

# 设备(解码器)

## 网络 SDK 编程指南

### V5.2

# 声 明

非常感谢您购买我公司的产品，如果您有什么疑问或需要请随时联系我们。

- 我们已尽量保证手册内容的完整性与准确性，但也不免出现技术上不准确、与产品功能及操作不相符或印刷错误等情况，如有任何疑问或争议，请以我司最终解释为准。
- 产品和手册将实时进行更新，恕不另行通知。
- 本手册中内容仅为用户提供参考指导作用，请以 SDK 实际内容为准。

# 目 录

声 明 .....	I
目 录 .....	II
1 SDK 简介 .....	1
2 版本更新 .....	4
3 函数调用顺序 .....	7
3.1 解码器接入调用流程 .....	7
3.2 主动解码模块流程 .....	8
3.2.1 解码实时流 .....	8
3.2.2 远程文件回放 .....	9
3.3 被动解码模块流程 .....	10
3.4 电视墙相关配置 .....	11
3.4.1 DS_64XXHD_S .....	11
3.4.2 DS64XXHD_T、DS63XXD_T .....	12
4 函数调用实例 .....	14
4.1 主动解码模块的示例代码 .....	14
4.1.1 实时流解码 .....	14
4.1.2 远程回放解码 .....	19
4.2 被动解码模块的示例代码 .....	24
5 函数说明 .....	27
5.1 SDK 初始化 .....	27
5.1.1 初始化 SDK NET_DVR_Init .....	27
5.1.2 释放 SDK 资源 NET_DVR_Cleanup .....	27
5.2 SDK 本地功能 .....	27
SDK 本地参数配置 .....	27
5.2.1 获取 SDK 本地参数 NET_DVR_GetSDKLocalCfg .....	27
5.2.2 设置 SDK 本地参数 NET_DVR_SetSDKLocalCfg .....	28
连接和接收超时时间及重连设置 .....	28
5.2.3 设置网络连接超时时间和连接尝试次数 NET_DVR_SetConnectTime .....	28
5.2.4 设置重连功能 NET_DVR_SetReconnect .....	29
5.2.5 设置接收超时时间 NET_DVR_SetRecvTimeOut .....	29
多网卡绑定 .....	29
5.2.6 获取所有 IP，用于支持多网卡接口 NET_DVR_GetLocalIP .....	29
5.2.7 设置 IP 绑定 NET_DVR_SetValidIP .....	29
SDK 版本、状态和能力 .....	30
5.2.8 获取 SDK 的版本号和 build 信息 NET_DVR_GetSDKBuildVersion .....	30
5.2.9 获取当前 SDK 的状态信息 NET_DVR_GetSDKState .....	30
5.2.10 获取当前 SDK 的功能信息 NET_DVR_GetSDKAbility .....	30
SDK 启用写日志 .....	30
5.2.11 启用写日志文件 NET_DVR_SetLogToFile .....	30
异常消息回调 .....	31
5.2.12 注册接收异常、重连等消息的窗口句柄或回调函数 NET_DVR_SetExceptionCallBack_V30 .....	31

获取错误信息 .....	33
5.2.13    返回最后操作的错误码 NET_DVR_GetLastError .....	33
5.2.14    返回最后操作的错误码信息 NET_DVR_GetErrorMsg .....	33
5.3    用户注册 .....	33
5.3.1    激活设备 NET_DVR_ActivateDevice .....	33
5.3.2    通过解析服务器, 获取设备的动态 IP 地址和端口号 NET_DVR_GetDVRIPByResolveSvr_EX .....	33
5.3.3    用户注册设备 NET_DVR_Login_V40 .....	34
5.3.4    用户注销 NET_DVR_Logout .....	34
5.4    获取设备能力集 .....	35
5.4.1    获取设备能力集 NET_DVR_GetDeviceAbility .....	35
5.5    显示通道配置和控制 .....	36
5.5.1    获取显示通道信息 NET_DVR_MatrixGetDisplayCfg_V41 .....	36
5.5.2    显示通道配置 NET_DVR_MatrixSetDisplayCfg_V41 .....	36
5.5.3    显示通道控制 NET_DVR_MatrixDiaplayControl .....	36
5.6    参数配置 .....	37
5.6.1    获取设备的配置信息 NET_DVR_GetDVRConfig .....	37
5.6.2    设置设备的配置信息 NET_DVR_SetDVRConfig .....	38
5.6.3    批量获取配置信息 NET_DVR_GetDeviceConfig .....	39
5.6.4    批量设置配置信息 NET_DVR_SetDeviceConfig .....	40
5.7    解码通道相关 .....	41
5.7.1    获取解码通道配置信息 NET_DVR_MatrixGetDecChanCfg .....	41
5.7.2    配置解码通道 NET_DVR_MatrixSetDecChanCfg .....	41
5.7.3    获取当前解码通道状态 NET_DVR_MatrixGetDecChanStatus .....	41
5.7.4    获取解码通道开关 NET_DVR_MatrixGetDecChanEnable .....	42
5.7.5    设置解码通道开关 NET_DVR_MatrixSetDecChanEnable .....	42
5.8    主动解码 .....	42
5.8.1    启动动态解码 NET_DVR_MatrixStartDynamic_V41 .....	42
5.8.2    停止动态解码 NET_DVR_MatrixStopDynamic .....	43
5.8.3    获取轮巡解码通道 NET_DVR_MatrixGetLoopDecChanInfo_V41 .....	43
5.8.4    设置轮巡解码通道 NET_DVR_MatrixSetLoopDecChanInfo_V41 .....	43
5.8.5    获取解码通道轮巡开关 NET_DVR_MatrixGetLoopDecChanEnable .....	44
5.8.6    设置解码通道轮巡开关 NET_DVR_MatrixSetLoopDecChanEnable .....	44
5.8.7    获取所有解码通道轮巡开关 NET_DVR_MatrixGetLoopDecEnable .....	44
5.8.8    获取当前解码通道信息 NET_DVR_MatrixGetDecChanInfo_V41 .....	45
5.8.9    远程控制文件回放解码 NET_DVR_RemoteControl .....	45
5.8.10    远程回放文件解码配置 NET_DVR_MatrixSetRemotePlay .....	45
5.8.11    远程文件回放控制 NET_DVR_MatrixSetRemotePlayControl .....	46
5.8.12    获取回放状态 NET_DVR_MatrixGetRemotePlayStatus .....	46
5.9    被动解码 .....	47
5.9.1    启动被动解码 NET_DVR_MatrixStartPassiveDecode .....	47
5.9.2    向被动解码通道发送数据 NET_DVR_MatrixSendData .....	47
5.9.3    停止被动解码 NET_DVR_MatrixStopPassiveDecode .....	47
5.9.4    获取被动解码状态 NET_DVR_MatrixGetPassiveDecodeStatus .....	48
5.9.5    被动解码播放控制 NET_DVR_MatrixPassiveDecodeControl .....	48

5.10	LOGO 上传和显示控制 .....	48
5.10.1	LOGO 上传 NET_DVR_UploadLogo .....	48
5.10.2	LOGO 显示控制 NET_DVR_LogoSwitch .....	48
5.11	场景操作 .....	49
5.11.1	场景切换控制 NET_DVR_MatrixSceneControl .....	49
5.11.2	获取当前正在使用的场景模式 NET_DVR_MatrixGetCurrentSceneMode .....	49
5.12	远程控制 .....	50
5.12.1	远程控制 NET_DVR_RemoteControl .....	50
5.13	透明通道 .....	50
5.13.1	获取透明通道信息 NET_DVR_MatrixGetTranInfo_V30 .....	50
5.13.2	设置透明通道参数 NET_DVR_MatrixSetTranInfo_V30 .....	50
5.14	设备状态 .....	51
5.14.1	获取解码设备状态 NET_DVR_MatrixGetDeviceStatus_V41 .....	51
5.14.2	获取设备运行状态 NET_DVR_GetDeviceStatus .....	51
5.15	布防、撤防 .....	52
	设置报警等信息上传的回调函数 .....	52
5.15.1	注册报警信息回调函数 NET_DVR_SetDVRMessageCallBack_V31 .....	52
	布防撤防 .....	53
5.15.2	建立报警上传通道，获取报警等信息 NET_DVR_SetupAlarmChan_V41 .....	53
5.15.3	撤销报警上传通道 NET_DVR_CloseAlarmChan_V30 .....	53
5.16	监听报警 .....	54
5.16.1	启动监听，接收设备主动上传的报警等信息 NET_DVR_StartListen_V30 .....	54
5.16.2	停止监听（支持多线程）NET_DVR_StopListen_V30 .....	54
5.17	IPC 协议列表获取 .....	55
5.17.1	获取设备支持的 IPC 协议表 NET_DVR_GetIPCProtoList .....	55
5.18	设备维护管理 .....	55
	远程升级 .....	55
5.18.1	设置远程升级时网络环境 NET_DVR_SetNetworkEnvironment .....	55
5.18.2	远程升级 NET_DVR_Upgrade .....	55
5.18.3	获取远程升级的进度 NET_DVR_GetUpgradeProgress .....	56
5.18.4	获取远程升级的状态 NET_DVR_GetUpgradeState .....	56
5.18.5	关闭远程升级句柄，释放资源 NET_DVR_CloseUpgradeHandle .....	56
	恢复设备默认参数 .....	56
5.18.6	恢复设备默认参数 NET_DVR_RestoreConfig .....	56
5.18.7	完全恢复出厂默认参数 NET_DVR_RemoteControl .....	57
	导入/导出配置文件 .....	57
5.18.8	导出配置文件 NET_DVR_GetConfigFile_V30 .....	57
5.18.9	导出配置文件 NET_DVR_GetConfigFile .....	58
5.18.10	导入配置文件 NET_DVR_SetConfigFile_EX .....	58
5.18.11	导入配置文件 NET_DVR_SetConfigFile .....	58
5.19	关机和重启 .....	58
5.19.1	重启设备 NET_DVR_RebootDVR .....	58
5.19.2	关闭设备 NET_DVR_ShutDownDVR .....	59
6	错误代码及说明 .....	60

# 1 SDK 简介

设备网络 SDK 是基于设备私有网络通信协议开发的，为嵌入式网络硬盘录像机、视频服务器、网络摄像机、网络球机、解码器、多屏控制器、报警主机等产品服务的配套模块，用于远程访问和控制设备软件的二次开发。

本文档主要介绍解码器相关的功能，适用于但不仅限于以下产品型号：

DS-6300D(-JX/-T)、DS-6400HD(-JX/-T/-S)、DS-6500D(-T)系列解码器

**本文档仅介绍解码器支持的相关接口，结构体等更多内容参见《设备网络 SDK 使用手册.chm》。**

设备网络 SDK 包含网络通讯库、软解码库、硬解码库等功能组件，我们提供 Windows 和 Linux 两个版本的 SDK，各自所包含的组件如下：

表 1.1 Windows SDK 组件

网络通讯库	外部接口	HCNetSDK.h	头文件	
		HCNetSDK.lib	LIB 库文件	
		HCNetSDK.dll	DLL 库文件	
	核心组件	HCCore.lib	LIB 库文件	
		HCCore.dll	DLL 库文件	
组件库	设备配置核心组件	HCCoreDevCfg.dll	DLL 库文件	HCNetSDKCom 文件夹
	预览组件	HCPreview.lib	LIB 库文件	HCNetSDKCom 文件夹
		HCPreview.dll	DLL 库文件	HCNetSDKCom 文件夹
	回放组件	HCPlayBack.dll	DLL 库文件	HCNetSDKCom 文件夹
	语音组件	HCVoiceTalk.dll	DLL 库文件	HCNetSDKCom 文件夹
	报警组件	HCArm.lib	LIB 库文件	HCNetSDKCom 文件夹
		HCArm.dll	DLL 库文件	HCNetSDKCom 文件夹
	显示组件	HCDisplay.dll	DLL 库文件	HCNetSDKCom 文件夹
	行业应用管理配置组件	HCIndustry.dll	DLL 库文件	HCNetSDKCom 文件夹
	维护管理配置组件	HCGeneralCfgMgr.lib	LIB 库文件	HCNetSDKCom 文件夹
		HCGeneralCfgMgr.dll	DLL 库文件	HCNetSDKCom 文件夹
RTSP 通讯库		StreamTransClient.dll	DLL 库文件	HCNetSDKCom 文件夹
转封装库		SystemTransform.dll	DLL 库文件	HCNetSDKCom 文件夹
语音对讲库		AudioIntercom.dll	DLL 库文件	HCNetSDKCom 文件夹
		OpenAL32.dll	DLL 库文件	HCNetSDKCom 文件夹
字符转码库		libiconv2.dll	DLL 库文件	HCNetSDKCom 文件夹
模拟能力集		LocalXml.zip	XML 文件包	
软解码库		plaympeg4.h	头文件	

		PlayCtrl.lib	LIB 库文件	
		PlayCtrl.dll	DLL 库文件	
		AudioRender.dll	DLL 库文件	
		SuperRender.dll	DLL 库文件	
硬解码库		DsSdk.dll	DLL 库文件	HCNetSDKCom 文件夹

