __stdcall DXUninitialize()

说明：反初始化SDK资源

参数：无

4.2设备信息的获取

4.2.1 unsigned __stdcall DXGetDeviceCount()

说明：获取初始化成功的设备总数

返回值：　匹配的设备总数

4.2.2 unsigned __stdcall DXEnumVideoDevices(PDEVICE_TAG devTags, unsigned& num)

说明：枚举系统中可用的视频捕捉设备

参数：PDEVICE_TAG devTags用于返回获得的设备信息的数组包括设备名称和序号

Num返回枚举到得有效视频设备数量

返回值：正常为0

4.2.3 unsigned __stdcall DXEnumAudioDevices(PDEVICE_TAG devTags, unsigned& num)

说明：枚举系统中可用的音频捕捉设备

参数：PDEVICE_TAG devTags用于返回获得的设备信息的数组包括设备名称和序号

Num返回枚举到得有效设备数量

返回值：正常为0

4.2.3 unsigned __stdcall DXEnumAudioDevices(PDEVICE_TAG devTags, unsigned& num)

说明：枚举系统中可用的音频捕捉设备

参数：PDEVICE_TAG devTags用于返回获得的设备信息的数组包括设备名称和序号

Num返回枚举到得有效设备数量

返回值：正常为0

4.2.4 char* __stdcall DXGetDeviceName(device_handle device);

说明：获取当前视频设备名称

参数：device_handle device：视频设备通道句柄

返回值：返回当前视频设备名称

4.2.5 BOOL __stdcall DXDeviceIsHD (device_handle device);

说明：判断当前设备是否是高清设备

参数：device_handle device：视频设备通道句柄

返回值：TRUE:设备为高清采集设备

4.2.6 BOOL __stdcall DXDeviceIsUVC (device_handle device)

说明：判断当前视频设备是否UVC设备

参数：device_handle device：视频设备通道句柄

返回值：TRUE:设备为高清采集设备

4.2.7 unsigned __stdcall DXCreatePropertyPage(device_handle device,
int nPropertyType,
HWND hWndOwner,
unsigned x,
unsigned y,
const char *szCaption);

说明：device_handle device：视频设备通道句柄

参数：nsigned idx,视频卡号

nPropertyType ：属性页类型PropertyDlg_VideoCaptureFilter = 0
PropertyDlg_VideoCapturePin=1
PropertyDlg_VideoCrossbar=2

HWND hWndOwner:属性页显示窗口句柄

unsigned x,：显示位置X坐标

unsigned y, 显示位置Y坐标

const char *szCaption:窗体名称

返回值：正常为0

4.3视频设备的开启和预览

4.3.1 device_handle __stdcall DXOpenDevice(unsigned idx, unsigned* err = NULL)

说明：打开视频设备

参数：nsigned idx,视频卡号

unsigned* err = NULL 打开设备状态错误信息

返回值：设备的通道句柄，否则返回 NULL。可通过 err 获得错误代码

4.3.2 void __stdcall DXCloseDevice(device_handle device)
说明：关闭视频设备
参数：device_handle device：视频设备通道句柄
返回值：正常为0
4.3.3 unsigned __stdcall DXDeviceRun(device_handle device)
说明：开始运行设备
参数：device_handle device：视频设备通道句柄
返回值：正常为0
4.3.4 unsigned __stdcall DXDevicePause(device_handle device)
说明：暂停运行设备
参数：device_handle device：视频设备通道句柄
返回值：正常为0

4.3.5 unsigned __stdcall DXDeviceStop(device_handle device)
说明：停止运行设备
参数：device_handle device：视频设备通道句柄
返回值：正常为0

4.3.6 unsigned __stdcall DXGetDeviceState (device_handle device)
说明：得到设备运行状态
参数：device_handle device：视频设备通道句柄
返回值：设备运行状态：run/pause/stop

4.3.7 unsigned __stdcall DXStartPreview(device_handle device, HWND hWnd, PRECT rect, unsigned vmrtype);
说明：开始视频预览
参数：device_handle device：视频设备通道句柄
HWND hWnd：视频预览窗体
PRECT rect：指定视频预览的矩形区域
unsigned vmrtype：视频预览的模式：
值=0 VMR7开始标准Directhsow视频预览直接调用微软预览filter，多路时CPU占用高兼容性比较好,但是新的win8或win10系统不支持，
值=1 VMR9解决个别系统vmr7创建不成功直接创建VMR9模式
值=2 D3D预览模式，CPU占用比较低，如显卡没有开启视频directDraw加速功能时视频预览失败。直接在运行处输入dxdiag命令可以查看显卡是否有开始directDraw加速
返回值：正常为0
4.3.8 unsigned __stdcall DXStopPreview(device_handle device);
说明：停止视频预览
参数：device_handle device：视频设备通道句柄
返回值：正常为0
4.3.3 unsigned __stdcall DXFreezePreview(device_handle device, BOOL bFreeze);
说明：暂停视频预览
参数：device_handle device：视频设备通道句柄
BOOL bFreeze：是否开启暂停预览
返回值：正常为0

4.4 视频设备相关参数设置

4.4.1 unsigned __stdcall DXGetVideoPara(device_handle device,
unsigned& standard,
unsigned& colorspace,
unsigned& width,
unsigned& height,
float& framerate)

说明：获取视频设备的视频属性。**此函数调用在DXDeviceRun之后即可生效**

参数：device - [in]视频设备通道句柄
standard - [out] 视频制式
colorspace - [out] 色彩空间，YUV2格式时值为2，占用2个字节
width - [out] 视频流宽度（单位：像素）
height - [out] 视频流高度（单位：像素）
framerate - [out] 视频帧率（单位：帧/秒）

