

大华播放 SDK 编程手册

VERSION 3.30.0(Build 100831)

2010-08-31

版权所有 侵权必究



前 言

非常感谢您使用我们公司的设备，我们将为您提供最好的服务。

本手册可能包含技术上不准确的地方或印刷错误，欢迎指正。我们将会定期更新手册的内容。

修订记录

日期	修订内容
2006-12-18	创建
2006-12-27	整理文档
2007-9-3	增加对最新接口的说明
2007-10-16	整理文档
2008-03-25	整理文档，修改多显卡及设置高画质接口的定义
2009-07-15	整理文档，增加对最新接口的说明
2009-8-14	整理文档，修改文档版式，添加接口说明
2009-10-26	添加接口说明： PLAY_GetFreePort, PLAY_ReleasePort; PLAY_VerticalSyncEnable; PLAY_GetPicBMP, PLAY_GetPicJPEG; 添加媒体信息获取指令：PLAY_CMD_GetMediaInfo
2010-1-22	添加接口： PLAY_SetVisibleDecCallBack; PLAY_CatchResizePic
2010-1-29	添加接口： PLAY_SetFileRefCallBackEx PLAY_GetRealFrameBitRate
2010-4-8	添加接口： PLAY_SetPandoraWaterMarkCallBack
2010-4-14	添加接口： PLAY_StartAVIConvert PLAY_StopAVIConvert PLAY_SetWaterMarkCallBackEx
2010-5-14	添加接口： PLAY_CutFileSegment
2010-7-5	添加接口： PLAY_SetVideoPerTimer PLAY_GetVideoPerTimer

目 录

1 简介	10
1.1 概述	10
1.2 适用性	10
2 设计原则	10
2.1 典型调用	10
2.1.1 函数调用顺序.....	10
2.1.2 回调及功能设置.....	11
2.1.3 各功能操作及信息获取.....	12
2.1.4 可随时调用的函数.....	14
2.2 编程补充说明	14
3 数据结构定义	14
3.1 宏定义	14
3.1.1 最大通道数.....	14
3.1.2 声音波形范围.....	14
3.1.3 定时器类型.....	14
3.1.4 缓冲类型.....	15
3.1.5 错误类型.....	15
3.1.6 最大区域显示数.....	16
3.1.7 显示类型.....	16
3.1.8 解码缓冲数.....	16
3.1.9 定位类型.....	16
3.1.10 数据流原始缓冲大小	16
3.1.11 数据流播放模式	16
3.1.12 解码回调音频帧类型	16
3.1.13 解码回调视频帧类型	17
3.1.14 媒体信息查询指令	17
3.1.15 系统功能	17
3.1.16 抓图格式类型定义	17
3.2 媒体信息结构	17
3.3 帧信息结构	18
3.3.1 帧位置.....	18
3.3.2 帧信息.....	18

3.3.3 帧类型.....	18
4 接口定义	19
4.1 通道控制	19
4.1.1 获取空闲通道号PLAY_GetFreePort	19
4.1.2 释放通道号PLAY_ReleasePort	19
4.2 播放文件	19
4.2.1 打开文件PLAY_OpenFile.....	19
4.2.2 打开文件并自动分配通道号PLAY_CreateFile.....	20
4.2.3 开启播放PLAY_Play	20
4.2.4 停止播放PLAY_Stop	20
4.2.5 关闭文件PLAY_CloseFile	20
4.2.6 关闭播放文件并释放自动分配的通道号PLAY_DestroyFile	21
4.3 播放流数据	21
4.3.1 打开流PLAY_OpenStream	21
4.3.2 打开流接口并自动分配通道号PLAY_CreateStream	21
4.3.3 输入流数据PLAY_InputData	22
4.3.4 关闭流PLAY_CloseStream	22
4.3.5 关闭数据流，并释放自动分配的通道号PLAY_DestroyStream	22
4.3.6 打开流（以音视频分开方式输入方式）PLAY_OpenStreamEx.....	23
4.3.7 输入视频流PLAY_InputVideoData	23
4.3.8 输入音频流PLAY_InputAudioData	23
4.3.9 关闭流（以音视频分开方式输入方式）PLAY_CloseStreamEx.....	24
4.3.10 流方式历史数据（包括本地文件和远程录像文件）播放简单示例：	24
4.3.11 实时流数据播放简单示例：	24
4.4 回放控制	25
4.4.1 播放暂停/恢复PLAY_Pause.....	25
4.4.2 快速播放PLAY_Fast.....	25
4.4.3 慢速播放PLAY_Slow.....	25
4.4.4 单帧播放PLAY_OneByOne	25
4.4.5 单帧回退PLAY_OneByOneBack.....	26
4.4.6 单帧回退（与PLAY_OneByOneBack重复）PLAY_BackOne	26
4.4.7 反向回放PLAY_Back *	26
4.5 音频控制	26
4.5.1 以独占方式打开声音PLAY_PlaySound.....	26

4.5.2 关闭声音（独占方式）PLAY_StopSound	27
4.5.3 以共享方式播放声音PLAY_PlaySoundShare.....	27
4.5.4 关闭声音(共享方式) PLAY_StopSoundShare.....	27
4.5.5 设置音量PLAY_SetVolume	27
4.5.6 获取音量PLAY_GetVolume.....	28
4.5.7 调整WAVE波形PLAY_AdjustWaveAudio *	28
4.6 数据回调	28
4.6.1 设置解码回调PLAY_SetDecCallBack.....	28
4.6.2 设置解码回调（增加用户传递参数）PLAY_SetDecCallBackEx	29
4.6.3 解码回调（同时可显示视频）PLAY_SetVisibleDecCallBack	30
4.6.4 设置解码回调流类型PLAY_SetDecCBStream	30
4.6.5 设置抓图回调PLAY_SetDisplayCallBack	31
4.6.6 音频解码后的WAVE数据回调PLAY_SetAudioCallBack	31
4.6.7 设置数据校验PLAY_SetVerifyCallBack *	32
4.6.8 源数据分析完的数据回调PLAY_SetDemuxCallBack.....	33
4.6.9 设置水印数据回调PLAY_SetWaterMarkCallBack	33
4.6.10 设置水印数据回调PLAY_SetWaterMarkCallBackEx	34
4.6.11 设置编码水印数据回调PLAY_SetPandoraWaterMarkCallBack	35
4.6.12 切割文件PLAY_CutFileSegment	35
4.7 消息回调	36
4.7.1 分辨率改变通知消息PLAY_SetEncChangeMsg.....	36
4.7.2 设置文件结束时要发送的消息PLAY_SetFileEndMsg	36
4.8 函数回调	37
4.8.1 设置源缓冲区阈值及回调指针PLAY_SetSourceBufCallBack	37
4.8.2 重置回调标志为有效状态PLAY_ResetSourceBufFlag	37
4.8.3 图像分辨率改变通知回调PLAY_SetEncTypeChangeCallBack.....	38
4.8.4 设置建立索引回调PLAY_SetFileRefCallBack.....	38
4.8.5 设置建立索引回调(返回索引创建情况)PLAY_SetFileRefCallBackEx	39
4.8.6 设置文件结束回调PLAY_SetFileEndCallBack.....	39
4.8.7 开始AVI转换并设置AVI转换状态回调 PLAY_StartAVIConvert.....	40
4.8.8 停止AVI转换PLAY_StopAVIConvert.....	40
4.9 文件索引	41
4.9.1 设置文件索引PLAY_SetRefValue.....	41
4.9.2 获取文件索引PLAY_GetRefValue.....	41
4.10 文件定位	41

4.10.1 设置文件当前帧播放帧号PLAY_SetCurrentFrameNum.....	41
4.10.2 设置文件当前播放时间（毫秒）PLAY_SetPlayedTimeEx	42
4.10.3 设置文件播放指针的相对位置（百分比）PLAY_SetPlayPos.....	42
4.10.4 获取文件播放指针的相对位置（百分比）PLAY_GetPlayPos	42
4.11 设置属性	42
4.11.1 设置视频参数PLAY_SetColor.....	42
4.11.2 设置播放缓冲区最大缓冲帧数PLAY_SetDisplayBuf	43
4.11.3 设置显示模式PLAY_SetDisplayType	43
4.11.4 垂直同步显示开关PLAY_VerticalSyncEnable.....	44
4.11.5 调整图像流畅性PLAY_AdjustFluency *	44
4.11.6 改变图像播放的帧率PLAY_ChangeRate.....	44
4.11.7 打开音频采集功能PLAY_OpenAudioRecord.....	45
4.11.8 关闭音频采集功能PLAY_CloseAudioRecord.....	45
4.11.9 设置Overlay模式和关键色PLAY_SetOverlayMode	45
4.11.10 设置图像质量PLAY_SetPicQuality	46
4.11.11 设置流播放模式PLAY_SetStreamOpenMode	47
4.11.12 设置播放使用的定时器类型PLAY_SetTimerType.....	47
4.11.13 设置每个定时器管理的视频个数PLAY_SetVideoPerTimer	47
4.12 获得属性	48
4.12.1 测试播放所需系统功能PLAY_GetCaps.....	48
4.12.2 获取视频参数PLAY_GetColor.....	48
4.12.3 获取播放缓冲区最大缓冲帧数PLAY_GetDisplayBuf	49
4.12.4 获取显示模式PLAY_GetDisplayType	49
4.12.5 获取OVERLAY关键色PLAY_GetColorKey	49
4.12.6 检查当前是否使用了OVERLAY显示模式PLAY_GetOverlayMode	49
4.12.7 获取图像质量PLAY_GetPictureQuality	49
4.12.8 获取流播放模式PLAY_GetStreamOpenMode	50
4.12.9 获取播放使用的定时器类型PLAY_GetTimerType	50
4.12.10 获取指定缓冲区的大小PLAY_GetBufferValue	50
4.12.11 获取当前播放的帧序号PLAY_GetCurrentFrameNum	50
4.12.12 获取当前帧率PLAY_GetCurrentFrameRate.....	51
4.12.13 获取文件头长度PLAY_GetFileHeadLength	51
4.12.14 获取文件总时间PLAY_GetFileTime.....	51
4.12.15 获取文件总帧数PLAY_GetFileTotalFrames.....	51

4.12.16 查找指定位置之前的关键帧位置PLAY_GetKeyFramePos	52
4.12.17 查找指定位置之后的关键帧位置PLAY_GetNextKeyFramePos	52
4.12.18 获取原始图像大小PLAY_GetPictureSize	53
4.12.19 获取已解码的视频帧数PLAY_GetPlayedFrames	53
4.12.20 获取当前文件播放时间PLAY_GetPlayedTime	53
4.12.21 获取文件当前播放时间（毫秒）PLAY_GetPlayedTimeEx	54
4.12.22 信息状态查询PLAY_QueryInfo	54
4.12.23 获取流播放模式下源缓冲区剩余数据大小PLAY_GetSourceBufferRemain	55
4.12.24 获取视频实时码率PLAY_GetRealFrameBitRate	55
4.12.25 获取视频实时码率PLAY_GetVideoPerTimer	55
4.13 多屏显示控制	56
4.13.1 枚举系统中的显示设备PLAY_InitDDrawDevice *	56
4.13.2 释放枚举显示设备的过程中分配的资源PLAY_ReleaseDDrawDevice *	56
4.13.3 设置播放窗口使用的显示设备PLAY_SetDDrawDevice *	56
4.13.4 设置播放窗口使用的显示设备（多显示区域）PLAY_SetDDrawDeviceEx *	56
4.13.5 获取指定显卡和监视器信息PLAY_GetDDrawDeviceInfo *	57
4.13.6 获取显示设备（显卡）个数PLAY_GetDDrawDeviceTotalNums *	57
4.13.7 获取指定显示设备的系统信息PLAY_GetCapsEx *	58
4.14 抓图	58
4.14.1 图像格式转为BMP格式PLAY_ConvertToBmpFile	58
4.14.2 图像格式转为JPEG格式PLAY_ConvertToJpegFile	58
4.14.3 直接抓取图像（BMP）PLAY_CatchPic	59
4.14.4 直接抓取图像（可选择格式）PLAY_CatchPicEx	59
4.14.5 抓图（可选择格式并指定宽高）PLAY_CatchResizePic	59
4.14.6 直接抓取BMP图像PLAY_GetPicBMP	60
4.14.7 直接抓取JPEG图像PLAY_GetPicJPEG	60
4.15 字符叠加	61
4.15.1 画图回调PLAY_RigisterDrawFun	61
4.15.2 画图回调(多显示区域) PLAY_RigisterDrawFunEx	61
4.16 多区域显示	62
4.16.1 设置或增加显示区域PLAY_SetDisplayRegion	62
4.16.2 刷新显示（多显示区域）PLAY_RefreshPlayEx	63
4.17 数据流录像	63
4.17.1 开始流数据录像PLAY_StartDataRecord	63

4.17.2 停止流数据录像PLAY_StopDataRecord	63
4.17.3 开始AVI数据录像(可指定视频宽高)PLAY_StartAVIResizeConvert	64
4.17.4 停止AVI数据录像PLAY_StopAVIResizeConvert.....	64
4.18 清缓冲	64
4.18.1 清空流播放模式下源缓冲区的剩余数据PLAY_ResetSourceBuffer.....	64
4.18.2 清空指定缓冲区的剩余数据PLAY_ResetBuffer	64
4.19 智能搜索	65
4.19.1 设置搜索区域及范围PLAY_SetMDRange	65
4.19.2 设置智能搜索的阈值PLAY_SetMDThreShold	65
4.19.3 获取搜索到的数据帧的帧序号PLAY_GetMDPosition	66
4.20 获得版本号	66
4.20.1 获取播放库SDK版本号、次版本号和补丁号PLAY_GetSdkVersion	66
4.21 获得错误号	66
4.21.1 获取错误号PLAY_GetLastError	66
4.22 其它	67
4.22.1 初始化DirectDraw表面PLAY_InitDDraw *.....	67
4.22.2 释放DirectDraw表面PLAY_RealeseDDraw *	67
4.22.3 设置丢B帧个数PLAY_ThrowBFrameNum *	67
4.22.4 刷新显示PLAY_RefreshPlay.....	67

1 简介

1.1 概述

播放 SDK 是大华压缩卡和硬盘录像机的配套产品，支持大华所有码流格式以及海思公司的 h264 码流和 ADI 的 h264 码流。本文档详细描述了开发包中各函数实现的功能及接口

播放 SDK 的主要功能有：支持文件或流数据的播放、回放控制（如暂停\恢复、快放慢放）、音频控制、流数据录像、多区域显示、按帧序号或按时间定位、数据回调、消息回调、字符叠加、抓图.....