表 1.2 Linux SDK 组件

网络通讯库	外部接口	HCNetSDK.h	头文件	
		libhcnetsdk.so	SO 库文件	
	核心组件	libHCCore.so	SO 库文件	
组件库	设备配置核心组件	libHCCoreDevCfg.so	SO 库文件	HCNetSDKCom 文件夹
	预览组件	libHCPreview.so	SO 库文件	HCNetSDKCom 文件夹
	回放组件	libHCPlayBack.so	SO 库文件	HCNetSDKCom 文件夹
	语音组件	libHCVoiceTalk.so	SO 库文件	HCNetSDKCom 文件夹
	报警组件	libHCAAlarm.so	SO 库文件	HCNetSDKCom 文件夹
	显示组件	libHCDisplay.so	SO 库文件	HCNetSDKCom 文件夹
	行业应用管理配置组件	libHCIndustry.so	SO 库文件	HCNetSDKCom 文件夹
	维护管理配置组件	libHCGeneralCfgMgr.so	SO 库文件	HCNetSDKCom 文件夹
hpr 库		libhpr.so	SO 库文件	
RTSP 通讯库		libStreamTransClient.so	SO 库文件	HCNetSDKCom 文件夹
转封装库		libSystemTransform.so	SO 库文件	HCNetSDKCom 文件夹
字符转码库		libiconv2.so	SO 库文件	HCNetSDKCom 文件夹
软解码库		LinuxPlayM4.h	头文件	
		PlayM4.h	头文件	
		libMPCtrl.so	SO 库文件	
		libPlayCtrl.so	SO 库文件	

本版本的设备网络 SDK 开发包中包含以上各个组件，**HCNetSDK.dll、HCCore.dll 必须加载**（对于 Linux SDK，即 libhcnetsdk.so、libHCCore.so），其他组件，用户可以根据需要选择其中的一部分或者全部，以下将对各个组件在 SDK 中的作用和使用条件分别说明。

- **网络通讯库**：设备网络 SDK 的主体，主要用于网络客户端与各类产品之间的通讯交互，负责远程功能调控，远程参数配置及码流数据的获取和处理等。设备网络 SDK V5.0 针对产品应用业务进行细化，对之前版本的 SDK 的功能模块进行组件化，其中外部接口（HCNetSDK.dll）仍然保持和设备网络 SDK V4.x 版本保存一致(向下兼容)，其他单独的业务功能（预览、回放等）可以加载单独的模块组件，多个业务功能也可以组合使用。**更新 SDK 时，HCNetSDK.dll、HCCore.dll 以及 HCNetSDKCom 文件夹下的功能组件库文件都需要更新加载，且 HCNetSDKCom 文件夹名不能修改。**

- **hpr 库：**网络通讯库的依赖库，Linux SDK 使用时和网络通讯库同时加载。
- **RTSP 通讯库：**支持 RTSP 传输协议的网络库。当需要对支持 RTSP 协议的产品进行取流等操作时就必须加载该项组件。
- **转封装库：**库的功能可以分为两种：一种是将标准码流转换成采用我们公司封装格式的码流。当用户需要对支持 RTSP 协议的产品捕获采用本公司封装格式的码流数据时（即当设置 [NET\\_DVR\\_RealPlay\\_V40](#) 接口中的回调函数捕获数据或者调用 [NET\\_DVR\\_SetRealDataCallBack](#) 接口捕获数据时）必须加载该组件。另一种功能是将标准码流转换成其他格式的封装，如 3GPP、PS 等。例如，当用户需要对支持 RTSP 协议的产品实时捕获指定封装格式的码流数据（对应的 SDK 接口为 [NET\\_DVR\\_SaveRealData](#)）时必须加载该项组件。
- **语音对讲库：**用于语音对讲时通过声卡采集数据并按照指定的编码格式编码码流或者解码播放音频码流数据（不带封装格式的码流数据）。V4.2.2.5 及以前版本 SDK 均采用 windows API 实现相关功能。之后版本默认使用语音对讲库的方式，通过接口 [NET\\_DVR\\_SetSDKLocalCfg](#) 可以选择之前的 windows API 模式。OpenAL32.dll 为依赖库，语音对讲库模式下必须加载。**Windows64 位或者 Linux 系统下无语音对讲功能。CVR 暂不支持语音对讲功能。**
- **字符转换库：**电脑字符集和设备字符集不一致时，SDK 内部需要进行字符编码转换，SDK 默认使用 libiconv 库进行类型转换。如果用户不想使用 libiconv 编码库，可以调用 [NET\\_DVR\\_SetSDKLocalCfg](#) (类型: [NET\\_SDK\\_LOCAL\\_CFG\\_TYPE\\_BYTE\\_ENCODE](#))设置字符转码回调函数，将用户自己的字符编码接口告知 SDK，然后 SDK 将使用用户提供的字符编码接口进行字符串处理。
- **模拟能力集：**如果需要获取设备能力集（[NET\\_DVR\\_GetDeviceAbility](#)），建议调用 [NET\\_DVR\\_SetSDKLocalCfg](#) 启用模拟能力集，此时需要加载 LocalXml.zip（要求和网络通讯库放在同一个目录下）。
- **软解码库：**主要用于对实时码流数据进行解码显示（实现预览功能）和对录像文件进行回放解码等。用户如果需要在 SDK 内部进行对实时流和录像码流播放显示时（即 [NET\\_DVR\\_RealPlay\\_V40](#) 接口的第二个结构体参数的播放句柄设置成有效句柄时）必须加载该组件，而如果用户仅需要用网络通讯库捕获到数据后再外部自行处理就不需要加载该组件，这种情况下用户在外部分自行解码将更灵活，可参见软解码库函数说明《播放器 SDK 编程指南》。**Linux 64 位系统不支持软解码功能，预览、回放等窗口句柄传空，仅支持只取流不解码。**
- **硬解码库：**需在配备硬解码卡（MD 卡）的前提下使用，通过解码卡的解码与输出功能实现实时流的解码显示及向监视器上矩阵输出的功能。**Windows64 位或者 Linux 系统下无该硬解码库。对于解码器设备，不需要使用该库文件。**



## 2 版本更新

### Version 5.1.3.5 (build20150701)

- DS-6400HD-T V3.5.0
- 新增窗口漫游模式开关配置功能（对应接口：[NET\\_DVR\\_GetDVRConfig](#) 和 [NET\\_DVR\\_SetDVRConfig](#)）：  
命令：NET\_DVR\_GET\_WIN\_ROAM\_SWITCH\_CFG、NET\_DVR\_SET\_WIN\_ROAM\_SWITCH\_CFG。
- NET\_DVR\_MATRIX\_DECCHAN\_CONTROL(窗口参数配置)使用 5 个保留字节新增参数：byEnableVcaDec (是否启用智能解码)、dwAllCtrlType(所有子窗口一起操作的类型)。
- 设备能力集扩展（对应接口：NET\_DVR\_GetDeviceAbility）：
  - 1) 电视墙能力集(WALL\_ABILITY)中新增节点：<VcaDecode>(是否支持智能解码)。
  - 2) 解码器能力集(DECODER\_ABILITY)中新增节点：<SupportWinRoamSwitch>(是否支持窗口漫游)。

### Version 5.0.2.10 (build20141113)

- DS-6400HD-T V3.0.0
- 新增解码器智能报警配置功能（对应接口：[NET\\_DVR\\_GetDVRConfig](#) 和 [NET\\_DVR\\_SetDVRConfig](#)）：  
命令：NET\_DVR\_GET\_DEC\_VCA\_CFG、NET\_DVR\_SET\_DEC\_VCA\_CFG。
- 新增智能解码报警信息上传功能（对应接口：[NET\\_DVR\\_SetDVRMessageCallBack\\_V31](#) 和 [NET\\_DVR\\_StartListen\\_V30](#)）：  
类型：COMM\_ALARM\_DEC\_VCA。
- NET\_DVR\_DEV\_WORK\_MODE(设备工作模式)使用 1 个保留字节新增参数：byEnableVcaDec (是否启用智能解码)。
- 设备软硬件能力集(DEVICE\_SOFTWARE\_ABILITY)中节点<NeedReboot>(是否需要重启)新增子节点：<VcaDecModeChange>(开启智能解码模式是否需要重启)。
- 解码器能力集(DECODER\_ABILITY)新增节点：<VcaDec>(智能解码能力)。

### Version 4.3.0.4 (build20140610)

- DS-6400HD-T V2.4.0
- 新增获取解码相关状态功能（对应接口：[NET\\_DVR\\_GetDeviceStatus](#)）：  
NET\_DVR\_GET\_DEC\_CHAN\_STATUS(获取解码通道状态)、NET\_DVR\_GET\_DISP\_CHAN\_STATUS(获取显示通道状态)
- 新增状态获取功能（对应接口：[NET\\_DVR\\_GetDeviceStatus](#)）：  
NET\_DVR\_GET\_ALARM\_IN\_STATUS(获取报警输入状态)、NET\_DVR\_GET\_ALARM\_OUT\_STATUS(获取报警输出状态)、NET\_DVR\_GET\_AUDIO\_CHAN\_STATUS(获取语音对讲状态)
- NET\_DVR\_DEV\_CHAN\_INFO\_EX(前端编码设备信息)使用 4 个保留字节新增参数：dwChannel(通道号)，byChanType(通道类型)新增取值类型：3- 本地输入源。
- NET\_DVR\_MATRIX\_CHAN\_STATUS(解码通道状态)使用 4 个保留字节新增参数：dwDecChan(解码通道号)。
- NET\_DVR\_MATRIX\_VOUTCFG(显示通道配置)使用 4 个保留字节新增参数：dwDispChanNum(显示通道号)。
- NET\_DVR\_DISP\_CHAN\_STATUS\_V41(显示通道状态)使用 4 个保留字节新增参数：dwDispChan(显示通道号)。
- 解码器能力集(DECODER\_ABILITY)新增节点<SupportLocalInputDec>(支持本地输入源)。

### Version 4.2.6.7 (build20131031)

- DS64XXHD\_T、DS63XXD\_T V2.3.0
- 新增远程文件回放解码控制功能（对应接口 [NET\\_DVR\\_RemoteControl](#)）：  
控制命令：NET\_DVR\_DEC\_PLAY\_REMOTE\_FILE，向下兼容 NET\_DVR\_MatrixSetRemotePlay。
- NET\_DVR\_DEC\_STREAM\_MODE(取流配置联合体)：  
新增支持零通道取流和通过流 ID 从前端设备取流的功能：NET\_DVR\_DEC\_STREAM\_DEV 扩展为 NET\_DVR\_DEC\_STREAM\_DEV\_EX；  
新增通过 DDNS 域名方式连接设备取流方式：联合体中新增 NET\_DVR\_DEC\_DDNS\_DEV 结构体。
- NET\_DVR\_PU\_STREAM\_CFG\_V41(动态解码取流参数)、NET\_DVR\_MATRIX\_CHAN\_INFO\_V41(动态解码轮巡通道信息)、NET\_DVR\_MATRIX\_DEC\_CHAN\_INFO\_V41(解码通道信息)的参数 byStreamMode 新增取值：3-通过动态域名解析向设备取流。
- NET\_DVR\_DEVICEINFO\_V30(设备参数)中 bySupport3 新增取值：bySupport3&0x20 表示是否支持通过 DDNS 域名解析取流。
- 解码器 xml 能力集 DECODER\_ABILITY（对应接口 [NET\\_DVR\\_GetDeviceAbility](#)）新增节点：  
<mediaServerVersion>、<heartBeatType>、<SupportStreamIdDec>、<SupportDdnsDec>、<SupportZeroChanDec>、<DeviceDisplayMode>。

### Version 4.2.5.6 (build20130808)

- DS\_64XXHD\_S V1.5
- 新增远程控制功能（对应接口 [NET\\_DVR\\_RemoteControl](#)）：  
命令：NET\_DVR\_SWITCH\_WIN\_TOP(窗口置顶)、NET\_DVR\_SWITCH\_WIN\_BOTTOM(窗口置底)。

### Version 4.2.1.4 (2012.12.11)

- DS63XXD\_T、DS64XXD\_T、DS65XXD V2.0.0
- 新增解码器扩展接口（向下兼容 V30 的接口）：  
[NET\\_DVR\\_MatrixStartDynamic\\_V41](#)  
[NET\\_DVR\\_MatrixGetLoopDecChanInfo\\_V41](#)  
[NET\\_DVR\\_MatrixSetLoopDecChanInfo\\_V41](#)  
[NET\\_DVR\\_MatrixGetDecChanInfo\\_V41](#)。
- NET\_DVR\_MATRIX\_PASSIVEMODE(被动解码参数)使用 2 个保留字节增加参数 byReconnectFlag、byRequestType。
- NET\_DVR\_DEVICEINFO\_V30(设备参数)使用 1 个保留字节增加参数 bySupport2。