返回值：0 - 成功；失败则返回错误代码

4.4.2 unsigned __stdcall DXSetVideoPara(device_handle device,
unsigned standard,
unsigned colorspace,
unsigned width,
unsigned height,
float framerate)

说明：设置视频设备的视频属性，**此函数调用要在DXOpenDevice之后和DXDeviceRun之前**

参数：device - [in]视频设备通道句柄
standard - [out] 视频制式
colorspace - [out] 色彩空间，YUV2格式时值为2，占用2个字节
width - [out] 视频流宽度（单位：像素）
height - [out] 视频流高度（单位：像素）
framerate - [out] 视频帧率（单位：帧/秒）

返回值：0 - 成功；失败则返回错误代码

4.4.1 unsigned __stdcall DXGetVideoParaHD (device_handle device,
unsigned& colorspace,
unsigned& width,
unsigned& height,
float& framerate)

说明：得到高清视频设备的视频属性。**此函数调用在DXOpenDevice之后即可生效**

参数：device - [in]视频设备通道句柄
standard - [out] 视频制式
colorspace - [out] 色彩空间，YUV2格式时值为2，占用2个字节
width - [out] 视频流宽度（单位：像素）
height - [out] 视频流高度（单位：像素）
framerate - [out] 视频帧率（单位：帧/秒）

返回值：0 - 成功；失败则返回错误代码

4.4.2 unsigned __stdcall DXSetVideoParaHD (device_handle device,
unsigned colorspace,
unsigned width,

unsigned height,
float framerate)

说明：设置高清设备视频设备的视频属性，**此函数调用要在DXOpenDevice之后和DXDeviceRun之前**

参数：device - [in]视频设备通道句柄

colorspace - [out] 色彩空间，YUV2格式时值为2，占用2个字节标清设备只支持YUV2数据流格式，高清可以支持到rgb24

width - [out] 视频流宽度（单位：像素）

height - [out] 视频流高度（单位：像素）

framerate - [out] 视频帧率（单位：帧/秒）

返回值：0 - 成功；失败则返回错误代码

4.4.3 unsigned __stdcall DXGetDisplayParaRange(device_handle device,
unsigned paraType,
long* pMin,
long* pMax,
long* pSteppingDelta,
long* pDefault,
long* pCapsFlags);

说明：获取显示属性的取值范围、默认值、步长、标志

参数：device - [in] 视频设备通道句柄

paraType - [in] 参数类型 0 - 亮度，1 - 对比度，2 - 饱和度，3 - 色度，4 - 锐度

pMin - [in] 最小值

pMax - [in] 最大值

pSteppingDelta - [in] 步长

pDefault - [in] 默认值

pCapsFlags - [in] 自动/手动标志

返回值：0 - 成功；失败则返回错误代码

4.4.4 unsigned __stdcall DXGetDisplayPara(device_handle device,
unsigned paraType,
long& value,
long& flags);

说明：获取设备的显示属性

参数：device - [in]视频设备通道句柄

paraType - [in] 参数类型，0 - 亮度，1 - 对比度，2 - 饱和度，3 - 色度，4 - 锐度

value - [out] 得到的视频显示属性参数值

flag - [out] 自动/手动标志

返回值：0 - 成功；失败则返回错误代码

4.4.5 unsigned __stdcall DXSetDisplayPara(device_handle device,
unsigned paraType,
long value,
long flags);

说明：设置设备的显示属性

参数：device - [in]视频设备通道句柄

paraType - [in] 参数类型，0 - 亮度，1 - 对比度，2 - 饱和度，3 - 色度，4 - 锐度

value - [out]视频显示属性参数值
flag - [out] 自动/手动标志
返回值：0 - 成功；失败则返回错误代码

4.4.6 unsigned __stdcall DXGetVideoSources(device_handle device,
unsigned* curSource,
unsigned* sources = NULL,
unsigned char* num = NULL);

说明：获取设备当前选择视频源输入端口和设备支持的所以视频源端口
参数：device - [in]视频设备通道句柄
curSource - [out] 当前选择的输入端子的序号。NULL - 忽略
sources - [out] 包含的输入端子的类型数组（比如：S-VIDEO、DV...）。NULL - 忽略
num - [in/out] 端子数量。NULL - 忽略，此时尽可获取当前选择的输入端子
返回值：0 - 成功；失败则返回错误代码

4.4.7 unsigned __stdcall DXSetVideoSource(device_handle device, unsigned source)
描述：设置视频设备的视频源输入端口，此函数接口适用标清采集卡
参数：device - [in]视频设备通道句柄
source - [in] 当前选择的输入端子的序号，1：AV1 2:AV2 3：SVIDEO
返回值：0 - 成功；失败则返回错误代码

4.4.7 unsigned __stdcall DXSetVideoSourceEx (device_handle device, unsigned source)
描述：设置视频设备的视频源输入端口，当前选择的输入端子的数组序号,具体设备类型参见通过DXGetVideoSources获取到的sources数组；
参数：device - [in]视频设备通道句柄
source - [in] 当前选择的输入端子的序号，下标0开始
返回值：0 - 成功；失败则返回错误代码

4.4.8 unsigned __stdcall DXGetSignalPresent(device_handle device, unsigned& signal)
说明：获取视频设备信号输入状态
参数：device - [in] 视频设备通道句柄
signal - [out] 信号状态。0 - 信号丢失；1 - 信号正常
返回值：0 - 成功；失败则返回错误代码