开发包中包括的文件有：**dhplay.dll**、**dhplay.h**、**dhplay.lib**、底层 mpeg4 解码库（**dllmpeg4.dll**）、底层 h264 解码库 **dllh264.dll**（解大华的 h264 码流和 ADI 公司的码流）和海思的 h264 解码库，包括（**AmrDLL.dll**、**DLLDeinterlace.dll**、**hi_h264dec_w.dll**）。

1.2 适用性

- ❖ 支持大华所有码流以及ADI的h264码流和海思的h264码流的解码
- ❖ 支持HB、HBE、LB、LBE、GB、GBE、NVS机型码流设计原则

2 设计原则

2.1 典型调用

2.1.1 函数调用顺序

应用程序初始化

文件模式下：

[PLAY_GetFreePort](#)

[回调及功能设置函数](#)

[PLAY_OpenFile](#)

[PLAY_Play](#)

[各功能操作及信息获取函数](#)

[PLAY_Stop](#)

[PLAY_CloseFile](#)

[PLAY_ReleasePort](#)

流模式下：

[PLAY_GetFreePort](#)

[回调及功能设置函数](#)

[PLAY_SetStreamOpenMode](#)

[PLAY_OpenStream](#) / [PLAY_OpenStreamEx](#)

[PLAY_Play](#)

[PLAY_InputData](#) 以及[各功能操作及信息获取函数](#)

[PLAY_Stop](#)

[PLAY_CloseStream](#) / [PLAY_CloseStreamEx](#)

[PLAY_ReleasePort](#)

应用程序结束

2.1.2 回调及功能设置

设置文件结束消息回调	PLAY_SetFileEndMsg
设置文件结束函数回调	PLAY_SetFileEndCallBack
设置图像格式改变消息回调	PLAY_SetEncChangeMsg
设置图像格式改变函数回调	PLAY_SetEncTypeChangeCallBack
设置文件索引建立后回调	PLAY_SetFileRefCallBack PLAY_SetFileRefCallBackEx
设置流缓冲小于阈值回调	PLAY_SetSourceBufCallBack
设置数据解码回调	PLAY_SetDecCallBack PLAY_SetDecCallBackEx
设置数据解码回调同时显示视频	PLAY_SetVisibleDecCallBack
设置视频解码回调	PLAY_SetDisplayCallBack
设置音频解码回调	PLAY_SetAudioCallBack
设置水印数据回调	PLAY_SetVerifyCallBack *
设置分析数据回调	PLAY_SetDemuxCallBack
设置文件索引	PLAY_SetRefValue

设置播放缓冲帧数	<u>PLAY_SetDisplayBuf</u>
设置 overlay 模式及关键色	<u>PLAY_SetOverlayMode</u>
设置流打开模式	<u>PLAY_SetStreamOpenMode</u>
设置定时器类型	<u>PLAY_SetTimerType</u>
设置水印数据回调	<u>PLAY_SetWaterMarkCallBack</u> <u>PLAY_SetWaterMarkCallBackEx</u>

2.1.3 各功能操作及信息获取

播放声音	<u>PLAY_PlaySound</u> <u>PLAY_PlaySoundShare</u>
设置音量	<u>PLAY_SetVolume</u>
获取音量	<u>PLAY_GetVolume</u>
关闭声音	<u>PLAY_StopSound</u> <u>PLAY_StopSoundShare</u>
开始流数据录像	<u>PLAY_StartDataRecord</u>
输入流数据	<u>PLAY_InputData</u> <u>PLAY_InputVideoData</u> <u>PLAY_InputAudioData</u>
结束流数据录像	<u>PLAY_StopDataRecord</u>
重置流缓冲回调标志	<u>PLAY_ResetSourceBufFlag</u>
快放	<u>PLAY_Fast</u>
慢放	<u>PLAY_Slow</u>
暂停	<u>PLAY_Pause</u>
按帧号定位	<u>PLAY_SetCurrentFrameNum</u>
按时间定位	<u>PLAY_SetPlayedTimeEx</u>
按文件偏移定位	<u>PLAY_SetPlayPos</u>
刷新	<u>PLAY_RefreshPlay</u> <u>PLAY_RefreshPlayEx</u>
单帧播放	<u>PLAY_OneByOne</u>
单帧倒放	<u>PLAY_OneByOneBack</u> <u>PLAY_BackOne</u>
反向回放	<u>PLAY_Back *</u>
抓图(写文件)	<u>PLAY_CatchPic</u>

	PLAY_CatchPicEx
	PLAY_CatchResizePic
抓图(获取图片数据)	PLAY_GetPicBMP
	PLAY_GetPicJPEG
清缓冲	PLAY_ResetBuffer
颜色调整	PLAY_SetColor
多区域显示	PLAY_SetDisplayRegion
显示模式设置	PLAY_SetDisplayType
垂直同步使能	PLAY_VerticalSyncEnable
改变图象播放的帧率	PLAY_ChangeRate
打开音频采集功能	PLAY_OpenAudioRecord
关闭音频采集功能	PLAY_CloseAudioRecord
获取颜色参数	PLAY_GetColor
获取关键色	PLAY_GetColorKey
获取当前帧数	PLAY_GetCurrentFrameNum
获取当前帧率	PLAY_GetCurrentFrameRate
获取缓冲帧数	PLAY_GetDisplayBuf
获取显示类型	PLAY_GetDisplayType
获取当前时间	PLAY_GetPlayedTime
	PLAY_GetPlayedTimeEx
获取当前信息状态	PLAY_QueryInfo
获取总帧数	PLAY_GetFileTotalFrames
获取总时间	PLAY_GetFileTime
获取文件偏移	PLAY_GetPlayPos
获取索引信息	PLAY_GetRefValue
获取流剩余缓冲	PLAY_GetSourceBufferRemain
获取流打开模式	PLAY_GetStreamOpenMode
获取关键帧信息	PLAY_GetKeyFramePos
	PLAY_GetNextKeyFramePos
获取 OVERLAY 模式	PLAY_GetOverlayMode
获取图像大小	PLAY_GetPictureSize
获取已播放的帧数	PLAY_GetPlayedFrames
获取视频实时码率	PLAY_GetRealFrameBitRate

2.1.4 可随时调用的函数

获取系统功能	PLAY_GetCaps
转 BMP 文件	PLAY_ConvertToBmpFile
获取文件头长度	PLAY_GetFileHeadLength
获取版本信息	PLAY_GetSdkVersion
获取错误号	PLAY_GetLastError

2.2 编程补充说明

- 回调及功能设置函数只能在 **PLAY_play** 之前设置一次。如果用户不调用这些函数，播放库会采用默认值，如播放缓冲帧数默认为 15 帧，默认用 **overlay** 显示，默认流打开模式为最实时模式。

各功能操作及信息获取函数，在 **PLAY_Play** 和 **PLAY_Stop** 之间调用。但其中播放声音的函数 **PLAY_PlaySound**、**PLAY_PlaySoundShare** 可以在 **PLAY_Play** 之前调用，以防止部分声音数据不能播放，这在播放一个纯音频文件时较为明显。。

3 数据结构定义

3.1 宏定义

3.1.1 最大通道数

#define FUNC_MAX_PORT	501	//最大播放通道数
-----------------------	-----	-----------

3.1.2 声音波形范围

#define MIN_WAVE_COEF	-100
#define MAX_WAVE_COEF	100

3.1.3 定时器类型

#define TIMER_1	1	//默认定时器，精确定时
#define TIMER_2	2	//不精确定时，个数据不限制

3.1.4 缓冲类型

#define BUF_VIDEO_SRC	1	//视频源缓冲
#define BUF_AUDIO_SRC	2	//音频源缓冲
#define BUF_VIDEO_RENDER	3	//解码后视频数据缓冲
#define BUF_AUDIO_RENDER	4	//解码后音频数据缓冲

注:

BUF_VIDEO_SRC	视频数据源缓冲, 缓冲解码之前视频数据, 只对流模式有效, 单位 byte
BUF_AUDIO_SRC	音频数据源缓冲, 缓冲解码之前音频数据, 只对流模式有效, 单位 byte
BUF_VIDEO_RENDER	解码后视频数据缓冲, 单位帧数
BUF_AUDIO_RENDER	解码后音频数据缓冲, 单位帧数, 音频 40ms 数据定为一帧

3.1.5 错误类型

#define DH_PLAY_NOERROR	0	//没有错误
#define DH_PLAY_PARA_OVER	1	//输入参数非法
#define DH_PLAY_ORDER_ERROR	2	//调用顺序不对
#define DH_PLAY_TIMER_ERROR	3	//多媒体时钟设置失败
#define DH_PLAY_DEC_VIDEO_ERROR	4	//视频解码失败
#define DH_PLAY_DEC_AUDIO_ERROR	5	//音频解码失败
#define DH_PLAY_ALLOC_MEMORY_ERROR	6	//分配内存失败
#define DH_PLAY_OPEN_FILE_ERROR	7	//文件操作失败
#define DH_PLAY_CREATE_OBJ_ERROR	8	//创建线程事件等失败
#define DH_PLAY_CREATE_DDRAW_ERROR	9	//创建 directDraw 失败
#define DH_PLAY_CREATE_OFFSCREEN_ERROR	10	//创建后端缓存失败
#define DH_PLAY_BUF_OVER	11	//缓冲区满, 输入流失败
#define DH_PLAY_CREATE_SOUND_ERROR	12	//创建音频设备失败
#define DH_PLAY_SET_VOLUME_ERROR	13	//设置音量失败
#define DH_PLAY_SUPPORT_FILE_ONLY	14	//只能在播放文件时才能使用
#define DH_PLAY_SUPPORT_STREAM_ONLY	15	//只能在播放流时才能使用
#define DH_PLAY_SYS_NOT_SUPPORT	16	//系统不支持, 解码器只能工作在 Pentium 3 以上
#define DH_PLAY_FILEHEADER_UNKNOWN	17	//没有文件头
#define DH_PLAY_VERSION_INCORRECT	18	//解码器和编码器版本不对应
#define DH_PLAY_INIT_DECODER_ERROR	19	//初始化解码器失败
#define DH_PLAY_CHECK_FILE_ERROR	20	//文件太短或码流无法识别
#define DH_PLAY_INIT_TIMER_ERROR	21	//初始化多媒体时钟失败

#define DH_PLAY_BLT_ERROR	22	//位拷贝失败
#define DH_PLAY_UPDATE_ERROR	23	//显示 overlay 失败
#define DH_PLAY_MEMORY_TOOSMALL	24	//memory too small

3.1.6 最大区域显示数

#define MAX_DISPLAY_WND	4	//同时最多打开 4 个区域显示窗口
-------------------------	---	--------------------

3.1.7 显示类型

#define DISPLAY_NORMAL	1	//以正常分辨率显示
#define DISPLAY_QUARTER	2	//以四分之一分辨率显示

3.1.8 解码缓冲数

#define MAX_DIS_FRAMES	50	//最大解码缓冲帧数
#define MIN_DIS_FRAMES	6	//最小解码缓冲帧数

3.1.9 定位类型

#define BY_FRAMENUM	1	//按帧号
#define BY_FRAME_TIME	2	//按时间

3.1.10 数据流原始缓冲大小

#define SOURCE_BUF_MAX	1024*100000	//最大原始缓冲
#define SOURCE_BUF_MIN	1024*50	//最小原始缓冲