### Version 4.1.10.2 (2012.10.31)

- DS-64xxHD-S V1.0，新增设备类型 DS\_64XXHD\_S
- 新增配置功能（对应接口 [NET\\_DVR\\_GetDeviceConfig](#) 和 [NET\\_DVR\\_SetDeviceConfig](#)）：  
电视墙中屏幕参数配置(NET\_DVR\_SINGLEWALLPARAM)、电视墙中窗口参数配置(NET\_DVR\_WALLWINCFG)、电视墙中场景参数配置(NET\_DVR\_WALLSCENECFG)。
- 新增场景切换控制和状态获取接口：  
[NET\\_DVR\\_MatrixSceneControl](#)  
[NET\\_DVR\\_MatrixGetCurrentSceneMode](#)。
- 新增获取窗口解码状态的接口：  
[NET\\_DVR\\_GetDeviceStatus](#)。
- 新增能力集类型（对应接口 [NET\\_DVR\\_GetDeviceAbility](#)）：WALL\_ABILITY（宏值：0x212）。

- NET\_DVR\_MatrixStartDynamic\_V30 和 NET\_DVR\_MatrixStopDynamic 的参数 dwDecChanNum 新增支持传入窗口号。

### Version 4.1.8.4 (2012.10.12)

- 新增支持 DS-65xxD 解码器，设备类型新增 DS\_65XXD
- 新增解码器自动重启配置功能（对应接口 [NET\\_DVR\\_GetDVRConfig](#) 和 [NET\\_DVR\\_SetDVRConfig](#)）：  
NET\_DVR\_AUTO\_REBOOT\_CFG。
- NET\_DVR\_MATRIX\_ABILITY\_V41 使用 1 个保留字节增加参数 bySupportAutoReboot。

### Version 4.1.8 (2012.09.19)

- 新增支持 63-T 标清解码器，设备类型新增 DS63XXD\_T
- 增加错误号 99（NET\_DVR\_DEC\_CHAN\_REBIND）。

### Version 4.1.0 (2012.01.12)

- DS-64XX-T 系列，设备类型新增 DS64XXHD\_T
- 新增解码器扩展能力集（对应接口 [NET\\_DVR\\_GetDeviceAbility](#)）：  
能力集类型：MATRIXDECODER\_ABILITY\_V41（宏值：0x260），能力参数结构体：  
NET\_DVR\_MATRIX\_ABILITY\_V41。
- 新增解码器显示通道配置扩展接口（向下兼容 V30 的接口）：  
[NET\\_DVR\\_MatrixGetDisplayCfg\\_V41](#)、[NET\\_DVR\\_MatrixSetDisplayCfg\\_V41](#)。
- 新增解码器设备状态扩展接口（向下兼容 V30 的接口）：  
[NET\\_DVR\\_MatrixGetDeviceStatus\\_V41](#)。
- NET\_DVR\_MATRIX\_DECCHAN\_CONTROL 使用 1 个保留字节新增参数：byDecodeDelay(解码延时)。
- 新增 64T 高清解码器大屏拼接功能（对应接口 [NET\\_DVR\\_GetDVRConfig](#) 和 [NET\\_DVR\\_SetDVRConfig](#)）：  
NET\_DVR\_BIGSCREENCFG。

### Version 3.5.0 (2010.02.09)

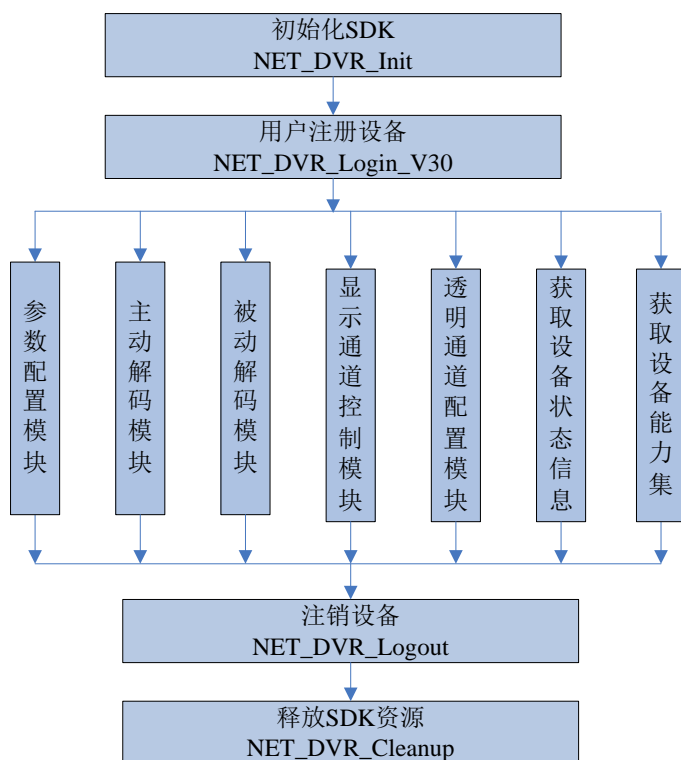
- 多路解码器 DS-630X-D
- 新增接口功能：  
NET\_DVR\_MatrixStartDynamic\_V30、NET\_DVR\_MatrixSetLoopDecChanInfo\_V30、  
NET\_DVR\_MatrixGetLoopDecChanInfo\_V30、NET\_DVR\_MatrixGetDecChanInfo\_V30、  
[NET\\_DVR\\_MatrixSetTranInfo\\_V30](#)、[NET\\_DVR\\_MatrixGetTranInfo\\_V30](#)、NET\_DVR\_MatrixGetDisplayCfg、  
NET\_DVR\_MatrixSetDisplayCfg、[NET\\_DVR\\_MatrixStartPassiveDecode](#)、[NET\\_DVR\\_MatrixSendData](#)、  
[NET\\_DVR\\_MatrixStopPassiveDecode](#)、[NET\\_DVR\\_UploadLogo](#)、[NET\\_DVR\\_LogoSwitch](#)、  
NET\_DVR\_MatrixGetDeviceStatus、[NET\\_DVR\\_MatrixDiaplayControl](#)、[NET\\_DVR\\_MatrixGetDecChanCfg](#)、  
[NET\\_DVR\\_MatrixSetDecChanCfg](#)、[NET\\_DVR\\_MatrixGetPassiveDecodeStatus](#)、  
[NET\\_DVR\\_MatrixGetVideoStandard](#)、[NET\\_DVR\\_MatrixSetVideoStandard](#)、  
[NET\\_DVR\\_MatrixPassiveDecodeControl](#)。

## 3 函数调用顺序

注：流程图中虚线框的部分是可选的，不会影响其他流程和模块的功能使用。

### 3.1 解码器接入调用流程

图 3.1 SDK 基本流程



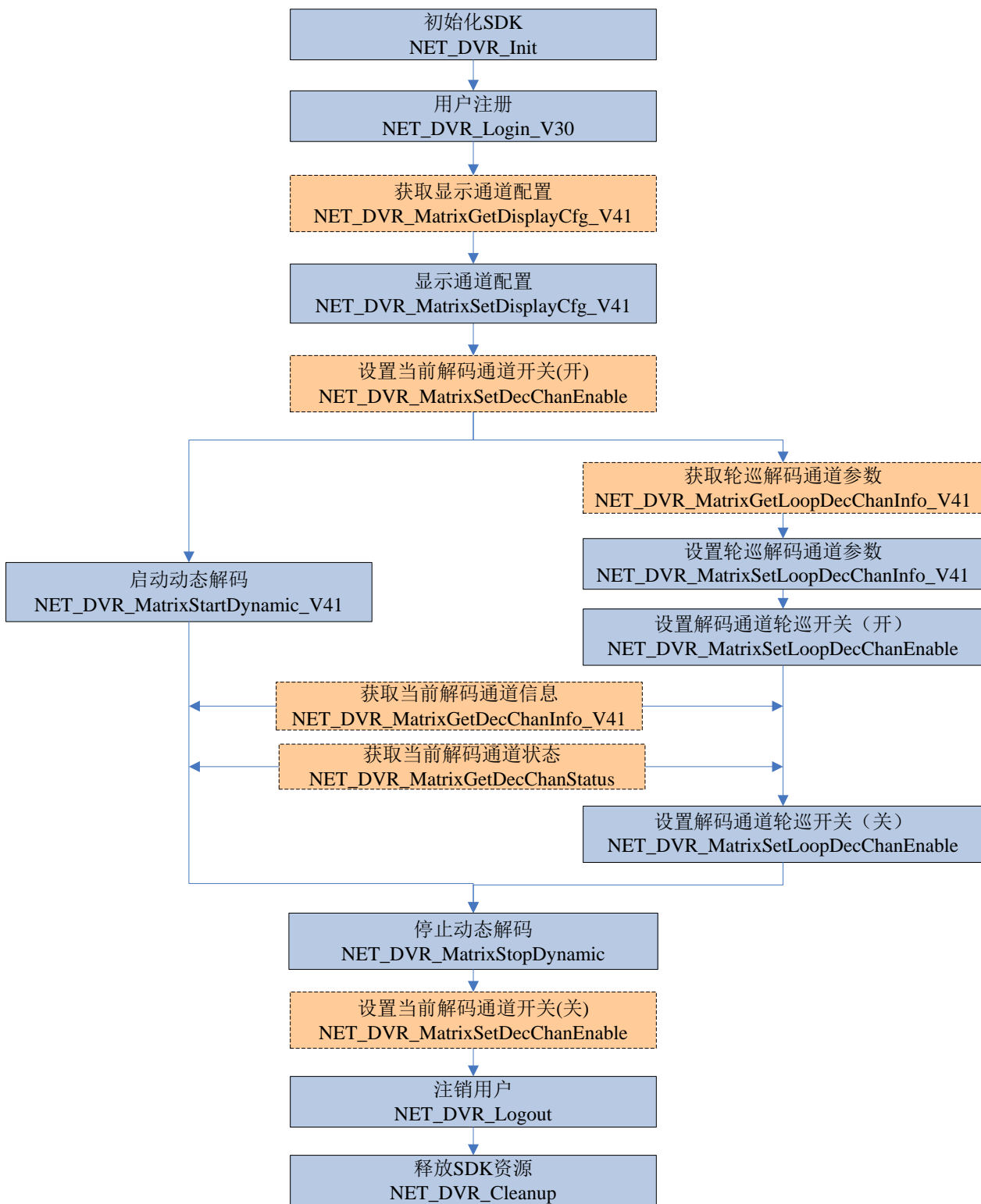
多路解码器的功能模块包括显示通道控制、参数配置、主动解码、被动解码、透明通道配置、获取设备状态信息和获取设备能力集模块。该模块的所有功能都需要先进行用户注册，[NET\\_DVR\\_Login\\_V30](#) 接口返回的用户 ID 号作为其他功能接口的参数。

- 显示通道控制模块：显示通道各项参数的配置，音频开启关闭控制和子窗口缩放控制等。详细接口可见下文的接口定义中的“多路解码器”的[“显示通道配置和控制”](#)部分。
- 参数配置模块：主要完成对多路解码器的基本参数的配置功能，相关接口：[NET\\_DVR\\_GetDVRConfig](#)、[NET\\_DVR\\_SetDVRConfig](#)和 [NET\\_DVR\\_GetDeviceConfig](#)、[NET\\_DVR\\_SetDeviceConfig](#)，其中可配置的参数可见下文的接口定义中的“多路解码器”的[“参数配置”](#)部分。
- 主动解码模块：配置和获取多路解码器主动解码参数，对主动解码进行各种控制，包括起停动态解码；起停轮巡解码；按文件及按时间回放的控制；以及主动解码状态的获取、主动解码起停的总体控制；上传 LOGO 等功能。详细内容请参见[主动解码模块流程](#)。
- 被动解码模块：被动解码通道的启动解码、发送数据和停止解码。详细内容请参见[被动解码模块流程](#)。
- 透明通道配置模块：配置透明通道的相关参数。详细接口可见下文的接口定义中的“多路解码器”的[“透明通道配置”](#)部分。
- 获取设备状态信息：通过 [NET\\_DVR\\_MatrixGetDeviceStatus\\_V41](#) 接口获取设备通道的解码、报警输入、报警输出和语音对讲等状态信息。
- 获取设备能力集：通过 [NET\\_DVR\\_GetDeviceAbility](#) 获取解码器的显示和解码通道能力等相关信息。