4.4.8、unsigned __stdcall DXEnumVideoCaps(device_handle device, PVIDEOCAPS vidCaps, unsigned& num)
说明：枚举设备中支持的视频尺寸
参数：
device - [in] 设备句柄
devTags - [out] 用于返回获得的过滤器的 TAG 的数组,传入结构体 videocaps

```
typedef struct _videocaps
{    int width   视频流宽度
     int height; 视频流高度
}VIDEOCAPS, *PVIDEOCAPS;
```

num - [in/out] 指定 vidCaps 数组的元素个数，返回时则是实际上获得的输出格式 Caps 的个数
返回值：0 - 成功；失败则返回错误代码

4.5 视频隔行降噪算法和视频图像处理

4.5.1 unsigned __stdcall DXEnableDeinterlace(device_handle device, unsigned deinterlace)

说明：视频去隔算法

参数：device - [in] 设备句柄

deinterlace - [in] 去隔行的算法类型 大于 3 取消隔行算法，取值范围为 0/1/2，默认值为 2

返回值：0 - 成功；失败则返回错误代码

4.5.2 unsigned __stdcall DXEnableDenoise(device_handle device, unsigned denoise)

说明：视频降噪算法

参数：device - [in] 设备句柄

denoise - [in] 去噪的算法值，取值范围 0-100. 0 取消此功能。默认一般为 50

返回值：0 - 成功；失败则返回错误代码

4.5.3 unsigned __stdcall DXEnableSharpen(device_handle device, unsigned deSharpness);

说明：视频锐化算法

参数：device - [in] 设备句柄

denoise - [in] 视频锐化算法值，取值范围 0-255. 0 取消此功能。

返回值：0 - 成功；失败则返回错误代码

4.5.4 unsigned __stdcall DXFlipVideo(device_handle device, BOOL flip)

说明：视频翻转功能

参数：device - [in] 设备句柄

flip - [in] TRUE - 启用翻转，FALSE - 禁用翻转

返回值：0 - 成功；失败则返回错误代码

4.5.5 unsigned __stdcall DXMirrorVideo(device_handle device, BOOL mirror)

说明：视频镜像功能

参数：device - [in] 设备句柄

mirror - [in] TRUE - 启用镜像，FALSE - 禁用镜像

返回值：0 - 成功；失败则返回错误代码

4.5.6、unsigned __stdcall DXEnDataRaterPara(device_handle device, unsigned RateType, void* pPara);

说明：视频亮度对比度增强算法

参数：device - [in] 设备句柄

unsigned RateType [in] 视频亮度对比度增强值。亮度取值（-255-255）对比度（-100-100）

void* pPara：传入结构体 DRateColorPara //亮度对比度结构体：

```
typedef struct{int nBrightness; // 亮度增强
               int nContrast;   //对比度增强
               }DRateColorPara;
```

4.5.7 unsigned __stdcall DXSetFilpHD (device_handle device, BOOL flip)

说明：高清视频翻转功能，算法在 fpga 硬件中实现

参数：device - [in] 设备句柄

flip - [in] TRUE - 启用翻转，FALSE - 禁用翻转

返回值：0 - 成功；失败则返回错误代码

4.5.8 unsigned __stdcall DXSetMirrorHD (device_handle device, BOOL mirror)

说明：高清视频镜像功能算法在 fpga 硬件中实现

参数：device - [in] 设备句柄

mirror - [in] TRUE - 启用镜像，FALSE - 禁用镜像

返回值：0 - 成功；失败则返回错误代码

4.5.9 unsigned __stdcall DXSetGainHD(device_handle device,unsigned char nGainType,unsigned char

chValue);

说明：高清视频 rgb 三色增益调整，算法在 fpga 硬件中实现

参数：device - [in] 设备句柄

char nGainType - [in] rgb 三色增益类型： R:0 G:1 B:2

unsigned char chValue - [in] 设置的增益取值范围 0-128

返回值：0 - 成功；失败则返回错误代码

4.8.8 unsigned __stdcall DXConvertYUVtoRGB(device_handle device, void* pYUVBuf,
void* pRGBBuf,
long lImgWidth,
long lImgHeight,
BOOL bInverted,
BOOL bInvertColor);

说明:视频数据格式转换

参数：device_handle device 设备通道句柄

void* pYUVBuf：输入yuv2 buf指针

void* pRGBBuf, 输出RGB24 buf指针

long lImgWidth, 图片的宽度（单位：像素）

long lImgHeight, 图片的高度（单位：像素）

BOOL bInverted, 是否上下翻转

BOOL bInvertColor) 是否反色

返回值：0 – 成功；失败则返回错误代码

4.8.9 unsigned _DXYuy2Gray(unsigned char *src, unsigned char *dst, unsigned pixelsPerLines, unsigned width, unsigned height);

说明：yuv2 转换成 8 位灰度图。

参数：unsigned char *src, //输入源 YUV2 buf 地址

unsigned char *dst //转换 GRAY buf 地址

unsigned pixelsPerLines, 每行像素大小：width×2

unsigned width：输入源视频宽度

unsigned height：输入源视频高度。

4.8.10 unsigned _ DXYUYRotate90 (unsigned char *src, unsigned char *dst, unsigned pixelsPerLines, unsigned width, unsigned height);