3.1.11 数据流播放模式

#define STREAME_REALTIME	0	//最实时方式
#define STREAME_FILE	1	//最流畅方式

3.1.12 解码回调音频帧类型

#define T_AUDIO16	101	
#define T_AUDIO8	100	

3.1.13 解码回调视频帧类型

#define T_UYVY	1
#define T_YV12	3
#define T_RGB32	7

3.1.14 媒体信息查询指令

#define PLAY_CMD_GetTime	1
#define PLAY_CMD_GetFileRate	2
#define PLAY_CMD_GetMediaInfo	3

3.1.15 系统功能

#define SUPPORT_DDRAW	1	//支持 DIRECTDRAW; 如果不支持, 则播放器不能工作
#define SUPPORT_BLT	2	//显卡支持 BLT 操作; 如果不支持, 则播放器不能工作
#define SUPPORT_BLTFOURCC	4	//显卡 BLT 支持颜色转换
#define SUPPORT_BLTSHRINKX	8	//显卡 BLT 支持 X 轴缩小
#define SUPPORT_BLTSHRINKY	16	//显卡 BLT 支持 Y 轴缩小
#define SUPPORT_BLTSTRETCHX	32	//显卡 BLT 支持 X 轴放大
#define SUPPORT_BLTSTRETCHY	64	//显卡 BLT 支持 Y 轴放大
#define SUPPORT_SSE	128	//CPU 支持 SSE 指令, Intel Pentium3 以上支持 SSE 指令
#define SUPPORT_MMX	256	//CPU 支持 MMX 指令集

3.1.16 抓图格式类型定义

typedef enum __tPicFormats
{
PicFormat_BMP = 0,
PicFormat_JPEG,
} tPicFormats;

3.2 媒体信息结构

typedef struct {

```

    long lWidth;
    long lHeight;
    long lFrameRate;
    long lChannel;
    long lBitPerSample;
    long lSamplesPerSec;
}MEDIA_INFO;
    
```

3.3 帧信息结构

3.3.1 帧位置

```

typedef struct{
    long nFilePos;           //指定帧在文件中的偏移位置
    long nFrameLen;          //帧长度
    long nFrameNum;          //帧序号
    long nFrameTime;         //帧时间
    long nErrorFrameNum;     //错误帧号
    SYSTEMTIME *pErrorTime;  //错误帧时间
    long nErrorLostFrameNum; //错误帧帧号
    long nErrorFrameSize;    //错误帧大小
}FRAME_POS,*PFRAME_POS;
    
```

3.3.2 帧信息

```

typedef struct{
    long nWidth;           //画面宽，单位像素。如果是音频数据则为 0
    long nHeight;          //画面高。如果是音频数据则为 0
    long nStamp;           //时标信息，单位毫秒
    long nType;            //视频帧类型，T_AUDIO16, T_RGB32, T_YV12
    long nFrameRate;       //编码时产生的图像帧率
}FRAME_INFO;
    
```

3.3.3 帧类型

```

typedef struct {
    char *pDataBuf;        //帧数据
    
```

```

    long nSize;           //帧大小
    long nFrameNum;       //帧序号
    BOOL bIsAudio;        //是否音频帧
    long nReserved;       //保留字
}FRAME_TYPE;
    
```

4 接口定义

4.1 通道控制

4.1.1 获取空闲通道号**PLAY_GetFreePort**

函数名称	BOOL PLAY_GetFreePort(LONG *pPort)	
参数说明	[out] LONG *pPort	获取的通道号
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	获取空闲通道号，上限为 501。与 PLAY_RealsePort 成对使用，	

[返回目录](#)

4.1.2 释放通道号**PLAY_ReleasePort**

函数名称	BOOL PLAY_ReleasePort (LONG lPort)	
参数说明	[in] LONG lPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	与 PLAY_GetFreePort 成对使用	

[返回目录](#)

4.2 播放文件

4.2.1 打开文件**PLAY_OpenFile**

函数名称	BOOL PLAY_OpenFile(LONG nPort, LPSTR sFileName)	
参数说明	[in] LONG nPort	通道号
	[in] LPSTR sFileName	文件名
返回值	成功返回 TRUE，不成功返回 FALSE	
注意	文件不能超过 4G 或小于 4K	

[返回目录](#)

4.2.2 打开文件并自动分配通道号**PLAY_CreateFile**

函数名称	BOOL PLAY_CreateFile(LONG nPort, LPSTR sFileName)	
参数说明	[in] LONG nPort	未使用
	[in] LPSTR sFileName	文件名
返回值	成功返回 TRUE，不成功返回 FALSE	
注意	文件不能超过 4G 或小于 4K	

[返回目录](#)

4.2.3 开启播放**PLAY_Play**

函数名称	BOOL PLAY_Play(LONG nPort, HWND hWnd)	
参数说明	[in] LONG nPort	通道号
	[in] HWND hWnd	播放窗口句柄
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	开始播放。如果已经播放，改变当前播放状态为正常速度播放	

[返回目录](#)

4.2.4 停止播放**PLAY_Stop**

函数名称	BOOL PLAY_Stop(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.2.5 关闭文件**PLAY_CloseFile**

函数名称	BOOL PLAY_CloseFile (LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	在 PLAY_Stop 后调用	

[返回目录](#)

4.2.6 关闭播放文件并释放自动分配的通道号**PLAY_DestroyFile**

函数名称	BOOL PLAY_DestroyFile (LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.3 播放流数据

4.3.1 打开流**PLAY_OpenStream**

函数名称	BOOL PLAY_OpenStream(LONG nPort, PBYTE pFileHeadBuf, DWORD nSize, DWORD nBufPoolSize)	
参数说明	[in] LONG nPort	通道号
	[in] PBYTE pFileHeadBuf	目前不使用，填NULL
	[in] DWORD nSize	目前不使用，填0
	[in] DWORD nBufPoolSize	设置播放器中存放数据流的缓冲区大小。范围是 [SOURCE_BUF_MIN, SOURCE_BUF_MAX]。一般设为900*1024，如果数据送过来相对均匀，可调小该值，如果数据传输不均匀，可增大该值。
返回值	成功返回 TRUE，不成功返回 FALSE	
补充说明	pFileHeadBuf 原先用于识别该码流是否是大华码流，后改由码流中的标志来识别，所以现在该参数实际不起作用。这样做的目的是为方便用户做二次开发，码流识别由播放库内部处理，而不需要用户传个特定厂家的文件头	

[返回目录](#)

4.3.2 打开流接口并自动分配通道号**PLAY_CreateStream**

函数名称	BOOL PLAY_CreatStream(LONG nPort, PBYTE pFileHeadBuf, DWORD nSize, DWORD nBufPoolSize)	
参数说明	[in] LONG nPort	通道号，未使用
	[in] PBYTE pFileHeadBuf	目前不使用，填NULL
	[in] DWORD nSize	目前不使用，填0
	[in] DWORD nBufPoolSize	置播放器中存放数据流的缓冲区大小。范围是 [SOURCE_BUF_MIN,

		SOURCE_BUF_MAX]。一般设为900*1024，如果数据送过来相对均匀，可调小该值，如果数据传输不均匀，可增大该值
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.3.3 输入流数据**PLAY_InputData**

函数名称	BOOL PLAY_InputData(LONG nPort, PBYTE pBuf, DWORD nSize)	
参数说明	[in] LONG nPort	通道号
	[in] PBYTE pBuf	缓冲区地址
	[in] DWORD nSize	缓冲区大小
返回值	TURE,表示已经输入数据。FALSE 表示失败，数据没有输入，一般为缓冲已满	
使用描述	打开流并调用 PLAY_Play 之后才能输入流数据。当调用顺序准确之后，返回 FALSE 时，即内部缓冲区已满，用户可暂停输入，一段时间之后再输入流，确保播放库不丢失数据。	

[返回目录](#)

4.3.4 关闭流**PLAY_CloseStream**

函数名称	BOOL PLAY_CloseStream(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.3.5 关闭数据流，并释放自动分配的通道号**PLAY_DestroyStream**

函数名称	BOOL PLAY_DestroyStream(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.3.6 打开流（以音视频分开方式输入方式） **PLAY_OpenStreamEx**

函数名称	BOOL PLAY_OpenStreamEx(LONG nPort, PBYTE pFileHeadBuf, DWORD nSize, DWORD nBufPoolSize)	
参数说明	[in] LONG nPort	通道号
	[in] PBYTE pFileHeadBuf	目前不使用，填NULL
	[in] DWORD nSize	目前不使用，填0
	[in] DWORD nBufPoolSize	设置播放器中存放数据流的缓冲区大小。范围是 [SOURCE_BUF_MIN, SOURCE_BUF_MAX]
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.3.7 输入视频流 **PLAY_InputVideoData**

函数名称	BOOL PLAY_InputVideoData(LONG nPort, PBYTE pBuf, DWORD nSize)	
参数说明	[in] LONG nPort	通道号
	[in] PBYTE pBuf	缓冲区地址
	[in] DWORD nSize	缓冲区大小
返回值	TURE，表示已经输入数据；FALSE 表示失败，数据没有输入	
使用描述	输入视频流（可以是复合流，但音频数据会被忽略）；打开流之后才能输入数据。当调用顺序准确之后，返回 FALSE 时，即内部缓冲区已满，用户可暂停输入，一段时间之后再输入流，确保播放库不丢失数据。	

[返回目录](#)

4.3.8 输入音频流 **PLAY_InputAudioData**

函数名称	BOOL PLAY_InputAudioData (LONG nPort, PBYTE pBuf, DWORD nSize)	
参数说明	[in] LONG nPort	通道号
	[in] PBYTE pBuf	缓冲区地址
	[in] DWORD nSize	缓冲区大小
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	输入音频流；打开声音之后才能输入数据。当调用顺序准确之后，返回 FALSE 时，即内部缓冲区已满，用户可暂停输入，一段时间之后再输入流，确保播放库不丢失数据。	

[返回目录](#)

4.3.9 关闭流（以音视频分开方式输入方式） **PLAY_CloseStreamEx**

函数名称	BOOL PLAY_CloseStreamEx (LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.3.10 流方式历史数据（包括本地文件和远程录像文件）播放简单示例：

```
PLAY_OpenStream(0, NULL, 0, 900*1024);
PLAY_SetStreamOpenMode(0, STREAME_FILE)//文件模式
PLAY_Play(0, hWnd);
FILE* fp = fopen("file.dav","rb");
BYTE pBuf[4096];
while (true)
{
    int len = fread(pBuf,1,4096,fp);
    if (len <= 0)
    {
        break;
    }
    While (PLAY_InputData(0, pBuf, len) == FALSE)
    {
        Sleep(45);
    }
}
```

4.3.11 实时流数据播放简单示例：

```
PLAY_OpenStream(0, NULL, 0, 900*1024);
PLAY_SetStreamOpenMode(0, STREAME_REALTIME)//实时模式,
                        //默认是实时模式
PLAY_Play(0, hWnd);
//网络流数据回调函数
void WINAPI CallFunction(LONG nPort, LPBYTE pDataBuffer, DWORD DataLength, long nUser)
{
    PLAY_InputData(nPort, pDataBuffer, DataLength);
    //网络实时流数据，数据只送一次,以保证实时性
}
```


4.4 回放控制

4.4.1 播放暂停/恢复 **PLAY_Pause**

函数名称	BOOL PLAY_Pause(LONG nPort, DWORD nPause)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nPause	TRUE 暂停, FALSE 恢复
返回值	成功返回 TRUE, 不成功返回 FALSE	

[返回目录](#)

4.4.2 快速播放 **PLAY_Fast**

函数名称	BOOL PLAY_Fast(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE, 不成功返回 FALSE	
使用描述	播放速度分为九级, 播放速度分别为每秒播放 1,3,6,12,25,50,75,100,125 帧图像。每次调用播放速度提升一级, 最多调用 4 次, 要恢复正常播放调用 PLAY_Play, 从当前位置开始正常播放	

[返回目录](#)

4.4.3 慢速播放 **PLAY_Slow**

函数名称	BOOL PLAY_Slow (LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE, 不成功返回 FALSE	
使用描述	关于播放速度的描述见 PLAY_Fast。每次调用播放速度降一级; 最多调用 4 次, 要恢复正常播放调用 PLAY_Play	

[返回目录](#)

4.4.4 单帧播放 **PLAY_OneByOne**

函数名称	BOOL PLAY_OneByOne(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE, 不成功返回 FALSE	
使用描述	要恢复正常播放调用 PLAY_Play	

[返回目录](#)

4.4.5 单帧回退PLAY_OneByOneBack

函数名称	BOOL PLAY_OneByOneBack(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	每调用一次倒退一帧	
注意	此接口只支持文件播放，必须在文件索引生成之后才能调用	

[返回目录](#)