## 3.2 主动解码模块流程

### 3.2.1 解码实时流

图 3.2 解码实时流



- 登录解码器后需要先配置解码器显示通道参数，设置显示通道关联的解码通道，否则无法正常启动解

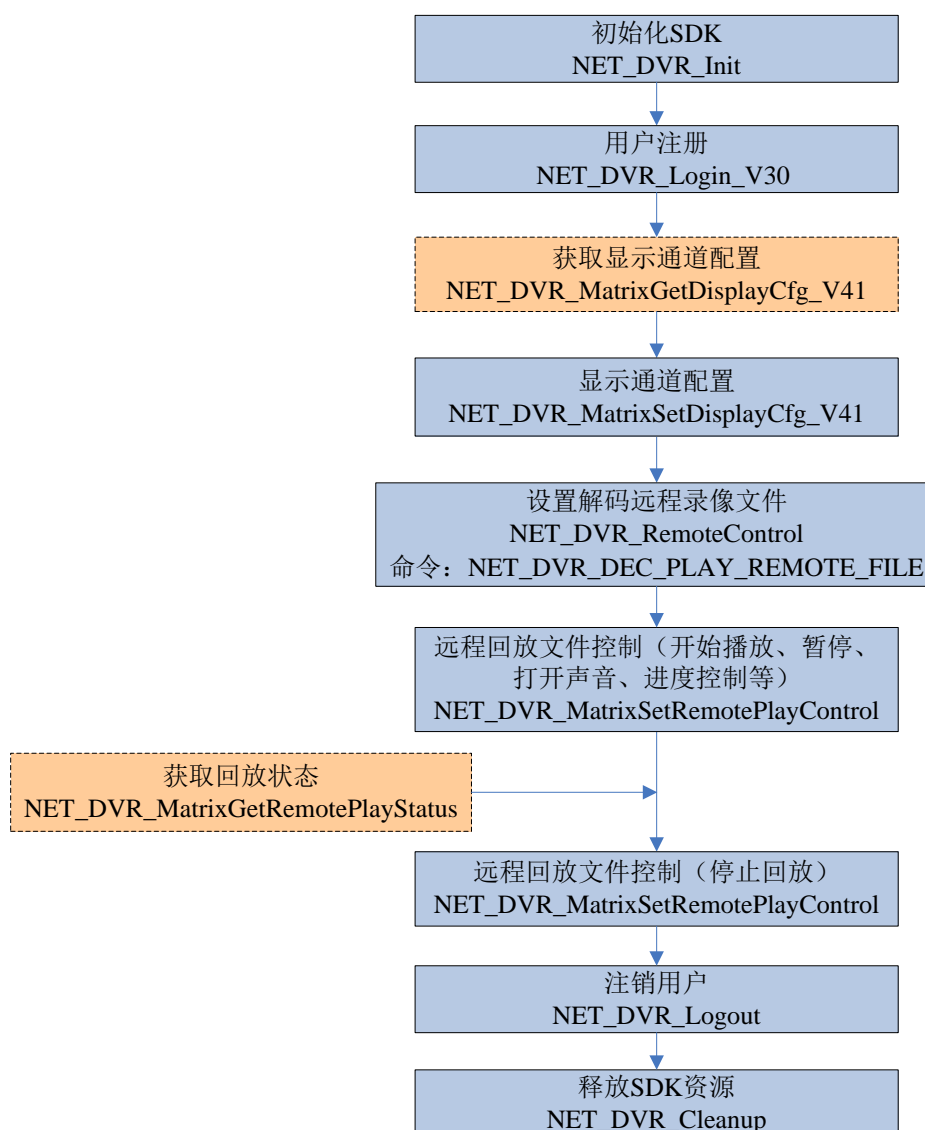
码。相关接口：[NET\\_DVR\\_MatrixGetDisplayCfg\\_V41](#)、[NET\\_DVR\\_MatrixSetDisplayCfg\\_V41](#)。

- 调用 [NET\\_DVR\\_MatrixStartDynamic\\_V41](#) 即启动主动解码，解码器从设备直接取流获取通过流媒体取流进行解码并且上墙显示，取流方式、通道或者流 ID 等都在该接口中指定。如果显示通道配置中的分辨率和制式没有指定，则必须指定一个有效的分辨率或制式，否则无法正常动态解码。
- 解码器直接轮巡解码，通过 [NET\\_DVR\\_MatrixSetLoopDecChanInfo\\_V41](#) 设置轮巡组，然后调用 [NET\\_DVR\\_MatrixSetLoopDecChanEnable](#) 启动轮巡。

[调用示例代码](#)

### 3.2.2 远程文件回放

图 3.3 解码远程文件

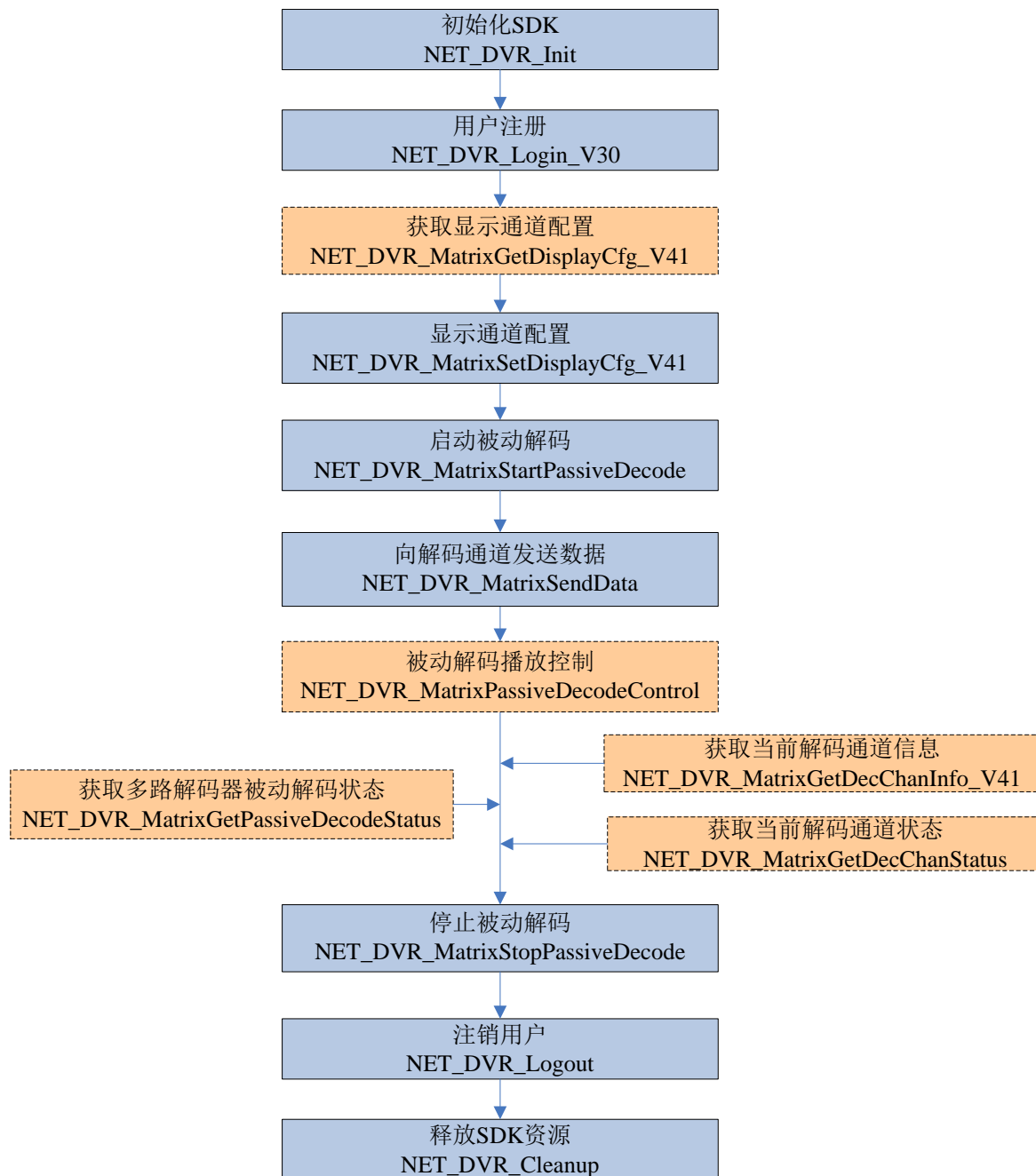


- 登录解码器后需要先配置解码器显示通道参数，设置显示通道关联的解码通道，否则无法正常启动解码。相关接口：[NET\\_DVR\\_MatrixGetDisplayCfg\\_V41](#)、[NET\\_DVR\\_MatrixSetDisplayCfg\\_V41](#)。
- 设置解码远程录像文件：首先通过 [NET\\_DVR\\_RemoteControl](#)（命令：NET\_DVR\_DEC\_PLAY\_REMOTE\_FILE）设置按时间或者按文件名回放远程设备的录像文件，然后调用 [NET\\_DVR\\_MatrixSetRemotePlayControl](#) 启动解码。

[调用示例代码](#)

### 3.3 被动解码模块流程

图 3.4 被动解码



- 登录解码器后需要先配置解码器显示通道参数，设置显示通道关联的解码通道，否则无法正常启动解码。相关接口：[NET\\_DVR\\_MatrixGetDisplayCfg\\_V41](#)、[NET\\_DVR\\_MatrixSetDisplayCfg\\_V41](#)。
- [NET\\_DVR\\_MatrixStartPassiveDecode](#) 启动被动解码后，通过接口 [NET\\_DVR\\_MatrixSendData](#) 向解码通道发送数据。数据可以从远程设备获取的实时流数据也可以是从录像文件读取的数据，每次发送的数据不能大于 30K 字节。
- 解码控制：暂停、快放、慢放、开启或者关闭音频、清空缓冲区等，相关接口：[NET\\_DVR\\_MatrixPassiveDecodeControl](#)。

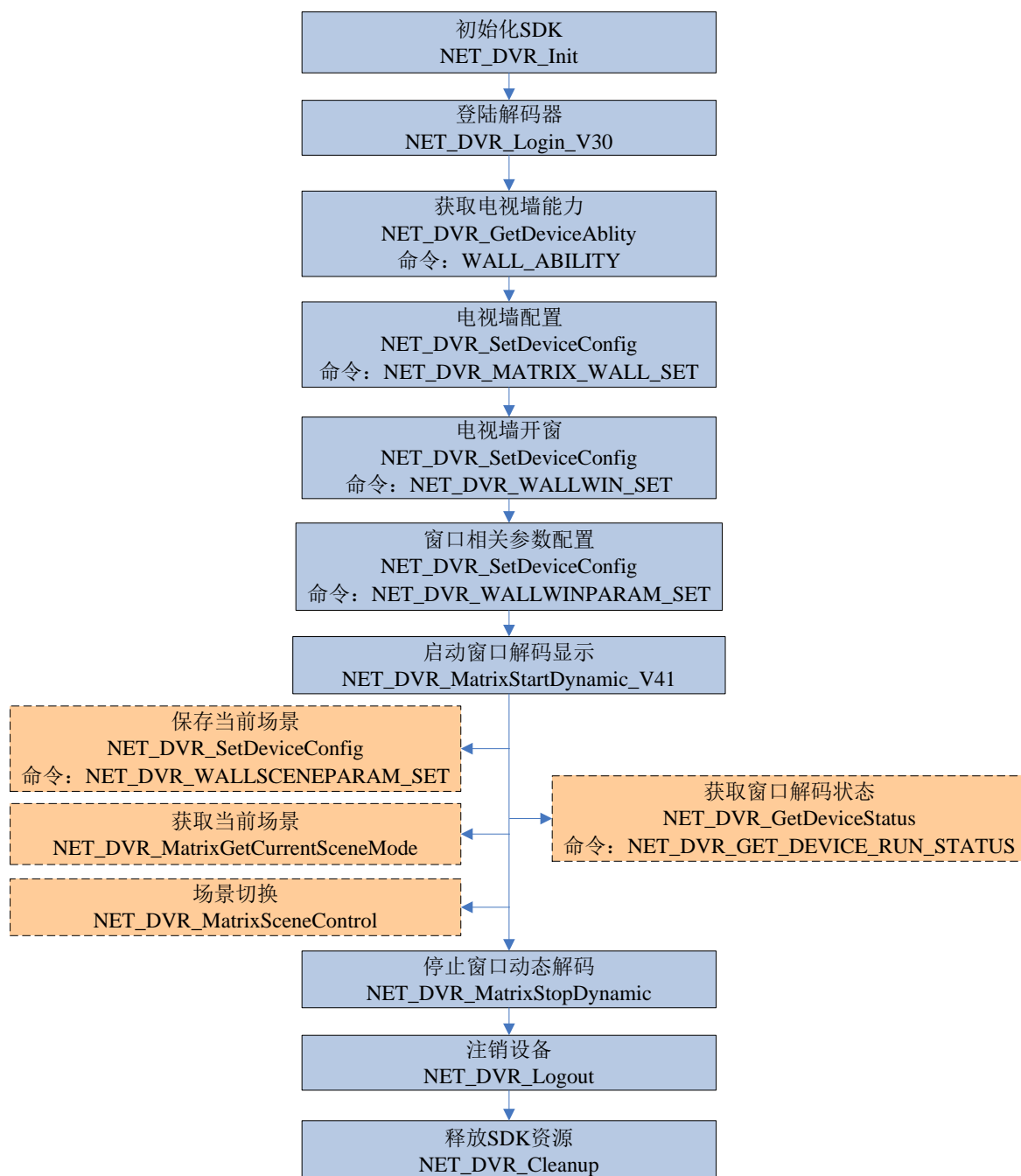
[调用示例代码](#)