说明：yuv2 转换成 8 位灰度图。

参数 char *dst - [out]视频目的地址

unsigned char * - [in]视频源地址

unsigned width, - [in]视频宽度

unsigned height - [in]视频高度

bRight - [in]true 顺时针,false 逆时针

nColorspace - [in]颜色空间，目前只支持 cs_rgb24 和_device_tag

返回值：

0 - 成功；失败则返回错误代码

unsigned __stdcall DXEnFScale(device_handle device,bool bEnFScale, RECT* rect, SwsAlogrithm enAlogrithm=SWS_SA_FAST_BILINEAR);

说明：开启视频局部放大算法，本算法是通过软件算法来实现的。

参数：device_handle device - [in] 设备句柄

bool bEnFScale - [in 是否开启缩放

RECT* rect,--[in]电子放大局部区域

enAlogrithm - [in]Scale 算法类型，默认双线性插值

```
enum SwsAlogrithm
{
    SWS_SA_FAST_BILINEAR = 0,
    SWS_SA_BILINEAR          ,
    SWS_SA_BICUBIC           ,
    SWS_SA_X                 ,
    SWS_SA_POINT             ,
    SWS_SA_AREA              ,
    SWS_SA_BICUBLIN          ,
    SWS_SA_GAUSS             ,
    SWS_SA_SINC              ,
    SWS_SA_LANCZOS           ,
    SWS_SA_SPLINE            ,
};
```

返回值：0 - 成功；失败则返回错误代码

unsigned __stdcall DXStartRawVidScale(VidScalePara * pScalePara, scale_handle& pscalehad);

说明：开启原始视频流的缩放

参数：

VidScalePara * pScalePara - [in]缩放初始化参数

```
typedef struct{unsigned int src_stride;
               unsigned int src_width;
               unsigned int src_height;
               unsigned int dst_stride;
               unsigned int dst_width;
               unsigned int dst_height;
               SwsAlogrithm algorithm;
               }VidScalePara;
```

pscalehad - [out]返回缩放设备句柄

返回值：0 - 成功；失败则返回错误代码

unsigned __stdcall DXRawVidScale(scale_handle scalehad, unsigned char* src, unsigned char* dst);

说明：视频缩放

参数：scale_handle scalehad [in]返回缩放设备句柄

unsigned char* src, [in]输入需要缩放数据的 buf

unsigned char* dst [out]输出缩放后数据的 buf

unsigned __stdcall DXStopRawVidScale(scale_handle scalehad);

说明：停止视频缩放处理

参数：scale_handle scalehad [in]返回缩放设备句柄

返回值：0 - 成功；失败则返回错误代码

4.6声音设备相关信息设置

4.6.1 unsigned __stdcall DXHasAudio(device_handle device, BOOL& bHasAudio)

说明：检查设备是否包含音频
参数：device_handle device：视频设备通道句柄
BOOL& bHasAudio：输出是否设备包含音频信息，true自己包含
返回值：正常为0

4.6.2 unsigned __stdcall DXSetAudioDevice(device_handle device, PDEVICE_TAG audioDevice = NULL)

说明：为设备分配音频采集设备。注意：!!! 只能在 state_stopped 状态下调用，如设备自己已包含音频设备可不调
用本函数
参数：device_handle device：视频设备通道句柄
PDEVICE_TAG audioDevice = NULL枚举到得音频设备信息
返回值：正常为0

4.4.3 unsigned __stdcall DXSetSoundOut(device_handle device, BOOL bSound)

说明：开启音频设备 注意：!!!只有开始了声音输出，静音操作、音量调节、录像时包含音频等操作才能成功用本函数
参数：device_handle device：视频设备通道句柄
BOOL bSound：是否开启音频设备，TRUE - 开始声音输出，FALSE - 终止声音输出
返回值：正常为0

4.6.4 unsigned __stdcall DXEnableMute(device_handle device, BOOL bMute)

说明：开启和停止声音设备静音
参数：device_handle device：视频设备通道句柄
BOOL& bHasAudio：TRUE - 静音，FALSE - 非静音
返回值：正常为0

4.6.5 unsigned __stdcall DXSetAudioVolume(device_handle device,
unsigned char volume,
unsigned char* const balance = NULL)

说明：设置音量和平衡
参数：device - [in] 设备句柄
volume - [in] 音量大小，取值范围(0, 100]
balance - [in] 左右平衡，取值范围[-10, 10]，负数表示左声道强，正数表述右声道强，0 - 左右平衡
注意：!!! NULL - 表示不设置平衡
返回值：0 - 成功；失败则返回错误代码，一般失败原因是因为没有开始声音输出

4.6.6 unsigned __stdcall DXGetAudioVolume(device_handle device, unsigned char* volume, unsigned char* balance = NULL)

说明：获取音量和平衡
参数：device - [in] 设备句柄
volume - [out] 音量大小，取值范围(0, 100]，注意：!!! NULL - 表示不获取音量
balance - [out] 左右平衡，取值范围[-10, 10]，负数表示左声道强，正数表述右声道强，0 - 左右平衡。注意：!!! NULL - 表示不获取平衡
返回值：0 - 成功；失败则返回错误代码，一般失败原因是因为没有开始声音输出

4.7 字符OSD叠加设置

本类函数接口主要包含osd叠加参数设置和OSD叠加功能的实现

4.7.1 unsigned __stdcall DXSetTimeOSD(/*in*/ device_handle device,
/*in*/ int x,

/*in*/ int y,
/*in*/ int pointSize,
/*in*/ char* faceName = "Arial",
/*in*/ COLORREF color = RGB(0,0,255),
/*in*/ COLORREF bgcolor = RGB(0, 0, 0),
/*in*/ BOOL transparent = TRUE