4.4.6 单帧回退（与PLAY_OneByOneBack重复）PLAY_BackOne

函数名称	BOOL PLAY_BackOne(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.4.7 反向回放PLAY_Back *

函数名称	BOOL PLAY_Back(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	
注意	此接口无效	

[返回目录](#)

4.5 音频控制

4.5.1 以独占方式打开声音PLAY_PlaySound

函数名称	BOOL PLAY_PlaySound (LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	打开声音；同一时刻只能有一路声音。如果现在已经有声音打开，则自动关闭原来已经打开的声音。	
注意	默认情况下声音是关闭的	

[返回目录](#)

4.5.2 关闭声音（独占方式） **PLAY_StopSound**

函数名称	BOOL PLAY_StopSound()	
参数说明	无	无
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.5.3 以共享方式播放声音 **PLAY_PlaySoundShare**

函数名称	BOOL PLAY_PlaySoundShare(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	以共享方式播放声音，播放本路声音而不去关闭其他路的声音	

[返回目录](#)

4.5.4 关闭声音(共享方式) **PLAY_StopSoundShare**

函数名称	BOOL PLAY_StopSoundShare(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	以共享方式关闭声音。 PLAY_PlaySound 和 PLAY_StopSound 是以独占方式播放声音的。	
注意	在同一个进程中，所有通道必须使用相同的方式播放或关闭声音	

[返回目录](#)

4.5.5 设置音量 **PLAY_SetVolume**

函数名称	BOOL PLAY_SetVolume(LONG nPort, WORD nVolume)	
参数说明	[in] LONG nPort	通道号
	[in] WORD nVolume	音量的值，范围[0, 0XFFFF]
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	设置的音量是指声卡输出的音量，会影响其他的声音应用。	

[返回目录](#)

4.5.6 获取音量 **PLAY_GetVolume**

函数名称	WORD PLAY_GetVolume(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	当前的音量值。这里的音量是指声卡输出的音量，会影响其他的声音应用	

[返回目录](#)

4.5.7 调整WAVE波形 **PLAY_AdjustWaveAudio ***

函数名称	BOOL PLAY_AdjustWaveAudio(LONG nPort, LONG nCoefficient)	
参数说明	[in] LONG nPort	通道号
	[in] LONG nCoefficient	调整的参数，范围从MIN_WAVE_COEF 到 MAX_WAVE_COEF, 0是不调整
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	调整 WAVE 波形，可以改变声音的大小。它和 PLAY_SetVolume 的不同在于，它是调整声音数据，只对该路起作用，而 PLAY_SetVolume 是调整声卡音量，对整个系统起作用。	
注意	此接口无效	


[返回目录](#)

4.6 数据回调


4.6.1 设置解码回调 **PLAY_SetDecCallBack**

函数名称	BOOL PLAY_SetDecCallBack(LONG nPort, void (CALLBACK* DecCBFun)(long nPort, char *pBuf, long nSize, FRAME_INFO *pFrameInfo, long nReserved1, long nReserved2))	
参数说明	[in] LONG nPort	通道号
	[in] void (CALLBACK* DecCBFun)	解码回调函数指针，不能为NULL
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	设置回调函数，替换播放器中的显示部分，由用户自己控制显示，该函数在 PLAY_Play 之前调用，在 PLAY_Stop 时自动失效，下次调用 PLAY_Play 之前需要重新设置。	
注意	解码部分不控制速度，只要用户从回调函数中返回，解码器就会解码下一部分数据。这个功能的使用需要用户对视频显示和声音播放有足够的了	

	解，否则请慎重使用。
--	------------

 回调函数参数说明：

回调函数	Void (CALLBACK* DecCBFun)(long nPort, char *pBuf, long nSize, FRAME_INFO *pFrameInfo, long nReserved1, long nReserved2)	
参数说明	[in] long nPort	通道号
	[in] char *pBuf	解码后的音视频数据
	[in] long nSize	解码后的音视频数据pBuf的长度
	[in] FRAME_INFO *pFrameInfo	图像和声音信息。详见下
	[in] long nReserved1	保留参数
	[in] long nReserved2	保留参数


 结构说明：

<pre>typedef struct{ long nWidth; //画面宽，单位像素。如果是音频数据则为0； long nHeight; //画面高。如果是音频数据则为0； long nStamp; //时标信息，单位毫秒。 long nType; //数据类型，T_AUDIO16，T_RGB32， T_YV12详见宏定义说明。 long nFrameRate; //编码时产生的图像帧率。 }FRAME_INFO;</pre>	
---	--

[返回目录](#)

4.6.2 设置解码回调（增加用户传递参数） **PLAY_SetDecCallBackEx**

函数名称	BOOL PLAY_SetDecCallBackEx(LONG nPort, void (CALLBACK* DecCBFun)(long nPort,char * pBuf,long nSize,FRAME_INFO * pFrameInfo, long nReserved1,long nReserved2), long nUser)	
参数说明	[in] LONG nPort	通道号
	[in] void (CALLBACK* DecCBFun)	解码回调函数指针，不能为NULL
	[in] long nUser	用户自定义回调参数
返回值	成功返回 TRUE，不成功返回 FALSE	
使用说明	和 PLAY_SetDecCallBack 的区别在于增加了用户传递参数	

 回调函数参数说明:

回调函数	void (CALLBACK* DecCBFun)(long nPort,char * pBuf,long nSize,FRAME_INFO * pFrameInfo, long nReserved1,long nReserved2)	
参数说明	[in] long nPort	通道号
	[in] char *pBuf	解码后的音视频数据
	[in] long nSize	解码后的音视频数据pBuf的长度
	[in] FRAME_INFO *pFrameInfo	图像和声音信息。详见下
	[in] long nReserved1	对应用户自定义回调参数
	[in] long nReserved2	保留参数

[返回目录](#)

4.6.3 解码回调（同时可显示视频） **PLAY_SetVisibleDecCallBack**

函数名称	BOOL PLAY_SetVisibleDecCallBack(LONG nPort, void (CALLBACK* DecCBFun)(long nPort,char * pBuf,long nSize,FRAME_INFO * pFrameInfo, long nReserved1,long nReserved2), long nUser)	
参数说明	[in] long nPort	通道号
	[in] void (CALLBACK* DecCBFun)	回调函数指针，不能为NULL
	[in] long nUser	用户自定义回调参数
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	与 PLAY_SetDecCallBackEx 基本相同，不同的是解码回调的同时可以显示视频	

[返回目录](#)


4.6.4 设置解码回调流类型 **PLAY_SetDecCBStream**

函数名称	BOOL PLAY_SetDecCBStream(LONG nPort, DWORD nStream)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nStream	1 视频流, 2 音频流, 3 复合流
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	在 PLAY_Play 之前调用	

[返回目录](#)

4.6.5 设置抓图回调PLAY_SetDisplayCallBack

函数名称	BOOL PLAY_SetDisplayCallBack(LONG nPort,void (CALLBACK* DisplayCBFun)(long nPort, char * pBuf, long nSize, long nWidth, long nHeight, long nStamp, long nType, long nReceaved), long nUser)	
参数说明	[in] LONG nPort	通道号
	[in] void (CALLBACK* DisplayCBFun)	抓图回调函数，可以为NULL；
	[in] long nUser	用户自定义数据
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	设置视频抓图回调函数。如果要停止回调，可以把回调函数指针 DisplayCBFun 设为 NULL。一旦设置回调函数，则一直有效，直到程序退出。该函数可以在任何时候调用	


 回调函数参数说明：

回调函数	void (CALLBACK* DisplayCBFun)(long nPort, char * pBuf, long nSize, long nWidth, long nHeight, long nStamp, long nType, long nReceaved)		
参数说明	[in] long nPort	通道号	
	[in] char * pBuf	返回图像数据	
	[in] long nSize	返回图像数据大小	
	[in] long nWidth	画面宽，单位像素	
	[in] long nHeight	画面高	
	[in] long nStamp	时标信息，单位毫秒	
	[in] long nType	数据类型，T_RGB32, T_UYVY，详见宏定义说明。	
	[in] long nReceaved	保留	

[返回目录](#)

4.6.6 音频解码后的WAVE数据回调PLAY_SetAudioCallBack

函数名称	BOOL PLAY_SetAudioCallBack(LONG nPort, void (CALLBACK * funAudio)(long nPort, char * pAudioBuf, long nSize, long nStamp, long nType, long nUser), long nUser)	
参数说明	[in] LONG nPort	通道号
	[in] void (CALLBACK * funAudio)	音频解码回调函数
	[in] long nUser	用户自定义数据
返回值	成功返回 TRUE，不成功返回 FALSE	


 回调函数参数说明:

回调函数	void CALLBACK FunAudio(long nPort, char * pAudioBuf, long nSize, long nStamp, long nType, long nUser)	
参数说明	[in] LONG nPort	通道号
	[in] char * pAudioBuf	wave格式音频数据
	[in] long nSize	音频数据长度
	[in] long nStamp	时标(ms)
	[in] long nType	音频类型T_AUDIO16, 采样率8000, 单声道, 每个采样点16位表示
	[in] long nUser	用户自定义数据

[返回目录](#)

4.6.7 设置数据校验PLAY_SetVerifyCallBack *

函数名称	BOOL PLAY_SetVerifyCallBack(LONG nPort, DWORD nBeginTime, DWORD nEndTime, void (CALLBACK * funVerify)(long nPort, FRAME_POS * pFilePos, DWORD bIsVideo, DWORD nUser), DWORD nUser)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nBeginTime	校验开始时间, 单位ms
	[in] DWORD nEndTime	校验结束时间, 单位ms
	[in] void (CALLBACK * funVerify)	水印回调函数指针
	[in] DWORD nUser	用户自定义数据
返回值	成功返回 TRUE, 不成功返回 FALSE	
使用描述	注册一个回调函数, 校验数据是否被修改, 实现水印功能。	
注意	此接口无效	


 回调函数参数说明:

回调函数	void CALLBACK funVerify (long nPort, FRAME_POS * pFilePos, DWORD bIsVideo, DWORD nUser)	
参数说明	[in] LONG nPort	通道号
	[in] FRAME_POS * pFilePos	帧信息
	[in] bIsVideo	是否是视频数据
	[in] DWORD nUser	用户自定义数据

[返回目录](#)

4.6.8 源数据分析完的数据回调PLAY_SetDemuxCallBack

函数名称	BOOL PLAY_SetDemuxCallBack(LONG nPort, void (CALLBACK* DemuxCBFun)(long nPort,char * pBuf,long nSize,void * pParam, long nReserved,long nUser), long nUser)	
参数说明	[in] LONG nPort	通道号
	[in] void (CALLBACK* DemuxCBFun)	分析数据回调指针
	[in] long nUser	用户自定义数据
返回值	成功返回 TRUE，不成功返回 FALSE	


 回调函数参数说明：

回调函数	void CALLBACK DemuxCBFun (long nPort,char * pBuf, long nSize, void * pParam, long nReserved, long nUser)	
参数说明	nPort	通道号
	pBuf	数据指针
	nSize	数据长度
	pParam	帧信息
	nReserved	保留
	nUser	用户自定义数据

[返回目录](#)

4.6.9 设置水印数据回调PLAY_SetWaterMarkCallBack

函数名称	BOOL PLAY_SetWaterMarkCallBack(LONG nPort, GetWaterMarkInfoCallbackFunc pFunc, long nUser)	
参数说明	[in] LONG nPort	通道号
	[in] GetWaterMarkInfoCallbackFunc pFunc	水印信息获取回调函数
	[in] long nUser	用户自定义数据
返回值	成功返回 TRUE，不成功返回 FALSE	

 回调函数参数说明：


回调函数	int (__stdcall* GetWaterMarkInfoCallbackFunc)(char* buf, long key, long len, long reallen, long reserved, long nUser)
------	---

参数说明	[in] char* buf	水印数据buffer指针
	[in] long key	区分不同水印信息
	[in] long len	缓冲的最大长度
	[in] long reallen	缓冲的实际长度
	[in] long reserved	0 I帧数据水印信息 1 帧水印 2 表水印校验 3 表智能分析帧
	[in] long nUser	用户自定义数据

[返回目录](#)

4.6.10 设置水印数据回调PLAY_SetWaterMarkCallBackEx

函数名称	BOOL PLAY_SetWaterMarkCallBackEx(LONG nPort, GetWaterMarkInfoCallbackEx pFunc, long nUser)	
参数说明	[in] LONG nPort	通道号
	[in] GetWaterMarkInfoCallbackEx pFunc	水印信息获取回调函数
	[in] long nUser	用户自定义数据
返回值	成功返回 TRUE，不成功返回 FALSE	