## 3.4 电视墙相关配置

### 3.4.1 DS\_64XXHD\_S

图 3.5 HD-S 电视墙配置



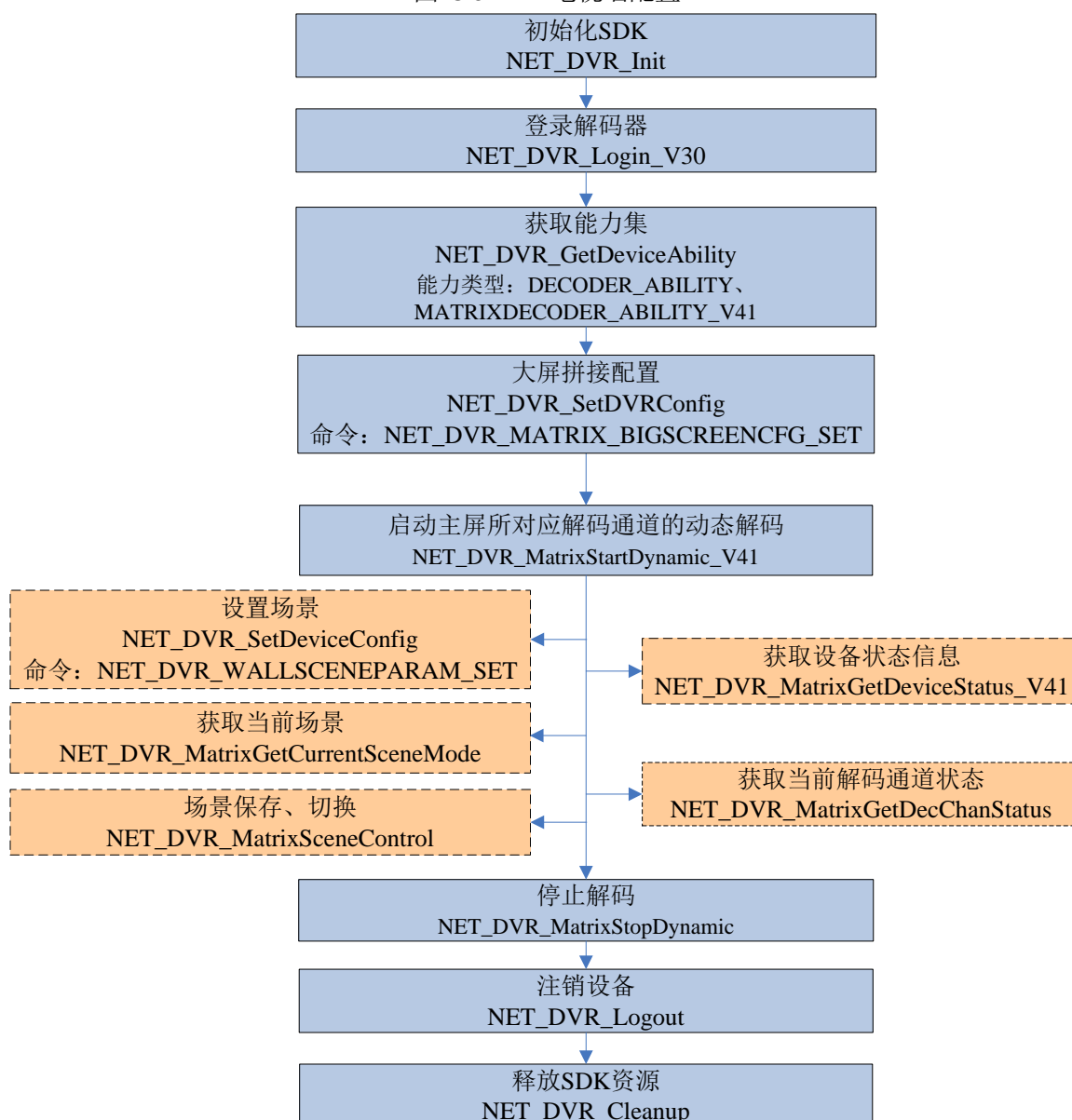
- 通过 [NET\\_DVR\\_GetDeviceAbility](#) 获取视频综合平台的电视墙能力（WALL\_ABILITY），包括电视墙屏幕基准值、支持的窗口数（窗口编号）、窗口支持的分屏数、输出口号等。
- 电视墙屏幕参数配置：[NET\\_DVR\\_SetDeviceConfig](#)（命令：NET\_DVR\_MATRIX\_WALL\_SET）设置屏幕参数（NET\_DVR\_SINGLEWALLPARAM）。



- 开窗：NET\_DVR\_SetDeviceConfig（命令：NET\_DVR\_WALLWIN\_SET）设置电视墙窗口参数，实现开窗。
- 开窗后，需要进行窗口相关参数配置：NET\_DVR\_SetDeviceConfig（命令：NET\_DVR\_WALLWINPARAM\_SET）设置电视墙窗口参数（NET\_DVR\_WALLWINPARAM），窗口透明度、分屏模式等。
- 完成参数配置后即可启动窗口解码显示，相关接口：NET\_DVR\_MatrixStartDynamic\_V41
- 场景操作：通过 NET\_DVR\_SetDeviceConfig（命令：NET\_DVR\_WALLSCENEPARAM\_SET）可以设置场景，NET\_DVR\_MatrixGetCurrentSceneMode 可以获取当前正在使用的场景模式。调用接口 NET\_DVR\_MatrixSceneControl 可以保存或者切换到其他已保存的场景。
- 状态获取：通过 NET\_DVR\_GetDeviceConfig（命令：NET\_DVR\_GET\_DEVICE\_RUN\_STATUS）可以获取各窗口解码状态（NET\_DVR\_WALL\_WIN\_STATUS）。

### 3.4.2 DS64XXHD\_T、DS63XXD\_T

图 3.6 HD-T 电视墙配置



- 登录解码器后调用接口 NET\_DVR\_GetDeviceAbility（能力集类型：MATRIXDECODER\_ABILITY\_V41）获取解码器能力，包括设备显示通道和解码通道信息、设备支持的大屏拼接个数、拼接的屏幕个数等。

- 通过命令 `NET_DVR_MATRIX_BIGSCREENCFG_SET`（对应接口：[NET\\_DVR\\_SetDVRConfig](#)）设置大屏拼接，包括拼接的主从屏及其所对应的显示通道号、制式、分辨率等。
- 大屏拼接参数（`NET_DVR_BIGSCREENCFG`）中主屏槽位号（`byMainDecodeSystem`）即关联的解码通道号，完成大屏拼接后即可调用 [NET\\_DVR\\_MatrixStartDynamic\\_V41](#) 启动解码。
- 场景操作
  - 通过 [NET\\_DVR\\_MatrixGetCurrentSceneMode](#) 可以获取当前正在使用的场景模式
  - 通过接口 [NET\\_DVR\\_MatrixSceneControl](#) 保存场景，或者切换场景。
  - 进行上述大屏拼接和开窗操作，然后保存当前配置信息为场景，通过命令 `NET_DVR_WALLSCENEPARAM_SET`（对应接口：[NET\\_DVR\\_SetDeviceConfig](#)）实现场景设置。
- 建议在每次设置某类参数之前，先调用获取参数的接口（[NET\\_DVR\\_GetDVRConfig](#) 或 [NET\\_DVR\\_GetDeviceConfig](#)）得到完整的参数结构，修改需要更改的参数，作为设置参数接口中的输入参数，最后调用设置参数接口（[NET\\_DVR\\_SetDVRConfig](#) 或 [NET\\_DVR\\_SetDeviceConfig](#)）。

## 4 函数调用实例

### 4.1 主动解码模块的示例代码

#### 4.1.1 实时流解码

[相关模块流程图](#)

```
#include <stdio.h>
#include <string.h>
#include <iostream>
#include "Windows.h"
#include "HCNetSDK.h"
using namespace std;

void main()
{
    int i=0,j=0;
    BYTE byDispMode;

    //初始化 SDK
    NET_DVR_Init();

    //注册设备
    NET_DVR_DEVICEINFO_V30 struDeviceInfo;
    memset(&struDeviceInfo, 0, sizeof(NET_DVR_DEVICEINFO_V30)); //存放设备参数的结构体
    LONG lUserID = NET_DVR_Login_V30("172.6.22.178", 8000, "admin", "12345", &struDeviceInfo);
    if (lUserID < 0)
    {
        printf("Login error, %d\n", NET_DVR_GetLastError());
        NET_DVR_Cleanup();
        return;
    }

    //获取解码器能力集
    NET_DVR_MATRIX_ABILITY_V41 struDecoderCapability;
    if (!NET_DVR_GetDeviceAbility(lUserID, MATRIXDECODER_ABILITY_V41, NULL, 0, (char *)&struDecoderCapability,
    sizeof(struDecoderCapability)))
    {
        printf("NET_DVR_GetDeviceAbility error, %d\n", NET_DVR_GetLastError());
        NET_DVR_Logout(lUserID);
        NET_DVR_Cleanup();
    }
}
```

```

        return;
    }
    else
    {
        printf("Starting number of decoding channel: %d\n",struDecoderCapability.byStartChan);
        printf("The total number of VGA output: %d\n",struDecoderCapability.struVgaInfo.byChanNums);
        printf("Starting number of VGA output: %d\n",struDecoderCapability.struVgaInfo.byStartChan);
        for (i=0;i<MAX_DISPNUM_V41;i++)
        {
            if (struDecoderCapability.struDispMode[i].byDispChanType==1&&struDecoderCapability.struDispMode[i].byDispChanSeq==1)
            {
                //获取 VGA 1 输出支持的画面分割模式
                for (j=0;j<MAX_WINDOWS_NUM;j++)
                {
                    if (struDecoderCapability.struDispMode[i].byDispMode[j]>0)
                    {
                        byDispMode=struDecoderCapability.struDispMode[i].byDispMode[j];
                        printf("The VGA1 supports the window mode: %d screen(s)\n",byDispMode);
                    }
                }
            }
            //...其他显示通道
        }

        //...其他能力
    }

    //下面配置第 2 个解码通道解码输出到第 1 个 VGA byDispMode 画面分割的左上角窗口
    DWORD DecChanNum=struDecoderCapability.byStartChan+1;

    DWORD dec = 0; //解码通道开关状态: 0 表示关闭, 1 表示打开
    if(!NET_DVR_MatrixGetDecChanEnable(IUserID, DecChanNum, &dec)) //获取解码通道开关
    {
        printf("NET_DVR_MatrixGetDecChanEnable error, %d\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }
    dec = 1; //设置解码通道为开: 0 表示关闭, 1 表示打开
    if(!NET_DVR_MatrixSetDecChanEnable(IUserID, DecChanNum, dec)) //设置解码通道开关, 如果设置为关, 通道将停止解码
    {
        printf("NET_DVR_MatrixSetDecChanEnable error, %d\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
    }

```

```

    return;
}

//配置显示通道参数，包括画面分割、关联解码通道
DWORD DispChanNum=struDecoderCapability.struVgaInfo.byStartChan;//VGA1
NET_DVR_MATRIX_VOUTCFG struVoutCfg;

if (!NET_DVR_MatrixGetDisplayCfg_V41(IUserID,DispChanNum,&struVoutCfg))
{
    printf("NET_DVR_MatrixGetDisplayCfg_V41 error, %d\\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

struVoutCfg.dwWindowMode=byDispMode;//取 VGA1 支持的画面分割能力的最后一个模式
for (i=0;i<MAX_WINDOWS_V41;i++)
{
    struVoutCfg.byJoinDecChan[i]=0; //所有的窗口关联的解码通道先清零
}

struVoutCfg.byJoinDecChan[0]=DecChanNum;//第一个窗口关联第 DecChanNum 个解码通道
//...其他参数设置

if (!NET_DVR_MatrixSetDisplayCfg_V41(IUserID,DispChanNum,&struVoutCfg))
{
    printf("NET_DVR_MatrixSetDisplayCfg_V41 error, %d\\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//启动动态解码
NET_DVR_PU_STREAM_CFG_V41 struDynamicInfo={0};
struDynamicInfo.dwSize=sizeof(struDynamicInfo);
//取流模式 byStreamMode: 0- 无效, 1- 通过 IP 或域名取流, 2- 通过 URL 取流, 3- 通过动态域名解析向设备取流
struDynamicInfo.byStreamMode=1;
//通道类型: 0-普通通道, 1-零通道, 2-流 ID, 解码器支持的通道类型可以通过能力集 DECODER_ABILITY 获取
struDynamicInfo.uDecStreamMode.struDecStreamDev.struDevChanInfo.byChanType=0;
//前端设备 IP 地址
strcpy((char *)struDynamicInfo.uDecStreamMode.struDecStreamDev.struDevChanInfo.byAddress,"172.6.22.38");
//前端设备服务端口
struDynamicInfo.uDecStreamMode.struDecStreamDev.struDevChanInfo.wDVRPort=8000;
//取通道 1 的码流
struDynamicInfo.uDecStreamMode.struDecStreamDev.struDevChanInfo.byChannel=1;
//前端设备登录用户名
strcpy((char *)struDynamicInfo.uDecStreamMode.struDecStreamDev.struDevChanInfo.sUserName,"admin");