说明：设置时间 OSD 叠加参数

参数：device - [in] 设备句柄

osdType - [in] OSD 的类型

x - [in] 指定 OSD 的 X 标

y - [in] 指定 OSD 的 Y 坐标

int pointSize - [in]时间 osd 字体大小

char* faceName：时间 osd 的字体类型

COLORREF color = RGB(0,0,255)：时间 osd 的字体颜色

COLORREF bgcolor = RGB(0, 0, 0)：时间 osd 的背景颜色

BOOL transparent = TRUE：是否透明叠加

返回值：0 - 成功；失败则返回错误代码

4.7.2unsigned __stdcall DXSetTextOSD(/*in*/ device_handle device,
/*in*/ int x,
/*in*/ int y,
/*in*/ int TextNO,
/*in*/ char* osdText,
/*in*/ int pointSize,
/*in*/ char* faceName = "Arial",
/*in*/ COLORREF color = RGB(0,0,255),
/*in*/ COLORREF bgcolor = RGB(0, 0, 0),
/*in*/ BOOL transparent = TRUE）

说明：设置文本 OSD 叠加参数

参数：device - [in] 设备句柄

osdType - [in] OSD 的类型

x - [in] 指定 OSD 的 X 标

y - [in] 指定 OSD 的 Y 坐标

int TextNO：文本 osd 的序号，我们可以支持任意个文本叠加，这个来标识文本 osd 的序号

char* osdText：叠加的文本信息

int pointSize - [in]时间 osd 字体大小

char* faceName：时间 osd 的字体类型

COLORREF color = RGB(0,0,255)：osd 的字体颜色

COLORREF bgcolor = RGB(0, 0, 0)：osd 的背景颜色

BOOL transparent = TRUE：是否透明叠加

返回值：0 - 成功；失败则返回错误代码

4.7.3unsigned __stdcall DXSetPictureOSD(/*in*/ device_handle device,
/*in*/ int x,
/*in*/ int y,
/*in*/ int PicNO,
/*in*/ char* picFileName,
/*in*/ BOOL transparent = TRUE,

/*in*/ unsigned char alpha = 255);

说明：设置文图片 OSD 叠加参数
参数：device - [in] 设备句柄
osdType - [in] OSD 的类型
x - [in] 指定 OSD 的 X 标
y - [in] 指定 OSD 的 Y 坐标
int PicNO：osd 叠加的序号，我们可以支持任意个图片叠加，这个来标识 osd 的序号
char* picFileName：叠加的图片名称信息
int pointSize - [in]时间 osd 字体大小
char* faceName：时间 osd 的字体类型
BOOL transparent = TRUE：是否透明叠加
unsigned char alpha = 255：调节图片叠加的透明度值范围 0-255.255 为不透明
返回值：0 - 成功；失败则返回错误代码

4.7.4unsigned __stdcall DXEnOSDDisp(device_handle device,
unsigned osdType,
int number,
BOOL enable);

说明：开启和取消 osd 叠加
参数：device - [in] 设备句柄
osdType - [in] OSD 的类型 0 表示时间叠加，1 表示文本叠加，2 表示图片叠加，3 表示直线叠加，4 表示矩形框叠加
int number - [in] 叠加 osd 序号，当 number = -1 时为全部
BOOL enable y - [in] 是否叠加 osd
返回值：0 - 成功；失败则返回错误代码

4.8 Dib位图的获取和图像格式转换

本类函数接口主要包括了位图buf数据的得到，存储，图片文件的抓拍以及DIB buf数据和图片文件的预览

4.8.1 unsigned __stdcall DXGetFrameBuffer(device_handle device,
unsigned char* buffer,
unsigned bufferLen,
unsigned* gotBufferLen = NULL,
unsigned* colorSpace = NULL,
unsigned* width = NULL,
unsigned* height = NULL,
unsigned* bytesWidth = NULL,
PRECT rect = NULL)

说明：得到当前帧的buf数据
参数：device - [in] 设备句柄
buffer - [in] 指向缓冲区。NULL - 后续参数返回相应的数据，比如，所需缓冲区的大小、颜色空间、视频尺寸
bufferLen - [in] 指向的缓冲区的大小（单位：字节）
gotBufferLen - [out] 实际获取的数据的大小（单位：字节）
colorSpace - [out] 获取的原始数据的色彩空间类型，YUV2:2, RGB24:3
width - [out] 获取的原始数据的像素宽度
height - [out] 获取的原始数据的像素高度

bytesWidth - [out] 获取的原始数据的字节宽度
rect - [in] 指定获取数据的区域
返回值：0 - 成功；失败则返回错误代码

4.8.2 unsigned __stdcall DXSaveBMPFile(TCHAR* szFileName,
unsigned char* buffer,
unsigned bufferLen,
unsigned colorSpace,
unsigned width,
unsigned height,
unsigned bytesWidth)

说明：将得到的当前Buf数据保存为 BMP文件
参数：szFileName - [in] 存储bmp的文件名
buffer - [in] 指向缓冲区
bufferLen - [in] 指向的缓冲区的大小（单位：字节）
colorSpace - [in] 获取的原始数据的色彩空间类型
width - [in] 获取的原始数据的像素宽度
height - [in] 获取的原始数据的像素高度
bytesWidth - [in] 获取的原始数据的字节宽度
返回值：0 - 成功；失败则返回错误代码

4.8.3 unsigned __stdcall DXSaveJPGFile(TCHAR* szFileName,
unsigned char* buffer,
unsigned bufferLen,
unsigned colorSpace,
unsigned width,
unsigned height,
unsigned bytesWidth,
unsigned quality)