 回调函数参数说明：


回调函数	int (__stdcall* GetWaterMarkInfoCallbackEx)(long nPort, char* buf, long lTimeStamp, long lInfoType, long len, long reallen, long lCheckResult, long nUser)	
参数说明	[in] long nPort	通道号
	[in] char* buf	水印数据buffer指针
	[in] long lTimeStamp	水印的时间戳
	[in] long lInfoType	水印信息类型： #define WATERMARK_DATA_TEXT 0 #define WATERMARK_DATA_JPEG_BMP 1 #define WATERMARK_DATA_FRAMEDATA 3
	[in] long len	缓冲的最大长度
	[in] long reallen	缓冲的实际长度
	[in] long lCheckResult	1 没有错误 2 水印错误 3 帧数据错误

		4 帧号错误
	[in] long nUser	用户自定义数据

[返回目录](#)

4.6.11 设置编码水印数据回调 **PLAY_SetPandoraWaterMarkCallBack**

函数名称	BOOL PLAY_SetPandoraWaterMarkCallBack(LONG nPort, GetWaterMarkInfoCallbackFunc pFunc, long nUser)	
参数说明	[in] LONG nPort	通道号
	[in] GetWaterMarkInfoCallbackFunc pFunc	水印信息获取回调函数
	[in] long nUser	用户自定义数据
返回值	成功返回 TRUE，不成功返回 FALSE	

 回调函数参数说明：


回调函数	int (__stdcall* GetWaterMarkInfoCallbackFunc)(char* buf, long key, long len, long reallen, long reserved, long nUser)	
参数说明	[in] char* buf	水印数据buffer指针
	[in] long key	区分不同水印信息
	[in] long len	缓冲的最大长度
	[in] long reallen	缓冲的实际长度
	[in] long reserved	0 I帧帧数据水印信息 1 帧水印 2 表水印校验 3 表智能分析帧
	[in] long nUser	用户自定义数据

[返回目录](#)

4.6.12 切割文件 **PLAY_CutFileSegment**

函数名称	BOOL PLAY_CutFileSegment(LONG nPort, long lBeginTime, long lEndTime, CutProgressFunc pFunc, char *sOutFilePath, DWORD dwUser)	
参数说明	[in] LONG nPort	通道号
	[in] long lBeginTime	开始时间
	[in] long lEndTime	结束时间
	[in] CutProgressFunc pFunc	切割文件完成进度回调

	[in] char *sOutFilePath	文件名
	[in] DWORD dwUser	用户自定义数据
返回值	成功返回 TRUE，不成功返回 FALSE	

 回调函数参数说明：

回调函数	void (__stdcall *CutProgressFunc)(DWORD nPort, int iProgress, DWORD dwUser)	
参数说明	[in] DWORD nPort	通道号
	[in] int iProgress	切割进度[0, 100]
	[in] DWORD dwUser	用户自定义数据

[返回目录](#)

4.7 消息回调

4.7.1 分辨率改变通知消息 **PLAY_SetEncChangeMsg**

函数名称	BOOL PLAY_SetEncChangeMsg(LONG nPort, HWND hWnd, UINT nMsg)	
功能描述	设置解码时编码格式发生改变时要发送的消息	
参数说明	[in] LONG nPort	通道号
	[in] HWND hWnd	消息发送的窗口
	[in] UINT nMsg	用户输入的消息，当解码时编码格式发生改变时用户在hWnd窗口过程中收到这个消息
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.7.2 设置文件结束时要发送的消息 **PLAY_SetFileEndMsg**


函数名称	BOOL PLAY_SetFileEndMsg(LONG nPort, HWND hWnd, UINT nMsg)	
参数说明	[in] LONG nPort	通道号
	[in] HWND hWnd	消息发送的窗口
	[in] UINT nMsg	用户自定义的输入的消息，当播放到文件结束时用户在hWnd窗口过程中收到这个消息。此消息中的wParam或者lParam都可以获得nPort的值
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.8 函数回调

4.8.1 设置源缓冲区阈值及回调指针 **PLAY_SetSourceBufCallBack**

函数名称	BOOL PLAY_SetSourceBufCallBack(LONG nPort, DWORD nThreShold, void (CALLBACK * SourceBufCallBack)(long nPort, DWORD nBufSize, DWORD dwUser,void*pResvered), DWORD dwUser, void *pReserved)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nThreShold	阈值
	[in] (CALLBACK * SourceBufCallBack)	回调函数指针
	[in] DWORD dwUser	用户数据
	[in] void *pReserved	保留
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	设置源缓冲区阈值和剩余数据小于等于阈值时的回调函数指针。只有在数据量小于等于指定阈值的时候，才会触发回调。一次触发后需要调用 PLAY_ResetSourceBufFlag 接口重置参数，使再次有效。	

 回调函数参数说明：

回调函数	void CALLBACK SourceBufCallBack(long nPort, DWORD nBufSize, DWORD dwUser, void* pResvered)	
参数说明	[in] long nPort	通道号
	[in] DWORD nBufSize	缓冲中的数据长度
	[in] DWORD dwUser	用户数据
	[in] void* pResvered	保留

[返回目录](#)


4.8.2 重置回调标志为有效状态 **PLAY_ResetSourceBufFlag**

函数名称	BOOL PLAY_ResetSourceBufFlag(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.8.3 图像分辨率改变通知回调 **PLAY_SetEncTypeChangeCallBack**

函数名称	BOOL PLAY_SetEncTypeChangeCallBack(LONG nPort, void(CALLBACK *funEncChange)(long nPort, long nUser), long nUser)	
参数说明	[in] LONG nPort	通道号
	[in] void(CALLBACK *funEncChange)	回调函数
	[in] long nUser	用户自定义数据
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	设置解码时图象格式发生改变通知用户的回调函数；打开文件前使用	


 回调函数参数说明：

回调函数	void (CALLBACK *funEncChange)(long nPort, long nUser)	
参数说明	[in] LONG nPort	通道号
	[in] long nUser	用户数据

[返回目录](#)

4.8.4 设置建立索引回调 **PLAY_SetFileRefCallBack**

函数名称	BOOL PLAY_SetFileRefCallBack(LONG nPort, void (CALLBACK *pFileRefDone) (DWORD nPort, DWORD nUser), DWORD nUser)	
参数说明	[in] LONG nPort	通道号
	[in] void (CALLBACK *pFileRefDone)	回调函数指针
	[in] DWORD nUser	用户自定义数据
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	文件索引建立后回调。我们在文件打开时生成文件索引。这个过程耗时比较长，大约每秒处理 40M 左右的数据，主要是因为从硬盘读数据比较慢。建立索引的过程是在后台完成，需要使用索引的函数要等待这个过程结束，而其他接口不会受到影响。	


 回调函数参数说明：

回调函数	void FileRefDone(DWORD nPort, DWORD nUser)	
参数说明	[in] DWORD nPort	通道号
	[in] DWORD nUser	用户数据

[返回目录](#)

4.8.5 设置建立索引回调(返回索引创建情况)**PLAY_SetFileRefCallBackEx**

函数名称	BOOL PLAY_SetFileRefCallBackEx(LONG nPort, void (CALLBACK * pFileRefDoneEx) (DWORD nPort, BOOL bIndexCreated, DWORD nUser), DWORD nUser)	
参数说明	[in] LONG nPort	通道号
	[in]void (CALLBACK * pFileRefDoneEx)	回调函数指针
	[in] DWORD nUser	用户自定义数据
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	PLAY_SetFileRefCallBack 的扩展接口，可以返回索引的创建情况。	


 回调函数参数说明：

回调函数	void FileRefDoneEx(DWORD nPort, BOOL bIndexCreated, DWORD nUser)	
参数说明	[in] DWORD nPort	通道号
	[in] BOOL bIndexCreated	索引创建标志，TRUE--索引创建成功，FALSE--失败
	[in] DWORD nUser	用户数据

[返回目录](#)

4.8.6 设置文件结束回调**PLAY_SetFileEndCallBack**

函数名称	BOOL CALLMETHOD PLAY_SetFileEndCallBack(LONG nPort, void (CALLBACK *pFileEnd)(DWORD nPort, DWORD nUser), DWORD nUser)	
参数说明	[in] LONG nPort	通道号
	[in] void (CALLBACK *pFileEnd)	回调函数指针
	[in] DWORD nUser	用户自定义数据
返回值	成功返回 TRUE，不成功返回 FALSE	


 回调函数参数说明：

回调函数	void FileEnd (DWORD nPort, DWORD nUser)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nUser	用户数据

[返回目录](#)

4.8.7 开始AVI转换并设置AVI转换状态回调 **PLAY_StartAVIConvert**

函数名称	BOOL PLAY_StartAVIConvert(LONG nPort, char *sFileName, AVIConvertCallback pAVIFunc, long lUser);	
参数说明	[in] LONG nPort	通道号
	[in] char *sFileName	文件名
	[in] AVIConvertCallback pAVIFunc	回调函数指针
	[in] long lUser	用户自定义数据
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	开始录制 AVI 格式录像，当视频帧率或分辨率改变时，调用回调函数。	
注意	回调函数不能传空，为 NULL 时就不录制 AVI。	

 回调函数参数说明：

回调函数	typedef void (__stdcall* AVIConvertCallback)(long nPort, long lMediaChangeType, long lUser, BOOL *pbIfContinue, char *sNewFileName);	
参数说明	[in] LONG nPort	通道号
	[in] long lMediaChangeType	1 表示帧率改变；2 表示分辨率改变
	[in] long lUser	用户数据
	[out] BOOL *pbIfContinue	TRUE 继续转换；FALSE 停止转换
	[out] char *sNewFileName	如果继续转换，新的录像文件名

[返回目录](#)

4.8.8 停止AVI转换**PLAY_StopAVIConvert**

函数名称	BOOL PLAY_StopAVIConvert(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	与 PLAY_StartAVIConvert 配对使用	

[返回目录](#)

4.9 文件索引

4.9.1 设置文件索引 **PLAY_SetRefValue**

函数名称	BOOL PLAY_SetRefValue(LONG nPort, BYTE *pBuffer, DWORD nSize)	
参数说明	[in] nPort	LONG nPort
	[in] BYTE *pBuffer	索引信息
	[in] DWORD nSize	索引信息的长度
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	设置文件索引。如果已经有了文件索引信息，可以不再调用生成索引的回调函数 PLAY_SetFileRefCallBack ，直接输入索引信息。	
注意	索引信息及其长度必须准确	

[返回目录](#)

4.9.2 获取文件索引 **PLAY_GetRefValue**

函数名称	BOOL PLAY_GetRefValue(LONG nPort, BYTE *pBuffer, DWORD *pSize)	
参数说明	[in] LONG nPort	通道号
	[out] BYTE *pBuffer	索引信息
	[in/out] DWORD *pSize	输入 <i>pBuffer</i> 的大小，输出索引信息大小 注：可以在第一次指定 <i>pSize</i> =0, <i>pBuffer</i> =NULL, 从 <i>pSize</i> 的返回值获得需要的缓冲区大小。然后分配足够的缓冲，再调用一次
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	获取文件索引信息，以便下次打开同一个文件时直接使用这个信息。必须在索引建成后才能获得信息	

[返回目录](#)

4.10 文件定位

4.10.1 设置文件当前帧播放帧号 **PLAY_SetCurrentFrameNum**

函数名称	BOOL PLAY_SetCurrentFrameNum(LONG nPort, DWORD nFrameNum)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nFrameNum	帧序号
返回值	成功返回 TRUE，不成功返回 FALSE	

使用描述	设置当前播放播放位置到指定帧号，根据帧号来定位播放位置。
注意	此函数必须在文件索引生成之后才能调用

[返回目录](#)

4.10.2 设置文件当前播放时间（毫秒）**PLAY_SetPlayedTimeEx**

函数名称	BOOL PLAY_SetPlayedTimeEx(LONG nPort, DWORD nTime)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nTime	设置文件播放位置到指定时间。单位毫秒
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	根据时间设置文件播放位置，此接口比 PLAY_SetPlayPos 费时，但如果用时间来控制进度条（与 PLAY_GetPlayedTime(Ex)配合使用），那么可以使进度条平滑滚动	

[返回目录](#)

4.10.3 设置文件播放指针的相对位置（百分比）**PLAY_SetPlayPos**

函数名称	BOOL PLAY_SetPlayPos(LONG nPort, float fRelativePos)	
参数说明	[in] LONG nPort	通道号
	[in] float fRelativePos	范围0-100%
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.10.4 获取文件播放指针的相对位置（百分比）**PLAY_GetPlayPos**

函数名称	float PLAY_GetPlayPos(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	范围 0-100%	

[返回目录](#)

4.11 设置属性

4.11.1 设置视频参数**PLAY_SetColor**

函数名称	BOOL PLAY_SetColor(LONG nPort, DWORD nRegionNum, int nBrightness, int nContrast, int nSaturation, int nHue)	
------	---	--

参数说明	[in] LONG nPort	通道号
	[in] DWORD nRegionNum	显示区域, 参考PLAY_SetDisplayRegion; 如果只有一个显示区域(通常情况)设为0
	[in] int nBrightness	亮度, 默认64; 范围0-128
	[in] int nContrast	对比度, 默认64; 范围0-128
	[in] int nSaturation	饱和度, 默认64; 范围0-128
	[in] int nHue	色调, 默认64; 范围0-128
返回值	成功返回 TRUE, 不成功返回 FALSE	