```

```

//前端设备登录密码
strcpy((char *)struDynamicInfo.uDecStreamMode.struDecStreamDev.struDevChanInfo.sPassword,"12345");
//厂家私有协议，其他类型可以通过接口 NET_DVR_GetIPCProtoList 获取
struDynamicInfo.uDecStreamMode.struDecStreamDev.struDevChanInfo.byFactoryType=0;

if (!NET_DVR_MatrixStartDynamic_V41(IUserID,DecChanNum,&struDynamicInfo))
{
    printf("NET_DVR_MatrixStartDynamic_V41 error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}
//调用动态解码接口后，设备会自动启用解码开关，前面可不调用 NET_DVR_MatrixGetDecChanEnable

printf("Dynamic decode started, ... decoding channel: %d \n",DecChanNum);
printf("Please wait 10s...");
Sleep(10000);

if (!NET_DVR_MatrixStopDynamic(IUserID,DecChanNum))
{
    printf("NET_DVR_MatrixStopDynamic error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}
printf("Dynamic decode stopped.\n"); //停止解码

//获取轮巡解码配置信息
NET_DVR_MATRIX_LOOP_DECINFO_V41 struLoopDecInfoV41;
if (!NET_DVR_MatrixGetLoopDecChanInfo_V41(IUserID,DecChanNum,&struLoopDecInfoV41))
{
    printf("NET_DVR_MatrixGetLoopDecChanInfo_V41 error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}
struLoopDecInfoV41.dwPoolTime=10; //设置轮巡时间为 10s
//一路解码通道最大支持 64 路前端设备通道进行轮巡，这里只设置 3 个通道轮巡
for (i=0;i<3;i++)
{
    struLoopDecInfoV41.struchanConInfo[i].byEnable=1; //启用
    //取流模式： 0-无效， 1-通过 IP 或普通域名取流， 2-通过 URL 取流， 3-通过 DDNS 域名解析向设备取流
    struLoopDecInfoV41.struchanConInfo[i].byStreamMode=1;
    //通道类型： 0-普通通道， 1-零通道， 2-流 ID， 解码器支持的通道类型可以通过能力集 DECODER_ABILITY 获取

```

```

    struLoopDecInfoV41.struchanConInfo[i].uDecStreamMode.struDecStreamDev.struDevChanInfo.byChanType=0;
    //前端设备 IP 地址
    strcpy((char *)struLoopDecInfoV41.struchanConInfo[i].uDecStreamMode.struDecStreamDev.struDevChanInfo.byAddress,
"172.6.22.38");
    //前端设备服务端口号
    struLoopDecInfoV41.struchanConInfo[i].uDecStreamMode.struDecStreamDev.struDevChanInfo.wDVRPort=8000;
    //前端设备通道
    struLoopDecInfoV41.struchanConInfo[i].uDecStreamMode.struDecStreamDev.struDevChanInfo.byChannel=i+1;
    //厂家私有协议，其他类型可以通过接口 NET_DVR_GetIPCProtoList 获取
    struLoopDecInfoV41.struchanConInfo[i].uDecStreamMode.struDecStreamDev.struDevChanInfo.byFactoryType=0;
    //传输码流模式：0-主码流，1-子码流
    struLoopDecInfoV41.struchanConInfo[i].uDecStreamMode.struDecStreamDev.struDevChanInfo.byTransMode=0;
    //传输协议类型：0-TCP，1-UDP
    struLoopDecInfoV41.struchanConInfo[i].uDecStreamMode.struDecStreamDev.struDevChanInfo.byTransProtocol=0;
    //前端设备登录用户名
    strcpy((char *)struLoopDecInfoV41.struchanConInfo[i].uDecStreamMode.struDecStreamDev.struDevChanInfo.sUserName,"admin");
    //前端设备登录密码
    strcpy((char *)struLoopDecInfoV41.struchanConInfo[i].uDecStreamMode.struDecStreamDev.struDevChanInfo.sPassword,"12345");
}
//...其他参数设置

//设置轮巡解码通道参数
if (!NET_DVR_MatrixSetLoopDecChanInfo_V41(IUserID,DecChanNum,&struLoopDecInfoV41))
{
    printf("NET_DVR_MatrixSetDisplayCfg_V41 error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

DWORD chanNum = 0;
//获取当前通道解码开关,返回值 chanNum 为 0 表示关闭，1 表示打开
if (!NET_DVR_MatrixGetLoopDecChanEnable(IUserID, DecChanNum, &chanNum))
{
    printf("NET_DVR_MatrixGetLoopDecChanEnable error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

chanNum = 1; //设置解码开关为开
if (!NET_DVR_MatrixSetLoopDecChanEnable(IUserID, DecChanNum, chanNum))
{
    printf("NET_DVR_MatrixSetLoopDecChanEnable error, %d\n", NET_DVR_GetLastError());
}

```

```

        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }
    printf("Cycle decode started!\n");
    printf("Please wait 50s...\n");
    Sleep(50000);

    chanNum = 0; //设置解码开关为关
    //当把当前正在轮循的解码通道轮巡开关设置为关时，该解码通道停止循环，转为动态解码
    if (!NET_DVR_MatrixSetLoopDecChanEnable(IUserID, DecChanNum, chanNum))
    {
        printf("NET_DVR_MatrixSetLoopDecChanEnable error, %d\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }

    //关闭解码通道解码开关，停止解码
    if (!NET_DVR_MatrixSetDecChanEnable(IUserID, DecChanNum, chanNum))
    {
        printf("NET_DVR_MatrixSetDecChanEnable error, %d\n", NET_DVR_GetLastError());
        NET_DVR_Logout(IUserID);
        NET_DVR_Cleanup();
        return;
    }
    printf("Cycle decode stopped!\n");

    return;
}

```

### 4.1.2 远程回放解码

[相关模块流程图](#)

```

#include <stdio.h>
#include <string.h>
#include <iostream>
#include "Windows.h"
#include "HCNetSDK.h"
using namespace std;

void main()
{

```



```

int i=0,j=0;
BYTE byDispMode;

//初始化 SDK
NET_DVR_Init();

//注册设备
NET_DVR_DEVICEINFO_V30 struDeviceInfo={0};
memset(&struDeviceInfo, 0, sizeof(NET_DVR_DEVICEINFO_V30));//存放设备参数的结构体
LONG lUserID = NET_DVR_Login_V30("172.6.22.178", 8000, "admin", "12345", &struDeviceInfo);
if (lUserID < 0)
{
    printf("Login error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Cleanup();
    return;
}

//获取解码器能力集
NET_DVR_MATRIX_ABILITY_V41 struDecoderCapability={0};
if (!NET_DVR_GetDeviceAbility(lUserID, MATRIXDECODER_ABILITY_V41, NULL, 0, (char *)&struDecoderCapability,
sizeof(struDecoderCapability)))
{
    printf("NET_DVR_GetDeviceAbility error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(lUserID);
    NET_DVR_Cleanup();
    return;
}
else
{
    printf("Starting number of decoding channel: %d\n", struDecoderCapability.byStartChan);
    printf("The total number of VGA output: %d\n", struDecoderCapability.struVgaInfo.byChanNums);
    printf("Starting number of VGA output: %d\n", struDecoderCapability.struVgaInfo.byStartChan);
    for (i=0; i<MAX_DISPNUM_V41; i++)
    {
        if (struDecoderCapability.struDispMode[i].byDispChanType==1&&struDecoderCapability.struDispMode[i].byDispChanSeq==1)
        {
            //获取 VGA 1 输出支持的画面分割模式
            for (j=0; j<MAX_WINDOWS_NUM; j++)
            {
                if (struDecoderCapability.struDispMode[i].byDispMode[j]>0)
                {
                    byDispMode=struDecoderCapability.struDispMode[i].byDispMode[j];
                    printf("The VGA1 supports the window mode: %d screen(s)\n", byDispMode);
                }
            }
        }
    }
}

```

```

    }
}
//...其他显示通道
}
//...其他能力
}

//下面配置第 2 个解码通道解码输出到第 1 个 VGA byDispMode 画面分割的左上角窗口
DWORD DecChanNum=struDecoderCapability.byStartChan+1;

DWORD dec = 0; //解码通道开关状态: 0 表示关闭, 1 表示打开
if(!NET_DVR_MatrixGetDecChanEnable(IUserID, DecChanNum, &dec)) //获取解码通道开关
{
    printf("NET_DVR_MatrixGetDecChanEnable error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}
dec = 1; //设置解码通道为开: 0 表示关闭, 1 表示打开
if(!NET_DVR_MatrixSetDecChanEnable(IUserID, DecChanNum, dec)) //设置解码通道开关, 如果设置为关, 通道将停止解码
{
    printf("NET_DVR_MatrixSetDecChanEnable error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//配置显示通道参数, 包括画面分割、关联解码通道
DWORD DispChanNum=struDecoderCapability.struVgaInfo.byStartChan;//VGA1
NET_DVR_MATRIX_VOUTCFG struVoutCfg;

if (!NET_DVR_MatrixGetDisplayCfg_V41(IUserID,DispChanNum,&struVoutCfg))
{
    printf("NET_DVR_MatrixGetDisplayCfg_V41 error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

struVoutCfg.dwWindowMode=byDispMode;//取 VGA1 支持的画面分割能力的最后一个模式
for (i=0;i<MAX_WINDOWS_V41;i++)
{
    struVoutCfg.byJoinDecChan[i]=0; //所有的窗口关联的解码通道先清零
}

struVoutCfg.byJoinDecChan[0]=DecChanNum;//第一个窗口关联第 DecChanNum 个解码通道
//...其他参数设置

```

```
if (!NET_DVR_MatrixSetDisplayCfg_V41(IUserID,DispChanNum,&struVoutCfg))
{
    printf("NET_DVR_MatrixSetDisplayCfg_V41 error, %d\\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}
//设置解码远程设备录像文件
NET_DVR_MATRIX_DEC_REMOTE_PLAY_EX struDecRemotePlay={0};
struDecRemotePlay.dwSize=sizeof(struDecRemotePlay);
//解码通道号
struDecRemotePlay.dwDecChannel=DecChanNum;
//设备地址类型: 0- IP, 1- DDNS 域名
struDecRemotePlay.byAddressType=0;
//通道类型: 0- 普通通道, 1- 零通道, 2- 流 ID
struDecRemotePlay.byChannelType=0;
//前端设备登录用户名
strcpy((char *)struDecRemotePlay.sUserName,"admin");
//前端设备登录密码
strcpy((char *)struDecRemotePlay.sPassword,"12345");
//前端设备通道号
struDecRemotePlay.dwChannel =33;
//回放方式: 0-按文件,1-按时间
struDecRemotePlay.dwPlayMode=1;
//前端设备 IP 地址
strcpy((char *)struDecRemotePlay.unionAddr.strulpAddr.byDevAddress,"172.6.22.165");
//前端设备服务端口
struDecRemotePlay.unionAddr.strulpAddr.wDevPort=8000;

//按时间回放起止时间
struDecRemotePlay.unionPlayBackInfo.struPlayBackByTime.StartTime.dwYear=2013;
struDecRemotePlay.unionPlayBackInfo.struPlayBackByTime.StartTime.dwMonth=11;
struDecRemotePlay.unionPlayBackInfo.struPlayBackByTime.StartTime.dwDay=18;
struDecRemotePlay.unionPlayBackInfo.struPlayBackByTime.StartTime.dwHour=15;
struDecRemotePlay.unionPlayBackInfo.struPlayBackByTime.StartTime.dwMinute=45;
struDecRemotePlay.unionPlayBackInfo.struPlayBackByTime.StartTime.dwSecond=0;
struDecRemotePlay.unionPlayBackInfo.struPlayBackByTime.StopTime.dwYear=2013;
struDecRemotePlay.unionPlayBackInfo.struPlayBackByTime.StopTime.dwMonth=11;
struDecRemotePlay.unionPlayBackInfo.struPlayBackByTime.StopTime.dwDay=18;
struDecRemotePlay.unionPlayBackInfo.struPlayBackByTime.StopTime.dwHour=15;
struDecRemotePlay.unionPlayBackInfo.struPlayBackByTime.StopTime.dwMinute=50;
struDecRemotePlay.unionPlayBackInfo.struPlayBackByTime.StopTime.dwSecond=0;
```

```

//如果是按文件回放，需要先获知设备本地录像文件的文件名，文件查找请参考回放和下载模块流程
if (!NET_DVR_RemoteControl(IUserID,NET_DVR_DEC_PLAY_REMOTE_FILE, &struDecRemotePlay,
sizeof(NET_DVR_MATRIX_DEC_REMOTE_PLAY_EX)))
{
    printf("NET_DVR_DEC_PLAY_REMOTE_FILE error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

//开始回放
if (!NET_DVR_MatrixSetRemotePlayControl(IUserID,DecChanNum,NET_DVR_PLAYSTART,0,NULL))
{
    printf("NET_DVR_MatrixSetRemotePlayControl NET_DVR_PLAYSTART error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

printf("Start to decode the video remotely by time.\n");

NET_DVR_MATRIX_DEC_REMOTE_PLAY_STATUS struDecRemoteStatus={0};

while(struDecRemoteStatus.dwCurDataType!=21) //当前传输的数据类型：19-文件头，20-流数据，21-播放结束标志
{
    int iPos=0;
    DWORD dwCurMediaFileDuration=0;
    NET_DVR_MatrixGetRemotePlayStatus(IUserID,DecChanNum,&struDecRemoteStatus); //获取回放状态
    dwCurMediaFileDuration=struDecRemoteStatus.dwCurMediaFileDuration;
    if (dwCurMediaFileDuration>0)
    {
        iPos=struDecRemoteStatus.dwCurPlayTime*100/dwCurMediaFileDuration; //当前回放进度
        Sleep(10000); //每隔 10s 获取一下进度
        printf("Playback progress: %d%%\n", iPos);
    }
}

printf("The remote playing is finished.\n");

//停止回放
if (!NET_DVR_MatrixSetRemotePlayControl(IUserID,DecChanNum,NET_DVR_PLAYSTOP,0,NULL))
{
    printf("NET_DVR_MatrixSetRemotePlayControl NET_DVR_PLAYSTOP error, %d\n", NET_DVR_GetLastError());
    NET_DVR_Logout(IUserID);
    NET_DVR_Cleanup();
    return;
}