说明：得到的当前Buf数据保存为 JPG文件
参数：szFileName - [in] 存储JPG文件名
buffer - [in] 指向缓冲区
bufferLen - [in] 指向的缓冲区的大小（单位：字节）
colorSpace - [in] 获取的原始数据的色彩空间类型
width - [in] 获取的原始数据的像素宽度
height - [in] 获取的原始数据的像素高度
bytesWidth - [in] 获取的原始数据的字节宽度
quality - [in] JPG文件的画面质量
返回值：0 - 成功；失败则返回错误代码

4.8.4 unsigned __stdcall DXSnapToBMPFile(device_handle device, TCHAR* szFileName, PRECT rect = NULL);

说明：抓拍当前图片存储为bmp文件
参数：device - [in] 设备句柄
szFileName - [in] 存储的文件名
rect - [in] 指定图片抓拍区域
返回值：0 - 成功；失败则返回错误代码

4.8.5 unsigned __stdcall DXSnapToJPGFile(device_handle device, TCHAR* szFileName, unsigned

quality, PRECT rect = NULL);
说明：抓拍当前图片存储为jpg文件
参数：device - [in] 设备句柄
szFileName - [in] 存储的文件名
unsigned quality Jpg文件存储的质量，取值范围（0-100）值越大效果越好
rect - [in] 指定图片抓拍区域
返回值：0 - 成功；失败则返回错误代码

4.8.6 image_handle __stdcall DXCreateBufferImg(unsigned char* imgbuffer,
unsigned buffersize,
unsigned colorspace,
unsigned width,
unsigned height,
unsigned* err);

说明：打开需要预览的视频Buf数据
参数：imgbuffer - [in] 图片缓冲区
buffersize - [in] 图片缓冲区的大小（单位：字节）
colorspace - [in] 图片缓冲的数据的色彩空间
width - [in] 图片的宽度（单位：像素）
height - [in] 图片的高度（单位：像素）
err - [out] 返回错误代码
返回值：成功则返回图像句柄，否则返回 NULL。可通过 err 获得错误代码image_handle

4.8.7stdcall DXCreateFileImg(TCHAR* szImgFile, unsigned* err);

说明：打开需要预览的图片文件
参数：szImgFile - [in] 图片文件全路径
err - [out] 返回错误代码
返回值：成功则返回图像句柄，否则返回 NULL。可通过 err 获得错误代码// */

4.8.8 unsigned __stdcall DXUpdateImg(image_handle img,
HWND hPrevWnd,
RECT* rect);

说明：开始预览已打开的buf数据或图片文件
参数：img - [in] 已打开图片文件的句柄
返回值：0 - 成功；失败则返回错误代码

4.8.9 void __stdcall DXDestroyImg(image_handle img)
说明：停止dib数据或图片文件的预览
参数：img - [in] 已打开图像的句柄
返回值：无

4.9 音视频编码器设置

音视频编码器支持两种调用方法,一种是直接枚举系统中编码算法进行音视频的编码,另外一种是调用sdk中集成的编码算法直接进行编码。

4.9.1 unsigned __stdcall DXEnumVideoCodecs(PDEVICE_TAG devTags, unsigned& num);
说明： 枚举系统中视频编码过滤器
参数：devTags - [out] 用于返回获得的视频编码器的 TAG 的数组
num - [in/out] 指定 devTags 数组的元素个数，返回时则是实际上获得的编码器数量
返回值：0 - 成功；失败则返回错误代码

4.9.2 unsigned __stdcall DXEnumAudioCodecs (PDEVICE_TAG devTags, unsigned& num);
说明： 枚举系统中音频编码过滤器

参数：devTags - [out] 用于返回获得的音频编码器的 TAG 的数组
num - [in/out] 指定 devTags 数组的元素个数，返回时则是实际上获得的编码器数量
返回值：0 - 成功；失败则返回错误代码

4.9.3 unsigned __stdcall DXSetVideoCodec(device_handle device,
PDEVICE_TAG videoEncoder);

说明： 设置视频编码算法，注意：!!! 当正在进行录像时，则操作失败
参数：device - [in] 设备句柄
videoEncoder - [in] 视频编码器的 TAG，NULL - 表示不用编码，即录像时采用原始数据
返回值：0 - 成功；失败则返回错误代码

4.9.4 unsigned __stdcall DXSetAudioCodec(device_handle device,
PDEVICE_TAG audioEncoder);

说明：设置音频编码器 注意：!!! 当正在进行录像时，则操作失败
参数：device - [in] 设备句柄
audioEncoder - [in] 音频编码器的 TAG，NULL - 表示不用编码，即录像时采用原始数据
返回值： 0 - 成功；失败则返回错误代码

4.9.5 unsigned __stdcall DXGetVideoCodecPara(device_handle device,
unsigned& mode,
unsigned& bitrate,
unsigned& keyinterval);

说明：得到编码器参数
参数：device - [in]视频设备通道句柄
mode - [out] 编码模式： CBR/VBR
bitrate - [out] 码率（单位：kbps）
keyinterval - [out] 关键帧间隔
返回值：0 - 成功；失败则返回错误代码

4.9.6 unsigned __stdcall DXSetVideoCodecPara(device_handle device,
unsigned mode,
unsigned bitrate,
unsigned keyinterval);

说明： 设置视频编码器参数
参数：device - [in]视频设备通道句柄
mode - [in] 编码模式
bitrate - [in] 码率（单位：kbps）
keyinterval - [in] 关键帧间隔
返回值：0 - 成功；失败则返回错误代码