[返回目录](#)

4.11.2 设置播放缓冲区最大缓冲帧数PLAY_SetDisplayBuf

函数名称	BOOL PLAY_SetDisplayBuf(LONG nPort, DWORD nNum)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nNum	播放缓冲区最大缓冲帧数。范围： MIN_DIS_FRAMES ~ MAX_DIS_FRAMES 。一帧352*288图像的所需内存最小值是 352*288*3/2 大约150K。最大值是352*288*4大约405K MIN_DIS_FRAMES 播放缓冲最小值 MAX_DIS_FRAMES 播放缓冲最大值
返回值	成功返回 TRUE, 不成功返回 FALSE	
使用说明	设置播放缓冲区（即解码后的图像缓冲区）大小；这个缓冲区比较重要，他直接影响播放的流畅性和延时性。在一定范围内缓冲越大越流畅，同时延时越大。在播放文件时用户最好可以考虑开大缓冲（如果内存足够大），我们的默认值是 15（帧），在 25 帧/秒的情况下即 0.6 秒的数据。如果用户追求最大延时最小，可以考虑试当减小这个值	
注意	在 PLAY_Play 之前调用有效	

[返回目录](#)

4.11.3 设置显示模式PLAY_SetDisplayType

函数名称	BOOL PLAY_SetDisplayType(LONG nPort, LONG nType)	
参数说明	[in] LONG nPort	通道号
	[in] LONG nType	两种模式： DISPLAY_NORMAL 正常分辨率数据送显卡显示 DISPLAY_QUARTER 1/4分辨率数据送显卡显示

返回值	成功返回 TRUE，不成功返回 FALSE
使用说明	设置显示的模式，在小画面显示时，采用 DISPLAY_QUARTER 可以减小显卡工作量，从而支持更多路显示，但画面显示质量有下降。在正常和大画面显示时应该使用 DISPLAY_NORMAL

[返回目录](#)

4.11.4 垂直同步显示开关**PLAY_VerticalSyncEnable**

函数名称	BOOL PLAY_VerticalSyncEnable(LONG nPort, BOOL bEnable)	
参数说明	[in] LONG nPort	通道号
	[in] BOOL bEnable	TRUE 打开垂直同步 FALSE 关闭垂直同步
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	只支持offscreen显示模式。此接口需在 PLAY_Play 之后调用，重新播放时需重新设置。在播放动态图像出现断层时，可以使用此接口打开垂直同步功能，但CPU占用率会明显提高	

[返回目录](#)

4.11.5 调整图像流畅性**PLAY_AdjustFluency ***

函数名称	BOOL PLAY_AdjustFluency(LONG nPort, int level)	
参数说明	[in] LONG nPort	通道号
	[in] int level	将要调整图象的等级(0-6)
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	调整图象播放的流畅性。流畅性和实时性是一对平衡体。当 level 为 0 时，图象最流畅；当 level 为 6 时，图象最实时。Level 的默认值为 3	
注意	此接口无效	

[返回目录](#)


4.11.6 改变图像播放的帧率**PLAY_ChangeRate**

函数名称	BOOL PLAY_ChangeRate(LONG nPort, int rate)	
参数说明	[in] LONG nPort	通道号
	[in] int rate	播放帧率
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.11.7 打开音频采集功能**PLAY_OpenAudioRecord**

函数名称	BOOL PLAY_OpenAudioRecord(pCallFunction nProc, LONG nBitsPerSample, LONG nSamplesPerSec, long nLength, long nReserved, LONG nUser)	
参数说明	[in] pCallFunction nProc	音频采集数据回调指针
	[in] LONG nBitsPerSample	表示每个采样所需要的位数
	[in] LONG nSamplesPerSec	采样率
	[in] LONG nLength	数据缓冲的长度
	nReserved	保留
	[in] LONG nUser	用户自定义数据
返回值	成功返回 TRUE，不成功返回 FALSE	

 回调函数参数说明：

回调函数	void pCallFunction(LPBYTE pDataBuffer, DWORD DataLength, long nUser)	
参数说明	[in] LPBYTE pDataBuffer	回调数据指针
	[in] DWORD DataLength	回调数据长度
	[in] DWORD nUser	用户数据

[返回目录](#)

4.11.8 关闭音频采集功能**PLAY_CloseAudioRecord**

函数名称	BOOL PLAY_CloseAudioRecord()	
参数说明	无	无
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.11.9 设置Overlay模式和关键色**PLAY_SetOverlayMode**

函数名称	BOOL PLAY_SetOverlayMode(LONG nPort, BOOL bOverlay, COLORREF colorKey)	
参数说明	[in] LONG nPort	音频采集数据回调指针

	[in] BOOL bOverlay	TRUE 表示将首先尝试使用 OVERLAY 模式，如果不行再使用其他模式 FALSE则不进行 OVERLAY 模式的尝试
	[in] COLORREF colorKey	用户设置的透明色，透明色相当于一层透视图膜，显示的画面只能穿过这种颜色，而其他的颜色将挡住显示的画面。用户应该在显示窗口中涂上这种颜色，那样才能看到显示画面。一般应该使用一种不常用的颜色作为透明色。这是一个双字节值0x00rrggbb,最高字节为0，后三个字节分别表示r,g,b的值
返回值	成功返回 TRUE，不成功返回 FALSE	
使用说明	<p>设置OVERLAY模式显示画面。在一块显卡中同一时刻只能有一个OVERLAY表面处于活动状态，如果此时系统中已经有程序使用了OVERLAY，那么播放器就不能再创建OVERLAY表面，它将自动改用Off_Screen表面，并不返回FALSE。一些常用的播放器，以及我们卡的预览都可能使用了OVERLAY表面。同样，如果播放器使用了OVERLAY表面，那么，其他的程序将不能使用OVERLAY表面，特别注意，我们的卡在预览时可能也要使用OVERLAY (用户可设置)，如果先打开播放器（并且使用了OVERLAY），再启动预览，那么预览可能因为得不到OVERLAY而失败。</p> <p>使用 OVERLAY 模式的优点是：大部份的显卡都支持 OVERLAY，在一些不支持 BLT 硬件缩放和颜色转换的显卡上（如 SIS 系列显卡）使用OVERLAY模式（OVERLAY模式下的缩放和颜色转换由显卡支持），可以大大减小 CPU 利用率并提高画面质量（相对于软件缩放和颜色转换）。缺点是：只能有一路播放器使用。该设置必须在 PLAY 之前使用，而且需要设置透明色。</p>	

[返回目录](#)

4.11.10 设置图像质量**PLAY_SetPicQuality**

函数名称	BOOL PLAY_SetPicQuality(LONG nPort, BOOL bHighQuality)	
参数说明	[in] LONG nPort	通道号
	[in] BOOL bHighQuality	等于1时图像高质量，等于0时低质量（默认值）
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	设置图像质量，当设置成高质量时画面效果好，但 CPU 利用率高。在支	

	持多路播放时，可以设为低质量，以降低 CPU 利用率；当某路放大播放时将该路设置成高质量，以达到好的画面效果
--	--

[返回目录](#)

4.11.11 设置流播放模式 **PLAY_SetStreamOpenMode**

函数名称	BOOL PLAY_SetStreamOpenMode(LONG nPort, DWORD nMode)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nMode	STREAME_REALTIME 实时模式（默认） STREAME_FILE 文件模式 实时模式，适合播放网络实时数据，解码器会立刻解码； 文件模式，适合用户把文件数据用流方式输入；
返回值	成功返回 TRUE，不成功返回 FALSE	
注意	必须在播放之前设置	

[返回目录](#)

4.11.12 设置播放使用的定时器类型 **PLAY_SetTimerType**

函数名称	BOOL PLAY_SetTimerType(LONG nPort, DWORD nTimerType, DWORD nReserved)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nTimerType	TIMER_1 多媒体定时器，精度高，但一个进程中不能超过16个； TIMER_2 线程定时器，精度略低，无数量限制
	[in] DWORD nReserved	保留
返回值	成功返回 TRUE，不成功返回 FALSE	
注意	必须在 Open 之前调用	

[返回目录](#)

4.11.13 设置每个定时器管理的视频个数 **PLAY_SetVideoPerTimer**

函数名称	BOOL PLAY_SetVideoPerTimer(int iVal)	
参数说明	[in] int iVal	每个定时器管理几个视频

返回值	成功返回 TRUE，不成功返回 FALSE
-----	-----------------------

[返回目录](#)

4.12 获得属性

4.12.1 测试播放所需系统功能PLAY_GetCaps

函数名称	int PLAY_GetCaps()	
参数说明	无	
返回值	属性值，按位取。	

注：属性值，1~9 位分别表示以下信息（位与是 TRUE 表示支持）

SUPPORT_DDRAW	支持 DIRECTDRAW；如果不支持，则播放器不能工作
SUPPORT_BLT	显卡支持 BLT 操作；如果不支持，则播放器不能工作
SUPPORT_BLTFOURCC	显卡BLT支持颜色转换；如果不支持，播放器会使用软件方式作RGB转换
SUPPORT_BLTSHRINKX	显卡 BLT 支持 X 轴缩小；如果不支持，系统使用软件方式转换
SUPPORT_BLTSHRINKY	显卡 BLT 支持 Y 轴缩小；如果不支持，系统使用软件方式转换
SUPPORT_BLTSTRETCHX	显卡 BLT 支持 X 轴放大；如果不支持，系统使用软件方式转换
SUPPORT_BLTSTRETCHY	显卡 BLT 支持 Y 轴放大；如果不支持，系统使用软件方式转换
SUPPORT_SSE	CPU 支持 SSE 指令, Intel Pentium3 以上支持 SSE 指令
SUPPORT_MMX	CPU 支持 MMX 指令集

[返回目录](#)

4.12.2 获取视频参数PLAY_GetColor

函数名称	BOOL PLAY_GetColor(LONG nPort, DWORD nRegionNum, int *pBrightness, int *pContrast, int *pSaturation, int *pHue)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nRegionNum	显示区域，参考PLAY_SetDisplayRegion；如果只有一个显示区域(通常情况)设为0
	[out] int *pBrightness	亮度，默认64； 范围0-128
	[out] int *pContrast	对比度，默认64； 范围0-128
	[out] int *pSaturation	饱和度，默认64； 范围0-128
	[out] int *pHue	色调，默认64； 范围0-128

返回值	成功返回 TRUE，不成功返回 FALSE
-----	-----------------------

[返回目录](#)

4.12.3 获取播放缓冲区最大缓冲帧数 **PLAY_GetDisplayBuf**

函数名称	DWORD PLAY_GetDisplayBuf(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	播放缓冲区最大缓冲帧数	

[返回目录](#)

4.12.4 获取显示模式 **PLAY_GetDisplayType**

函数名称	long PLAY_GetDisplayType(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	DISPLAY_NORMAL 或 DISPLAY_QUARTER	

[返回目录](#)

4.12.5 获取OVERLAY关键色 **PLAY_GetColorKey**

函数名称	long PLAY_GetDisplayType(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	颜色值	

[返回目录](#)

4.12.6 检查当前是否使用了OVERLAY显示模式 **PLAY_GetOverlayMode**

函数名称	LONG PLAY_GetOverlayMode(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	0 表示没有使用 OVERLAY，1 表示使用了 OVERLAY 表面	

[返回目录](#)

4.12.7 获取图像质量 **PLAY_GetPictureQuality**

函数名称	BOOL PLAY_GetPictureQuality(LONG nPort, BOOL* bHighQuality)	
------	---	--

参数说明	[in] LONG nPort	通道号
	[out] BOOL* bHighQuality	1表示高质量, 0表示低质量
返回值	成功返回 TRUE, 不成功返回 FALSE	

[返回目录](#)

4.12.8 获取流播放模式**PLAY_GetStreamOpenMode**

函数名称	LONG PLAY_GetStreamOpenMode(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	STREAME_REALTIME 或 STREAME_FILE	

[返回目录](#)

4.12.9 获取播放使用的定时器类型**PLAY_GetTimerType**

函数名称	BOOL PLAY_GetTimerType(LONG nPort, DWORD *pTimerType, DWORD *pReserved)	
参数说明	[in] LONG nPort	通道号
	[out] DWORD *pTimerType	定时器类型, TIMER_1或TIMER_2
	DWORD *pReserved	保留
返回值	成功返回 TRUE, 不成功返回 FALSE	

[返回目录](#)

4.12.10 获取指定缓冲区的大小**PLAY_GetBufferValue**

函数名称	DWORD PLAY_GetBufferValue(LONG nPort, DWORD nBufType)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nBufType	缓冲区类型, 见 宏定义
返回值	根据不同参数返回缓冲区值, 源缓冲区返回 byte, 解码后缓冲区返回帧数	
使用描述	获取播放器中的缓冲区大小(帧数或者 byte)。这个接口可以帮助用户了解缓冲区中的数据, 从而在网络延时方面有所估计	