```

```
return;  
}
```

## 4.2 被动解码模块的示例代码

[相关模块流程图](#)

```
#include <stdio.h>  
#include <iostream>  
#include "Windows.h"  
#include "HCNetSDK.h"  
using namespace std;  
  
void main() {  
  
    //-----  
    // 初始化  
    NET_DVR_Init();  
    //设置连接时间与重连时间  
    NET_DVR_SetConnectTime(2000, 1);  
    NET_DVR_SetReconnect(10000, true);  
  
    //注册设备  
    NET_DVR_DEVICEINFO_V30 struDeviceInfo;  
    memset(&struDeviceInfo, 0, sizeof(NET_DVR_DEVICEINFO_V30)); //存放设备参数的结构体  
    LONG lUserID = NET_DVR_Login_V30("172.0.0.100", 8000, "admin", "12345", &struDeviceInfo);  
  
    if (lUserID < 0)  
    {  
        if (NET_DVR_GetLastError() == NET_DVR_PASSWORD_ERROR) //用户名密码错误  
        {  
            ..... //处理错误信息  
        }  
        else if (NET_DVR_GetLastError() == NET_DVR_OVER_MAXLINK) //连接到 DVR 的客户端达到最大  
        {  
            ..... //处理错误信息  
        }  
        ..... //处理其他类型错误信息  
    }  
    //获取多路解码器解码显示能力  
    NET_DVR_MATRIX_ABILITY m_matrixability;  
    NET_DVR_GetDeviceAbility(lUserID, MATRIXDECODER_ABILITY, NULL, 0, (char*)&m_matrixability,  
sizeof(NET_DVR_MATRIX_ABILITY));  
  
    //显示通道配置
```

```

DWORD DispChanNum=1;//显示通道，可以通过能力集获取
NET_DVR_MATRIX_VOUTCFG VoutCfg;
if(!NET_DVR_MatrixGetDisplayCfg_V41(IUserID, DispChanNum, &VoutCfg))
{
    ..... //出错处理
}
VoutCfg.dwWindowMode = 4; //画面分割设为 4 画面
VoutCfg.byJoinDecChan[0] = 1; //左上角第一个窗口关联解码通道 1
..... //设置其他显示通道参数

if(!NET_DVR_MatrixSetDisplayCfg_V41(IUserID, DispChanNum, &VoutCfg))
{
    ..... //出错处理
}

int DecChanNum = 1;//解码通道号，此处假设通道号为 1
if(!NET_DVR_MatrixGetDecChanEnable(IUserID, DecChanNum, &dec)) //获取解码通道开关
{
    ..... //出错处理
}

dec = 1; //设置解码通道为开；0 表示关闭，1 表示打开
if(!NET_DVR_MatrixSetDecChanEnable(IUserID, DecChanNum, dwEnable))
//设置解码通道开关，如果设置为关，通道将停止解码
{
    ..... //出错处理
}

//被动解码
DWORD m_PassivePort = 8000;
LONG IPassiveModeHandle = -1; //被动解码句柄
NET_DVR_MATRIX_PASSIVEMODE m_PassiveMode;
m_PassiveMode.wPassivePort = m_PassivePort;//UDP 时的端口号，TCP 时默认为 8000
//.....其他被动解码参数
IPassiveModeHandle = NET_DVR_MatrixStartPassiveDecode(IUserID, DecChanNum, &m_PassiveMode);
//启 动多路解码器被动解码
NET_DVR_MatrixSendData(IPassiveModeHandle, pSendBuf, dwBufSize);
//向解码器发送数据,pSendBuf 为发送数据缓冲区，dwBufSize 为数据大小

NET_DVR_MatrixStopPassiveDecode(IPassiveModeHandle); //停止多路解码器被动解码

//注销用户
NET_DVR_Logout(IUserID);
//释放 SDK 资源

```

```
NET_DVR_Cleanup();  
return;  
}
```

## 5 函数说明

### 5.1 SDK 初始化

#### 5.1.1 初始化 SDK **NET\_DVR\_Init**

函 数： BOOL NET\_DVR\_Init()  
参 数： 无  
返回值： TRUE 表示成功，FALSE 表示失败。  
说 明： 调用设备网络 SDK 其他函数的前提。

[返回目录](#)

#### 5.1.2 释放 SDK 资源 **NET\_DVR\_Cleanup**

函 数： BOOL NET\_DVR\_Cleanup()  
参 数： 无  
返回值： TRUE 表示成功，FALSE 表示失败。  
说 明： 在结束之前最后调用。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

[返回目录](#)

### 5.2 SDK 本地功能

#### SDK 本地参数配置

##### 5.2.1 获取 SDK 本地参数 **NET\_DVR\_GetSDKLocalCfg**

函 数： BOOL NET\_DVR\_GetSDKLocalCfg(NET\_SDK\_LOCAL\_CFG\_TYPE enumType, void \*lpOutBuff)  
参 数： [in] enumType 配置类型，不同的取值对应不同的 SDK 参数，详见表 5.1  
[out] lpOutBuff 输出参数，不同的配置类型，输出参数对应不同的结构，详见表 5.1  
返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。  
说 明：

表 5.1 本地参数类型

enumType 宏定义	类型值	含义	lpOutBuff 对应结构体
NET_SDK_LOCAL_CFG_TYPE_TCP_PORT_BIND	0	本地 TCP 端口绑定配置	NET_DVR_LOCAL_TCP_PORT_BIND_CFG
NET_SDK_LOCAL_CFG_TYPE_UDP_PORT_BIND	1	本地 UDP 端口绑定配置	NET_DVR_LOCAL_UDP_PORT_BIND_CFG
NET_SDK_LOCAL_CFG_TYPE_MEM_POOL	2	内存池本地配置	NET_DVR_LOCAL_MEM_POOL_CFG



NET_SDK_LOCAL_CFG_TYPE_MODULE_RECV_TIMEOUT	3	按模块配置超时时间	NET_DVR_LOCAL_MODULE_RECV_TIMEOUT_CFG
NET_SDK_LOCAL_CFG_TYPE_ABILITY_PARSE	4	是否使用能力集解析库	NET_DVR_LOCAL_ABILITY_PARSE_CFG
NET_SDK_LOCAL_CFG_TYPE_TALK_MODE	5	对讲模式配置	NET_DVR_LOCAL_TALK_MODE_CFG
NET_SDK_LOCAL_CFG_TYPE_CHECK_DEV	10	心跳交互间隔时间配置	NET_DVR_LOCAL_CHECK_DEV

[返回目录](#)

## 5.2.2 设置 SDK 本地参数 **NET\_DVR\_SetSDKLocalCfg**

函数: `BOOL NET_DVR_SetSDKLocalCfg(NET_SDK_LOCAL_CFG_TYPE enumType, void* const lpInBuff)`

参数: `[in] enumType` 配置类型，不同的取值对应不同的 SDK 参数，详见表 5.2  
`[in] lpInBuff` 输入参数，不同的配置类型，输出参数对应不同的结构，详见表 5.2

返回值: `TRUE` 表示成功，`FALSE` 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说明:

表 5.2 本地参数类型

enumType 宏定义	类型值	含义	lpInBuff 对应结构体
NET_SDK_LOCAL_CFG_TYPE_TCP_PORT_BIND	0	本地 TCP 端口绑定配置	NET_DVR_LOCAL_TCP_PORT_BIND_CFG
NET_SDK_LOCAL_CFG_TYPE_UDP_PORT_BIND	1	本地 UDP 端口绑定配置	NET_DVR_LOCAL_UDP_PORT_BIND_CFG
NET_SDK_LOCAL_CFG_TYPE_MEM_POOL	2	内存池本地配置	NET_DVR_LOCAL_MEM_POOL_CFG
NET_SDK_LOCAL_CFG_TYPE_MODULE_RECV_TIMEOUT	3	按模块配置超时时间	NET_DVR_LOCAL_MODULE_RECV_TIMEOUT_CFG
NET_SDK_LOCAL_CFG_TYPE_ABILITY_PARSE	4	是否使用能力集解析库	NET_DVR_LOCAL_ABILITY_PARSE_CFG
NET_SDK_LOCAL_CFG_TYPE_TALK_MODE	5	对讲模式配置	NET_DVR_LOCAL_TALK_MODE_CFG
NET_SDK_LOCAL_CFG_TYPE_CHECK_DEV	10	心跳交互间隔时间配置	NET_DVR_LOCAL_CHECK_DEV
NET_SDK_LOCAL_CFG_TYPE_CHAR_ENCODE	13	配置字符编码相关处理回调	NET_DVR_LOCAL_BYTE_ENCODE_CONVERT
NET_DVR_LOCAL_CFG_TYPE_LOG	15	日志参数配置	NET_DVR_LOCAL_LOG_CFG

[返回目录](#)

## 连接和接收超时时间及重连设置

## 5.2.3 设置网络连接超时时间和连接尝试次数 **NET\_DVR\_SetConnectTime**

函数: `BOOL NET_DVR_SetConnectTime(DWORD dwWaitTime, DWORD dwTryTime)`

参数: `[in] dwWaitTime` 超时时间，单位毫秒，取值范围[300,75000]，实际最大超时时间因系统的 connect 超时时间而不同。  
`[in] dwTryTimes` 连接尝试次数（保留）

返回值: `TRUE` 表示成功，`FALSE` 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说明: **SDK 默认建立连接的超时时间为 3 秒。** SDK4.0 及以后版本中当设置的超时时间超过或低于限制

的值时接口不返回失败，将取最接近的上下限限制值作为实际的超时时间。

[返回目录](#)

## 5.2.4 设置重连功能 **NET\_DVR\_SetReconnect**

函 数: BOOL NET\_DVR\_SetReconnect (DWORD dwInterval, BOOL bEnableRecon)

参 数: [in] dwInterval 重连间隔，单位:毫秒  
[in] bEnableRecon 是否重连，0-不重连，1-重连，参数默认为 1

返回值: TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明: 该接口可以同时控制预览、透明通道和布防的重连功能。不调用该接口时，SDK 默认启动预览、透明通道和布防的重连功能，重连时间间隔为 5 秒。

[返回目录](#)

## 5.2.5 设置接收超时时间 **NET\_DVR\_SetRecvTimeOut**

函 数: BOOL NET\_DVR\_SetRecvTimeOut (DWORD nRecvTimeOut)

参 数: [in] nRecvTimeOut 接收超时时间，单位毫秒，默认为 5000，最小为 3000 毫秒

返回值: TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明: 该接口用于设置接收超时时间，例如预览接收实时流数据、回放下载接收录像数据、报警接收报警信息等接收超时时间。

[返回目录](#)

## 多网卡绑定

## 5.2.6 获取所有 IP，用于支持多网卡接口 **NET\_DVR\_GetLocalIP**

函 数: BOOL NET\_DVR\_GetLocalIP (char strIP[16][16], DWORD \*pValidNum, BOOL \*pEnableBind)

参 数: [out] strIP 存放 IP 的缓冲区，不能为空  
[out] pValidNum 所有有效 IP 的数量  
[out] pEnableBind 是否绑定

返回值: TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明: 该接口获取客户端本地多网卡的所有 IP 地址，可以通过接口 NET\_DVR\_SetValidIP 选择要使用的 IP 地址。