4.9.7 unsigned __stdcall DXVideoCodecIsSupport(unsigned codecType);

说明：检查系统中视频编码器是否支持，主要是判断系统中是否支持gpu编码
编码格式：codec_intel264
参数：codecType：编码格式
返回值：0 – 表示支持；失败则返回错误代码

4.10视频录像

4.10.1 unsigned __stdcall DXStartCapture(device_handle device,
char* szFileName,
BOOL saveAudio,
unsigned* timeSize = NULL,
unsigned* dataSize = NULL,

unsigned vidRatio=1);

说明：开始视频录像

参数： device - [in] 视频设备通道句柄

szFileName - [in] 指定录像文件文件名。NULL - 停止录像

saveAudio - [in] 是否将音频也录制到文件中

timeSize - [in] 限定录像文件的时间长度（单位：秒），NULL - 不限制

dataSize - [in] 限定录像文件的数据长度（单位：字节），NULL - 不限制

saveNotify - [in] 录像事件通知接口指针

vidRatio 视频帧在录像中保存的比例，1为全保存；2为保存1/2；3为保存1/3；4为保存1/4；其他值表示全部保存，此功能只有x264 Codec intel h264有效

返回值：0 - 成功；失败则返回错误代码

4.10.2 unsigned __stdcall DXStartCaptureEx (device_handle device,

char* szFileName,

BOOL saveAudio,

int fileFormat

RECT *rct = NULL,

unsigned* Reserved0 = NULL,

unsigned* Reserved1 = NULL,

unsigned vidRatio=1);

说明：区域录像，支持录像成文件，也支持网传

参数： device - [in] 视频设备通道句柄

szFileName - [in] 指定录像文件文件名。NULL - 停止录像

saveAudio - [in] 是否将音频也录制到文件中

int fileFormat - [in] 指定录像文件格式，FILE_AVI avi文件；FILE_MP4 MP4文件。

RECT *rct - [in] 指定录制视频区域大小，NULL - 不限制

vidRatio 视频帧在录像中保存的比例，1为全保存；2为保存1/2；3为保存1/3；4为保存1/4；其他值表示全部保存，此功能只有x264 Codec intel h264有效

返回值：0 - 成功；失败则返回错误代码

4.10.3 unsigned __stdcall DXStopCapture(device_handle device);

说明：停止视频录像

参数：device - [in] 视频设备通道句柄

返回值：0 - 成功；失败则返回错误代码

4.10.4 unsigned __stdcall DXFreezeCaputre(device_handle device, BOOL bFreeze);

说明：暂停音视频录像（包括视频和音频）注意：!!! 适用于需要多段时间录像的情况

参数：device - [in] 视频设备通道句柄

bFreeze - [in] TRUE – 暂停录像，FALSE – 开始录像

返回值：0 - 成功；失败则返回错误代码

4.10.5 unsigned __stdcall DXNextFrameEncIDR(device_handle device);

说明：视频强制编码一个关键帧

参数：device - [in] 视频设备通道句柄

返回值：0 - 成功；失败则返回错误代码

4.10.6 void __stdcall DXMp4fileRepairSupport(bool bRepairSupport);

说明：MP4文件录像时增加修复支持，必须在录像开始前设置

参数： bSupport - [in] ture - 支持修复；false - 不支持修复，将无法恢复录像异常的mp4文件

返回值：0 - 成功；失败则返回错误代码

4.10.7 int __stdcall DXMp4fileRepairProgress(void);

说明：获取修复MP4文件进度
参数：无
返回值：（0~100）修复进度，=100表示修复成功完成；-1 失败件进度

4.10.8 unsigned __stdcall DXCreateMp4file(device_handle device, char* szFileName,bool bHasAud);

说明：创建MP4文件
参数：device - [in] 视频设备通道句柄
szFileName - [in] 创建的文件名称
bHasAud - [in] 是否包含音频
返回值：0 - 成功；失败则返回错误代码

4.10.9 unsigned __stdcall DXWriteMp4file(dxmp4_handle mp4hnd,int DataType, BYTE * pIn, int nlen, bool isKey, LONGLONG TimeStamp);

说明：写视频数据到mp4文件中
参数：mp4hnd - [in] mp4文件句柄
DataType - [in] 数据流类型
pIn - [in] buf数据
nlen - [in] buf数据长度
isKey - [in] 是否为关键帧
TimeStamp [in]时间戳
返回值：0 - 成功；失败则返回错误代码

4.10.10 unsigned __stdcall DXCloseMp4file(device_handle device);

说明：关闭mp4文件
参数：device - [in] 视频设备通道句柄
返回值：0 - 成功；失败则返回错误代码

4.10.11 unsigned __stdcall DXMp4fileRepair(const char *szFileName, char* szSaveName);

说明：修复异常的MP4文件，只有启用了添加录像修复信息生成的文件才能生效
为避免耗时等待，该接口返回成功时通过接口DXMp4fileRepairProgress查询实际的修复进度
参数： szFileName - [in] 需要修复的mp4文件路径
szSaveName - [in] 保存修复后的mp4文件路径
返回值：0 - 成功；失败则返回错误代码

4.11回调函数原型和回调函数设置

4.11.1 typedef unsigned (__stdcall *fnRawVideoCallback)(unsigned char* buffer,
unsigned colorSpace,
unsigned width,
unsigned height,
unsigned bytesWidth,
void* context);