[返回目录](#)

4.12.11 获取当前播放的帧序号**PLAY_GetCurrentFrameNum**

函数名称	DWORD PLAY_GetCurrentFrameNum(LONG nPort)
------	---

参数说明	[in] LONG nPort	通道号
返回值	当前播放的帧序号	
使用描述	得到当前播放的帧序号。而 PLAY_GetPlayedFrames 是总共解码的帧数。如果文件播放位置不被改变，那么这两个函数的返回值应该非常接近，除非码流丢失数据	

[返回目录](#)

4.12.12 获取当前帧率PLAY_GetCurrentFrameRate

函数名称	DWORD PLAY_GetCurrentFrameRate(LONG nPort)	
参数说明	nPort	通道号
返回值	当前码流中编码时的帧率值	

[返回目录](#)

4.12.13 获取文件头长度PLAY_GetFileHeadLength

函数名称	DWORD PLAY_GetFileHeadLength()	
参数说明	无	无
返回值	此版本播放器对应的文件头的长度	
使用描述	得到当前版本播放器能播放的文件的文件头长度	

[返回目录](#)

4.12.14 获取文件总时间PLAY_GetFileTime

函数名称	DWORD PLAY_GetFileTime(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	文件总的时间长度值，单位秒	

[返回目录](#)

4.12.15 获取文件总帧数PLAY_GetFileTotalFrames

函数名称	DWORD PLAY_GetFileTotalFrames(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	文件中的总帧数	

[返回目录](#)

4.12.16 查找指定位置之前的关键帧位置PLAY_GetKeyFramePos

函数名称	BOOL PLAY_GetKeyFramePos(LONG nPort,DWORD nValue, DWORD nType, PFRAME_POS pFramePos)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nValue	当前位置，可以是时间或帧号，类型由nType指定
	[in] DWORD nType	指定nValue的类型。 如果nType是BY_FRAMEENUM，则nValue表示帧号； 如果nType是BY_FRAMETIME，则nValue表示时间，单位ms
	[out] PFRAME_POS pFramePos	查找到的关键帧的文件位置信息结构指针。 typedef struct{ long nFilePos; //文件位置; long nFrameNum; //帧序号; long nFrameTime; //帧时标（ms）; }FRAME_POS,*PFRAME_POS
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	查找指定位置之前的关键帧位置信息。图像解码必须从关键帧开始，如果用户保存的文件不是从关键帧开始的，那么倒下一个关键帧之前的数据会被忽略。如果用户要截取文件中的一段数据，则应该考虑从关键帧开始截取。结束位置则关系不大，最多丢失 3 帧数据。 在文件索引建立完全的前提下，与 PLAY_GetNextKeyFramePos 联合使用，可以用来实现剪辑录像文件，剪辑精度与关键帧间隔有关。	

[返回目录](#)

4.12.17 查找指定位置之后的关键帧位置PLAY_GetNextKeyFramePos

函数名称	BOOL PLAY_GetNextKeyFramePos(LONG nPort, DWORD nValue, DWORD nType, PFRAME_POS pFramePos)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nValue	当前位置，可以是时间或帧号，类型由nType指定
	[in] DWORD nType	指定nValue的类型。 如果nType是BY_FRAMEENUM，则nValue表示帧号； 如果nType是BY_FRAMETIME，则nValue表示时间，

		单位ms
	[out] PFRAME_POS pFramePos	查找到的关键帧的文件位置信息结构指针。 typedef struct{ long nFilePos; //文件位置; long nFrameNum; //帧序号; long nFrameTime; //帧时标 (ms) ; }FRAME_POS,*PFRAME_POS
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	见 PLAY_GetKeyFramePos 的使用描述。	

[返回目录](#)

4.12.18 获取原始图像大小PLAY_GetPictureSize

函数名称	BOOL PLAY_GetPictureSize(LONG nPort, LONG *pWidth, LONG *pHeight)	
参数说明	[in] LONG nPort	通道号
	[out] LONG *pWidth	原始图像的宽。
	[out] LONG *pHeight	原始图像的高。
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	获得码流中原始图像的大小，根据此大小来设置显示窗口的区域，可以不用显卡做缩放工作，对于那些不支持硬件缩放的显卡来说非常有用	

[返回目录](#)

4.12.19 获取已解码的视频帧数PLAY_GetPlayedFrames

函数名称	DWORD PLAY_GetPlayedFrames(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	已经解码的视频帧数	

[返回目录](#)

4.12.20 获取当前文件播放时间PLAY_GetPlayedTime

函数名称	DWORD PLAY_GetPlayedTime(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	文件当前播放的时间，单位秒	

[返回目录](#)

4.12.21 获取文件当前播放时间（毫秒）**PLAY_GetPlayedTimeEx**

函数名称	DWORD PLAY_GetPlayedTimeEx(LONG nPort)	
参数说明	nPort	通道号
返回值	文件当前播放的时间，单位毫秒	

[返回目录](#)

4.12.22 信息状态查询**PLAY_QueryInfo**

函数名称	BOOL PLAY_QueryInfo(LONG nPort , int cmdType, char* buf, int buflen, int* returnlen)	
参数说明	[in] LONG nPort	通道号
	[in] int cmdType	指定状态查询指令 PLAY_CMD_GetTime 获取编码中时间信息，单位ms PLAY_CMD_GetFileRate 获取帧率信息 PLAY_CMD_GetMediaInfo 获取媒体信息，信息结构为 MEDIA_INFO
	[out] char* buf	存放信息的缓冲
	[in] int buflen	缓冲长度
	[out] int* returnlen	获取的信息的有效数据长度
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	目前实现了对当前时间和帧率信息的查询	

Ex：

```
int len;
MEDIA_INFO tMediaInfo;
PLAY_QueryInfo(0, PLAY_CMD_GetMediaInfo, (char*)&tMediaInfo, sizeof(MEDIA_INFO), &len);

typedef struct {
int nYear;
int nMonth;
int nDay;
int nHour;
int nMinute;
```

```
int nSecond;
}tTime;

tTime tm;
int len;

PLAY_QueryInfo(nPort, PLAY_CMD_GetTime, (char*)&tm, sizeof(tTime), &len);
```

[返回目录](#)

4.12.23 获取流播放模式下源缓冲区剩余数据大小**PLAY_GetSourceBufferRemain**

函数名称	DWORD PLAY_GetSourceBufferRemain(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	当前源缓冲的大小 (BYTE)	

[返回目录](#)

4.12.24 获取视频实时码率**PLAY_GetRealFrameBitRate**

函数名称	BOOL PLAY_GetRealFrameBitRate(LONG nPort, double* pBitRate)	
参数说明	[in] LONG nPort	通道号
	[out] double* pBitRate	视频码率
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.12.25 获取视频实时码率**PLAY_GetVideoPerTimer**

函数名称	BOOL PLAY_GetVideoPerTimer(int* pVal)	
参数说明	[out] int* pVal	每个通道管理的视频个数
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.13 多屏显示控制

4.13.1 枚举系统中的显示设备 **PLAY_InitDDrawDevice ***

函数名称	BOOL PLAY_InitDDrawDevice()	
参数说明	无	无
返回值	成功返回 TRUE，不成功返回 FALSE	
注意	此接口无效	

[返回目录](#)

4.13.2 释放枚举显示设备的过程中分配的资源 **PLAY_ReleaseDDrawDevice ***

函数名称	void PLAY_ReleaseDDrawDevice()	
参数说明	无	无
返回值	无	
注意	此接口无效	

[返回目录](#)

4.13.3 设置播放窗口使用的显示设备 **PLAY_SetDDrawDevice ***

函数名称	BOOL PLAY_SetDDrawDevice(LONG nPort, DWORD nDeviceNum)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nDeviceNum	显示设备的设备号。如果是0，表示主显示设备
返回值	成功返回 TRUE，失败返回 FALSE	
注意	此接口无效	

[返回目录](#)

4.13.4 设置播放窗口使用的显示设备（多显示区域） **PLAY_SetDDrawDeviceEx ***

函数名称	BOOL PLAY_SetDDrawDeviceEx(LONG nPort, DWORD nRegionNum, DWORD nDeviceNum)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nRegionNum	显示区域
	[in] DWORD nDeviceNum	显示设备的设备号
返回值	成功返回 TRUE，失败返回 FALSE	

注意	此接口无效
----	-------

[返回目录](#)

4.13.5 获取指定显卡和监视器信息 **PLAY_GetDDrawDeviceInfo ***

函数名称	BOOL PLAY_GetDDrawDeviceInfo(DWORD nDeviceNum, LPSTR lpDriverDescription, DWORD nDespLen, LPSTR lpDriverName, DWORD nNameLen, HMONITOR *hhMonitor)	
参数说明	[in] DWORD nDeviceNum	显示设备的设备号，如果是0，则表示主显示设备
	[out] LPSTR lpDriverDescription	显示设备的描述信息
	[in] DWORD nDespLen	表示lpDriverDescription已分配空间的大小，单位byte
	[out] LPSTR lpDriverName	显示设备的设备名
	[in] DWORD nNameLen	表示lpDriverName已分配空间的大小，单位byte
	[out] HMONITOR *hhMonitor	显示设备使用的监视器句柄，通过Windows API 函数GetMonitorInfo，可以得到详细信息，供用户定位窗口位置
返回值	成功返回 TRUE，不成功返回 FALSE	
注意	此接口无效	

[返回目录](#)

4.13.6 获取显示设备（显卡）个数 **PLAY_GetDDrawDeviceTotalNums ***

函数名称	DWORD PLAY_GetDDrawDeviceTotalNums()	
参数说明	无	无
返回值	0 表示系统中只有主显示设备 1 表示系统中安装了多块显卡，但只有一块显卡与Windows桌面绑定 其他值 表示系统中与桌面绑定的显卡数目。在多显卡的系统中可以通过设置显示属性，而指定任意一块显卡作为主显示设备	
使用说明	获得系统中与windows桌面绑定的总的显示设备数目(这里主要是指显卡)	
注意	此接口无效	

[返回目录](#)

4.13.7 获取指定显示设备的系统信息 **PLAY_GetCapsEx ***

函数名称	int PLAY_GetCapsEx(DWORD nDDrawDeviceNum)	
参数说明	[in] DWORD nDDrawDeviceNum	指定显示设备的设备号，如果是0，表示主显示设备。
返回值	指定显示设备的系统信息	
注意	此接口无效	

[返回目录](#)

补充说明：目前的多显卡显示方式不需要用户调用，SDK内部会自动显示在窗口所在显示器所对应的显卡上。

4.14 抓图

4.14.1 图像格式转为BMP格式 **PLAY_ConvertToBmpFile**

函数名称	BOOL PLAY_ConvertToBmpFile(char * pBuf,long nSize,long nWidth,long nHeight,long nType,char *sFileName)	
参数说明	[in] char * pBuf	图像数据指针
	[in] long nSize	图像数据大小
	[in] long nWidth	图像宽度
	[in] long nHeight	图像高度
	[in] long nType	数据类型。T_RGB32, T_UYVY等
	[in] char *sFileName	要保存的文件名。最好以BMP作为文件扩展名
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	将 YUV 图像数据保存成 BMP 文件。转换函数占用的 cpu 资源，如果不需要保存图片，则不要调用	

[返回目录](#)

4.14.2 图像格式转为JPEG格式 **PLAY_ConvertToJpegFile**

函数名称	BOOL PLAY_ConvertToJpegFile(char *pYUVBuf, long nWidth, long nHeight, int YUVtype, int quality, char *sFileName)	
参数说明	[in] char *pYUVBuf	图像数据指针
	[in] long nWidth	图像宽度

	[in] long nHeight	图像高度
	[in] int YUVtype	YUV数据类型，如T_YV12, T_UYVY
	[in] int quality	图片压缩质量，(0, 100]
	[in] char *sFileName	要保存的文件名。最好以jpg作为文件扩展名
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	将 YUV 图像数据压缩成 jpeg 格式，用法参见 PLAY_ConvertToBmpFile	

[返回目录](#)

4.14.3 直接抓取图像（BMP）PLAY_CatchPic

函数名称	BOOL PLAY_CatchPic(LONG nPort, char* sFileName)	
参数说明	[in] LONG nPort	通道号
	[in] char* sFileName	文件名称
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	抓图，将BMP图片保存为指定的文件。 PLAY_SetDisplayCallBack 设置的视频数据回调函数，只有在有视频数据解码出来时才调用，并由用户处理视频数据（如抓图），如果不断有解码的数据，就不断调用这个回调函数。而PLAY_CatchPic一次只抓一幅图，并能在暂停和单帧播放时实现抓图。建议：如果用户想实现抓图（一次抓一幅图），调用 PLAY_CatchPic ，而如果想得到一段时间内的视频数据，调用 PLAY_SetDisplayCallBack	