说明： 原始视频数据回调的函数原型
参数：buffer - [in] 缓冲区指针
colorSpace - [in] 获取的原始数据的色彩空间类型
width - [in] 获取的原始数据的像素宽度
height - [in] 获取的原始数据的像素高度
bytesWidth - [in] 获取的原始数据的字节宽度
context - [in] 回调函数的上下文
返回值：0 - 成功；失败则返回错误代码

4.11.2 unsigned __stdcall DXSetRawVideoCallback(device_handle device,

fnRawVideoCallback fn,
void* context);

说明： 设置原始视频数据回调的函数

参数：device - [in]视频设备通道句柄

fn - [in] 回调函数指针，参考回调函数原型 ，NULL - 停止回调

context - [in] 回调函数的上下文

返回值：0 - 成功；失败则返回错误代码

4.11.3 unsigned __stdcall DXStartRawVideoCallback(device_handle device,
fnRawVideoCallback fn,
void* context);

说明：开始原始视频数据回调

参数：device - [in]视频设备通道句柄

fn - [in] 回调函数指针，NULL - 停止回调

context - [in] 回调函数的上下文

返回值：0 - 成功；失败则返回错误代码

4.11.4 unsigned __stdcall DXStopRawVideoCallback(device_handle device);

说明：停止原始视频数据回调

参数：device - [in]视频设备通道句柄

返回值：0 - 成功；失败则返回错误代码

4.11.5 typedef unsigned (__stdcall *fnRawAudioCallback)(unsigned char* buffer,
unsigned bufferLen,
WAVEFORMATEX* wfx,
void* context);

说明： 原始音频数据回调的函数原型参数：

参数：buffer - [in] 缓冲区指针

bufferLen - [in] 指向的缓冲区的大小（单位：字节）

wfx - [in] 音频数据格式

context - [in] 回调函数的上下文

返回值：0 - 成功；失败则返回错误代码

4.11.6 unsigned __stdcall DXSetRawAudioCallback(device_handle device,
fnRawAudioCallback fn,
void* context);

说明：设置原始音频数据回调的函数

参数：device - [in] 设备句柄

fnRawAudioCallback fn,- [in] 回调函数指针，NULL - 停止回调

context - [in] 回调函数的上下文

返回值：0 - 成功；失败则返回错误代码

4.11.7 typedef unsigned (__stdcall *fnEncVideoCallback)(unsigned fourcc,
unsigned char* buffer,
unsigned bufferSize,
void* context);

说明： 编码视频数据回调的函数原型

参数：fourcc - [in] 获取的编码数据的编码器的FOURCC

buffer - [in] 缓冲区指针

bufferSize - [out] 获取的编码数据的缓冲区大小（单位：字节）

context - [in] 回调函数的上下文
返回值：0 - 成功；失败则返回错误代码
4.11.8 unsigned __stdcall DXSetEncVideoCallback(device_handle device,
fnEncVideoCallback fn,
void* context);
说明：设置编码视频数据回调的函数
参数：device - [in] 设备句柄
fn - [in] 回调函数指针，NULL - 停止回调
context - [in] 回调函数的上下文
返回值：0 - 成功；失败则返回错误代码
4.11.9 unsigned __stdcall DXStartEncVideoCallback(device_handle device,
fnEncVideoCallback fn,
void* context);
说明：开始编码视频数据回调的函数
参数：device - [in] 设备句柄
fn - [in] 回调函数指针，NULL - 停止回调
context - [in] 回调函数的上下文
返回值：0 - 成功；失败则返回错误代码
4.11.10 unsigned __stdcall DXStopEncVideoCallback(device_handle device);
说明：停止编码视频数据回调的函数
参数：device - [in] 设备句柄
返回值：0 - 成功；失败则返回错误代码

4.12网络传输

4.12.1unsigned __stdcall DXStartNetTrans(device_handle device, int nPrtclType, PNetTransPara stNetPara);

说明：开启网络传输
参数：device - [in] 设备句柄
nPrtclType - [in] 协议类型net_crtsp=0,net_crtmp=1,net_all=2
stNetPara - [in]具体参数，选择net_crtsp时必须填写port；选择net_crtmp时必须填crtmp_uri；
返回值：0 - 成功；失败则返回错误代码
4.12.2 unsigned __stdcall DXNetTransData(device_handle device, int nPrtclType, int DataType,unsigned char* data, unsigned len, bool bIskey, LONGLONG ms_pts);
说明： 编码回调中写入回调的音视频数据
参数：device - [in] 设备句柄
nPrtclType - [in] 协议类型net_crtsp=0,net_crtmp=1,net_all=2
DataType - [in] 数据流类型data_vid=0,data_aud=1
data - [in] 数据流缓冲区
len - [in] 数据长度
ms_pts - [in] 时间戳毫秒
返回值：0 - 成功；失败则返回错误代码
4.12.3 unsigned __stdcall DXNetTransMP4File(device_handle device, int nPrtclType, const char *szFileName, bool bAudio, LONGLONG timeBegin);
说明： 网络传输MP4录像文件
参数：device - [in] 设备句柄
nPrtclType - [in] 协议类型net_crtsp=0,net_crtmp=1,net_all=2
szFileName - [in] MP4录像文件名

bAudio - [in] 是否播放音频
timeBegin - [in] 开始时间（毫秒）
返回值：0 - 成功；失败则返回错误代码

4.12.4 unsigned __stdcall DXStopNetTrans(device_handle device, int nPrtclType);

说明： 关闭网络传输
参数： device - [in] 设备句柄
nPrtclType - [in] 协议类型net_crtsp=0,net_crtmp=1,net_all=2
返回值：0 - 成功；失败则返回错误代码