4.14.4 直接抓取图像（可选择格式）PLAY_CatchPicEx

函数名称	BOOL PLAY_CatchPicEx(LONG nPort, char* sFileName, tPicFormats ePicformat)	
参数说明	[in] LONG nPort	通道号
	[in] char* sFileName	文件名称
	[in] tPicFormats ePicformat	图片格式，见 tPicFormats
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	PLAY_CatchPic 的扩展接口，可指定图片格式，目前支持BMP，JPEG。	

[返回目录](#)

4.14.5 抓图（可选择格式并指定宽高）PLAY_CatchResizePic

函数名称	BOOL PLAY_CatchResizePic(LONG nPort, char* sFileName, LONG lTargetWidth,
------	--

	LONG lTargetHeight, tPicFormats ePicformat)	
参数说明	[in] LONG nPort	通道号
	[in] char* sFileName	文件名称
	[in] LONG lTargetWidth	指定的图像宽度
	[in] LONG lTargetHeight	指定的图像高度
	[in] tPicFormats ePicformat	图片格式，见 tPicFomats
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.14.6 直接抓取BMP图像PLAY_GetPicBMP

函数名称	BOOL PLAY_GetPicBMP(LONG nPort, PBYTE pBmpBuf, DWORD dwBufSize, DWORD* pBmpSize)	
参数说明	[in] LONG nPort	通道号
	[in] PBYTE pBmpBuf	存放 BMP 图像数据的缓冲地址，由用户分配，不得小于 bmp 图像大小，推荐大小： sizeof(BITMAPFILEHEADER) + sizeof(BITMAPINFOHEADER) + w * h * 4，其中 w 和 h 分别为图像宽高
	[in] DWORD dwBufSize	申请的缓冲区大小
	[out] DWORD* pBmpSize	获取到的实际bmp 图像大小
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.14.7 直接抓取JPEG图像PLAY_GetPicJPEG

函数名称	BOOL PLAY_GetPicJPEG(LONG nPort, PBYTE pJpegBuf, DWORD dwBufSize, DWORD* pJpegSize, int quality)	
参数说明	[in] LONG nPort	通道号
	[in] PBYTE pJpegBuf	用于存放 JPEG 图像数据的缓冲地址，由用户分配，不得小于 JPEG 图像大小，推荐大小： $w * h * 3/2$ ，其中 w 和 h 分别为图像宽高
	[in] DWORD dwBufSize	申请的缓冲区大小
	[out] DWORD* pJpegSize	获取到的实际JPEG图像大小
	[in] int quality	Jpeg图像的压缩质量，取值范围为 (0, 100]


返回值	成功返回 TRUE，不成功返回 FALSE
-----	-----------------------

[返回目录](#)

4.15 字符叠加

4.15.1 画图回调 **PLAY_RigisterDrawFun**

函数名称	BOOL PLAY_RigisterDrawFun(LONG nPort, void (CALLBACK* DrawFun)(long nPort, HDC hDc, LONG nUser), LONG nUser)	
参数说明	[in] LONG nPort	通道号
	[in] void (CALLBACK* DrawFun)	画图回调函数
	[in] nUser	用户数据
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	注册一个回调函数，获得当前表面的 device context，你可以在这个 DC 上画图（或写字），就好像在窗口的客户区 DC 上绘图，但这个 DC 不是窗口客户区的 DC，而是 DirectDraw 里的 <i>Off-Screen</i> 表面的 DC。注意，如果是使用 <i>overlay</i> 表面，这个接口无效，你可以直接在窗口上绘图，只要不是透明色就不会被覆盖。	

 回调函数参数说明：


函数名称	void CALLBACK DrawFun(long nPort, HDC hDc, LONG nUser)	
参数说明	[in] LONG nPort	通道号
	[in] HDC hDc	OffScreen表面设备上下文，你可以像操作显示窗口客户区DC那样操作它
	[in] nUser	用户数据，就是上面输入的用户数据

[返回目录](#)

4.15.2 画图回调(多显示区域) **PLAY_RigisterDrawFunEx**

函数名称	BOOL PLAY_RigisterDrawFunEx(LONG nPort, LONG nReginNum, void (CALLBACK* DrawFunEx)(long nPort,long nReginNum,HDC hDc,LONG nUser),LONG nUser)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nRegionNum	显示区域序号，

		0~(MAX_DISPLAY_WND-1)。如果 <i>nRegionNum</i> 为 0，则将设置的区域显示在主窗口中
	[in] void (CALLBACK* DrawFun)	画图回调函数
	[in] nUser	用户数据
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	注册一个回调函数，获得当前表面的 device context，你可以在这个 DC 上画图（或写字），就好像在窗口的客户区 DC 上绘图，但这个 DC 不是窗口客户区的 DC，而是 DirectDraw 里的 Off-Screen 表面的 DC。注意，如果是使用 overlay 表面，这个接口无效，你可以直接在窗口上绘图，只要不是透明色就不会被覆盖。	

 回调函数参数说明：

函数名称	void CALLBACK DrawFun(long nPort, HDC hDc, LONG nUser)	
参数说明	[in] LONG nPort	通道号
	[in] HDC hDc	OffScreen 表面设备上下文，你可以像操作显示窗口客户区 DC 那样操作它
	[in] nUser	用户数据，就是上面输入的用户数据

[返回目录](#)

4.16 多区域显示

4.16.1 设置或增加显示区域 **PLAY_SetDisplayRegion**

函数名称	BOOL PLAY_SetDisplayRegion(LONG nPort, DWORD nRegionNum, RECT *pSrcRect, HWND hDestWnd, BOOL bEnable)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nRegionNum	显示区域序号，0~(MAX_DISPLAY_WND-1)。如果 <i>nRegionNum</i> 为 0，则将设置的区域显示在主窗口中
	[in] RECT *pSrcRect	局部显示区域
	[in] HWND hDestWnd	显示窗口句柄
	[in] BOOL bEnable	打开（设置）或关闭显示区域
返回值	成功返回 TRUE，不成功返回 FALSE	

使用描述	设置或增加显示区域。可以做局部放大显示。
------	----------------------

[返回目录](#)

4.16.2 刷新显示（多显示区域）**PLAY_RefreshPlayEx**

函数名称	BOOL PLAY_RefreshPlayEx(LONG nPort, DWORD nRegionNum)	
功能描述	刷新显示，刷新多区域显示的窗口	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nRegionNum	显示区域序号
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	刷新显示，同 PLAY_RefreshPlay。为支持 PLAY_SetDisplayRegion 而增加一个参数	

[返回目录](#)

4.17 数据流录像

4.17.1 开始流数据录像**PLAY_StartDataRecord**

函数名称	BOOL PLAY_StartDataRecord(LONG nPort, char *sFileName, int idataType=0)	
参数说明	[in] LONG nPort	通道号
	[in] char *sFileName	录像文件名，如果文件名中有不存在的文件夹，就创建该文件夹
	[in] int idataType	0 表示原始视频流；1表示转换成AVI格式
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	只对流模式有用，在 PLAY_Play 之后调用	

[返回目录](#)

4.17.2 停止流数据录像**PLAY_StopDataRecord**

函数名称	BOOL PLAY_StopDataRecord(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.17.3 开始AVI数据录像(可指定视频宽高)**PLAY_StartAVIResizeConvert**

函数名称	BOOL PLAY_StartAVIResizeConvert(LONG nPort, char *sFileName, long lWidth, long lHeight);	
参数说明	[in] LONG nPort	通道号
	[in] char *sFileName	录像文件名，如果文件名中有不存在的文件夹，就创建该文件夹
	[in] long lWidth	视频格式宽度
	[in] long lHeight	视频格式高度
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	只对流模式有用，在 PLAY_Play 之后调用	

[返回目录](#)

4.17.4 停止AVI数据录像**PLAY_StopAVIResizeConvert**

函数名称	BOOL PLAY_StopAVIResizeConvert(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	与 PLAY_StartAVIResizeConvert 配对使用	

[返回目录](#)

4.18 清缓冲

4.18.1 清空流播放模式下源缓冲区的剩余数据**PLAY_ResetSourceBuffer**

函数名称	BOOL PLAY_ResetSourceBuffer(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	

[返回目录](#)

4.18.2 清空指定缓冲区的剩余数据**PLAY_ResetBuffer**

函数名称	BOOL PLAY_ResetBuffer(LONG nPort, DWORD nBufType)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nBufType	缓冲类型，见 宏定义
返回值	成功返回 TRUE，不成功返回 FALSE	

☞ 缓冲类型定义:

BUF_VIDEO_SRC	视频数据源缓冲, 缓冲解码之前视频数据, 只对流模式有效, 单位 byte
BUF_AUDIO_SRC	音频数据源缓冲, 缓冲解码之前音频数据, 只对流模式有效, 单位 byte
BUF_VIDEO_RENDER	解码后视频数据缓冲, 单位帧数
BUF_AUDIO_RENDER	解码后音频数据缓冲区, 单位帧数, 音频 40ms 数据定为一帧

[返回目录](#)

4.19 智能搜索

该部分功能只对有数据帧信息的文件有效

数据帧指音频数据和视频数据之外的数据块, 譬如通道标题, LOGO 等。

4.19.1 设置搜索区域及范围 **PLAY_SetMDRange**

函数名称	BOOL CALLMETHOD PLAY_SetMDRange(LONG nPort, RECT* rc, DWORD nVauleBegin, DWORD nValueEnd, DWORD nType)	
参数说明	[in] LONG nPort	通道号
	[in] RECT* rc	搜索区域, 为播放画面的某一部分
	[in] DWORD nVauleBegin	搜索范围的上限, 可以是时间或帧号, 取决于 <i>nType</i>
	[in] DWORD nValueEnd	搜索范围的下限, 可以是时间或帧号, 取决于 <i>nType</i>
	[in] DWORD nType	可以是 BY_FRAMENUM 或 BY_FRAMETIME
返回值	成功返回 TRUE, 不成功返回 FALSE	

[返回目录](#)

4.19.2 设置智能搜索的阈值 **PLAY_SetMDThreShold**

函数名称	BOOL PLAY_SetMDThreShold(LONG nPort, DWORD ThreShold)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD ThreShold	阈值
返回值	成功返回 TRUE, 不成功返回 FALSE	

[返回目录](#)

4.19.3 获取搜索到的数据帧的帧序号PLAY_GetMDPosition

函数名称	DWORD CALLMETHOD PLAY_GetMDPosition(LONG nPort, DWORD Direction, DWORD nFrame, DWORD* MDValue)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD Direction	搜索方向 0 向前搜索 1 向后搜索
	[in] DWORD nFrame	参考帧
	[in] DWORD* MDValue	搜索到的数据帧的阈值
返回值	搜索到的动检帧的帧序号	

[返回目录](#)

4.20 获得版本号

4.20.1 获取播放库SDK版本号、次版本号和补丁号PLAY_GetSdkVersion

函数名称	DWORD PLAY_GetSdkVersion()	
参数说明	无	无
返回值	高 16 位表示当前的主版本号。9~16 位表示次版本号，1~8 位表示次补丁号。如：返回值 0x00030107 表示：主版本号是 3，次版本号是 1，补丁号是 7	
使用描述	得到当前播放器 sdk 的主版本号、次版本号和补丁号	

[返回目录](#)

4.21 获得错误号

4.21.1 获取错误号PLAY_GetLastError

函数名称	DWORD PLAY_GetLastError(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	见 错误类型	
使用描述	获得当前错误的错误码。用户应该在调用某个函数失败时，调用此函数以获得错误的详细信息	

[返回目录](#)

4.22 其它

4.22.1 初始化DirectDraw表面**PLAY_InitDDraw ***

函数名称	BOOL PLAY_InitDDraw(HWND hWnd)	
参数说明	[in] HWND hWnd	应用程序主窗口的句柄
返回值	成功返回 TRUE，失败返回 FALSE	
注意	此接口无效	

[返回目录](#)

4.22.2 释放DirectDraw表面**PLAY_RealeseDDraw ***

函数名称	BOOL PLAY_RealeseDDraw()	
参数说明	无	无
返回值	成功返回 TRUE，失败返回 FALSE	
注意	此接口无效	

[返回目录](#)

4.22.3 设置丢B帧个数**PLAY_ThrowBFrameNum ***

函数名称	BOOL PLAY_ThrowBFrameNum(LONG nPort, DWORD nNum)	
参数说明	[in] LONG nPort	通道号
	[in] DWORD nNum	不解码B帧的帧数
返回值	成功返回 TRUE，失败返回 FALSE	
注意	此接口无效	

[返回目录](#)

4.22.4 刷新显示**PLAY_RefreshPlay**

函数名称	BOOL PLAY_RefreshPlay(LONG nPort)	
参数说明	[in] LONG nPort	通道号
返回值	成功返回 TRUE，不成功返回 FALSE	
使用描述	刷新显示。当用户暂停时如果刷新了窗口，则窗口中的图像因为刷新而消失，此时调用这个接口可以重新把图像显示出来。只有在暂停和单帧播放时才会执行，其它情况会直接返回	

[返回目录](#)