[返回目录](#)

## 5.2.7 设置 IP 绑定 **NET\_DVR\_SetValidIP**

函 数: BOOL NET\_DVR\_SetValidIP (DWORD dwIPIndex, BOOL bEnableBind)

参 数: [in] dwIPIndex 选择使用的 IP 下标，由 NET\_DVR\_GetLocalIP 获取  
[in] bEnableBind 是否绑定

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

## SDK 版本、状态和能力

### 5.2.8 获取 SDK 的版本号和 build 信息 **NET\_DVR\_GetSDKBuildVersion**

函 数: DWORD NET\_DVR\_GetSDKBuildVersion()

参 数:

返回值: 获取 SDK 的版本号和 build 信息。

说 明: SDK 的版本号和 build 信息。2 个高字节表示版本号 : 25~32 位表示主版本号, 17~24 位表示次版本号; 2 个低字节表示 build 信息。如 0x03000101: 表示版本号为 3.0, build 号是 0101。

[返回目录](#)

### 5.2.9 获取当前 SDK 的状态信息 **NET\_DVR\_GetSDKState**

函 数: BOOL NET\_DVR\_GetSDKState( LPNET\_DVR\_SDKSTATE pSDKState);

参 数: [out]pSDKState SDK 状态信息, 详见结构体: NET\_DVR\_SDKSTATE

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

### 5.2.10 获取当前 SDK 的功能信息 **NET\_DVR\_GetSDKAbility**

函 数: BOOL NET\_DVR\_GetSDKAbility( LPNET\_DVR\_SDKABL pSDKAbl)

参 数: [out] pSDKAbl SDK 功能信息, 详见结构体: NET\_DVR\_SDKABL

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

## SDK 启用写日志

### 5.2.11 启用写日志文件 **NET\_DVR\_SetLogToFile**

函 数: BOOL NET\_DVR\_SetLogToFile(DWORD bLogEnable,char\* strLogDir,BOOL bAutoDel)

参 数:	[in]bLogEnable	日志的等级（默认为 0）： 0-表示关闭日志 1-表示只输出 ERROR 错误日志 2-输出 ERROR 错误信息和 DEBUG 调试信息 3-输出 ERROR 错误信息、DEBUG 调试信息和 INFO 普通信息等所有信息
	[in]strLogDir	日志文件的路径，windows 默认值为"C:\\SdkLog\\"；linux 默认值"/home/sdklog/"
	[in]bAutoDel	是否删除超出的文件数，默认值为 TRUE
返回值:	TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 <a href="#">NET_DVR_GetLastError</a> 获取错误码，通过错误码判断出错原因。	
说 明:	<ul style="list-style-type: none"> <li>日志文件路径必须是绝对路径，且以"\"结尾，例如"C:\\SdkLog\\"，建议用户先手动创建文件。若未指定文件路径，则采用默认路径"C:\\SdkLog\\"。</li> <li>可多次调用该接口创建新的日志文件，更改目录时到下一次写文件时才会使用新的目录写文件。</li> <li>bAutoDel 为 TRUE 时表示覆盖模式，日志文件个数超过 SDK 限制个数时将会自动删除超出的文件。SDK 限制个数默认为 10 个，可以调用接口 <a href="#">NET_DVR_SetSDKLocalCfg</a>(配置类型：NET_DVR_LOCAL_CFG_TYPE_LOG)进行修改配置。</li> </ul>	

[返回目录](#)

## 异常消息回调

### 5.2.12 注册接收异常、重连等消息的窗口句柄或回调函数

#### NET\_DVR\_SetExceptionCallBack\_V30

函 数:	<b>Windows 系统下:</b> BOOL NET_DVR_SetExceptionCallBack_V30 (UINT nMessage,HWND hWnd,fExceptionCallBack cbExceptionCallBack,void* pUser) <b>Linux 系统下:</b> BOOL NET_DVR_SetExceptionCallBack_V30(UINT nMessage,void* hWnd,fExceptionCallBack cbExceptionCallBack,void* pUser)	
参 数:	[in]nMessage	消息,Linux 下该参数保留
	[in]hWnd	接收异常消息的窗口句柄，Linux 下该参数保留
	[in]cbExceptionCallBack	接收异常消息的回调函数，回调当前异常的相关信息
	[in]pUser	用户数据
	typedef void(CALLBACK* fExceptionCallBack)(DWORD dwType, LONG lUserID, LONG lHandle, void* pUser)	
	[out]dwType	异常或重连等消息的类型，详见表 5.3
	[out]lUserID	登录 ID
	[out]lHandle	出现异常的相应类型的句柄
	[out]pUser	用户数据

表 5.3 异常消息类型

dwType 宏定义	宏定义值	含义
EXCEPTION_EXCHANGE	0x8000	用户交互时异常（注册心跳超时，心跳间隔为 2 分钟）
EXCEPTION_AUDIOEXCHANGE	0x8001	语音对讲异常
EXCEPTION_ALARM	0x8002	报警异常
EXCEPTION_ALARMRECONNECT	0x8006	报警时重连
ALARM_RECONNECTSUCESS	0x8016	报警时重连成功
RESUME_EXCHANGE	0x8017	用户交互恢复

返回值：TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：**Windows 下该函数的 hWnd 和 cbExceptionCallBack 不能同时为 NULL，Linux 下 cbExceptionCallBack 不能设置为 NULL，否则将接收不到异常消息。

如果此结构是以回调方式反馈异常消息，那么应用程序中的异常回调函数实现如下，该函数中的参数 dwType 表示异常消息类型（见上表）；IHandle 表示发生异常的相应类型的句柄。

**示例代码：**

```
//注册接收异常消息的回调函数
NET_DVR_SetExceptionCallBack_V30(WM_NULL, NULL, g_ExceptionCallBack, NULL);
//接收异常消息的回调函数的外部实现
void CALLBACK g_ExceptionCallBack(DWORD dwType, LONG IUserID, LONG IHandle, void *pUser)
{
    char tempbuf[256];
    ZeroMemory(tempbuf,256);
    switch(dwType)
    {
        case EXCEPTION_ALARM: //报警上传时网络异常
            sprintf(tempbuf,"报警上传时网络异常!!!");
            TRACE("%s",tempbuf);
            //TODO: 关闭报警上传
            break;
        case EXCEPTION_EXCHANGE: //用户交互时异常
            sprintf(tempbuf,"用户交互时网络异常!!!");
            TRACE("%s",tempbuf);
            //TODO: 注销登录
            break;
        default:
            break;
    }
};
```

[返回目录](#)

## 获取错误信息

### 5.2.13 返回最后操作的错误码 **NET\_DVR\_GetLastError**

函 数: DWORD NET\_DVR\_GetLastError()

参 数:

返回值: 返回最后操作的错误码。详见[错误码宏定义](#)

说 明: 返回值为错误码。错误码主要分为网络通讯库错误码、RTSP 通讯库错误码和软硬解库错误码。

[返回目录](#)

### 5.2.14 返回最后操作的错误码信息 **NET\_DVR\_GetErrorMsg**

函 数: char\* NET\_DVR\_GetErrorMsg(LONG \*pErrorNo)

参 数: [out]pErrorNo 错误码数值的指针

返回值: 返回值为错误码信息的指针。错误码主要分为网络通讯库错误码、RTSP 通讯库错误码和软硬解库错误码。详见[错误码宏定义](#)

说 明:

[返回目录](#)

## 5.3 用户注册

### 5.3.1 激活设备 **NET\_DVR\_ActivateDevice**

函 数: BOOL NET\_DVR\_ActivateDevice(char\* sDVRIP, WORD wDVRPort, LPNET\_DVR\_ACTIVATECFG lpActivateCfg)

参 数: [in]sDVRIP 设备 IP 地址

[in]wDVRPort 设备端口

[in]lpActivateCfg 激活参数，包括激活使用的初始密码

返回值: TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明: 出厂设备需要先激活，然后再使用激活使用的初始密码登录设备。

[返回目录](#)

### 5.3.2 通过解析服务器，获取设备的动态 IP 地址和端口号

#### **NET\_DVR\_GetDVRIPByResolveSvr\_EX**

函 数: BOOL NET\_DVR\_GetDVRIPByResolveSvr\_EX (char\* sServerIP, WORD wServerPort, BYTE\* sDVRName, WORD wDVRNameLen, BYTE\* sDVRSerialNumber, WORD wDVRSerialLen, char\* sGetIP, DWORD\* dwPort)

**参 数:**

[in]sServerIP	解析服务器的 IP 地址
[in]wServerPort	解析服务器的端口号, IP Server 解析服务器端口号为 7071, HiDDNS 服务器的端口号为 80
[in]sDVRName	设备名称
[in]wDVRNameLen	设备名称的长度
[in]sDVRSerialNumber	设备的序列号
[in]wDVRSerialLen	设备序列号的长度
[out]sGetIP	获取到的设备 IP 地址指针
[out]dwPort	获取到的设备端口号指针

**返回值:** TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:** 该接口中的设备名称和设备序列号不能同时为空。通过设备域名或者序列号解析出设备当前 IP 地址和端口, 然后调用 [NET\\_DVR\\_Login\\_V30](#) 登录设备。  
支持的解析服务器有 IPServer 和 hiDDNS。

[返回目录](#)

### 5.3.3 用户注册设备 **NET\_DVR\_Login\_V40**

**函 数:** LONG NET\_DVR\_Login\_V40(LPNET\_DVR\_USER\_LOGIN\_INFO pLoginInfo, LPNET\_DVR\_DEVICEINFO\_V40 lpDeviceInfo)

**参 数:**

[in]pLoginInfo	登录参数, 包括设备地址、登录用户、密码等
[out]lpDeviceInfo	设备信息(同步登录即 pLoginInfo 中 bUseAsynLogin 为 0 时有效)

**返回值:** 异步登录的状态、用户 ID 和设备信息通过 NET\_DVR\_USER\_LOGIN\_INFO 结构体中设置的回调函数(fLoginResultCallBack)返回。对于同步登录, 接口返回-1 表示登录失败, 其他值表示返回的用户 ID 值。用户 ID 具有唯一性, 后续对设备的操作都需要通过此 ID 实现。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:**

- pLoginInfo 中 bUseAsynLogin 为 0 时登录为同步模式, 接口返回成功即表示登录成功; pLoginInfo 中 bUseAsynLogin 为 1 时登录为异步模式, 登录是否成功在输入参数设置的回调函数中返回。
- 设备同时最多允许 128 个用户注册。
- SDK 支持 2048 个注册, 返回 UserID 的取值范围为 0~2047。

[返回目录](#)

### 5.3.4 用户注销 **NET\_DVR\_Logout**

**函 数:** BOOL NET\_DVR\_Logout(LONG IUserID)

**参 数:** [in]IUserID 用户 ID 号, NET\_DVR\_Login\_V40 的返回值

**返回值:** TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:**

[返回目录](#)



## 5.4 获取设备能力集

### 5.4.1 获取设备能力集 **NET\_DVR\_GetDeviceAbility**

函 数: `BOOL NET_DVR_GetDeviceAbility(LONG lUserID, DWORD dwAbilityType, char* pInBuf, DWORD dwInLength, char* pOutBuf, DWORD dwOutLength)`

参 数:

- `[in] lUserID` 用户 ID 号, `NET_DVR_Login_V30` 返回值
- `[in] dwAbilityType` 能力类型, 具体定义详见表 5.4
- `[in] pInBuf` 输入缓冲区指针 (按照设备规定的能力参数的描述方式组合, 可以是 XML 文本或结构体形式), 详见表 5.5
- `[in] dwInLength` 输入缓冲区的长度
- `[out] pOutBuf` 输出缓冲区指针 (按照设备规定的能力集的描述方式, 可以是 XML 文本或结构体形式), 详见表 5.5
- `[in] dwOutLength` 接收数据的缓冲区的长度

表 5.4 设备能力集类型

dwAbilityType 宏定义	宏定义值	含义
MATRIXDECODER_ABILITY	0x200	多路解码器显示、解码能力
WALL_ABILITY	0x212	电视墙能力集
MATRIXDECODER_ABILITY_V41	0x260	解码器能力集 V41 扩展
DECODER_ABILITY	0x261	解码器 xml 能力集

返回值: `TRUE` 表示成功, `FALSE` 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明: 接口中 `pInBuf` 参数的具体定义格式按照不同的设备规定有所不同, 可以是以结构体的形式或者 XML 描述方式。同样地, 参数 `pOutBuf` 的输出表达格式也按不同的设备规定可以是以结构体的形式或者 XML 描述方式。表 5.5 列出了按照不同的能力类型获取时, 需要输入参数和输出参数的格式定义。

表 5.5 设备能力集描述

能力类型宏定义	能力类型说明	pInBuf	pOutBuf
MATRIXDECODER_ABILITY	获取多路解码器显示、解码能力	无	NET_DVR_MATRIX_ABILITY
WALL_ABILITY	获取电视墙能力	无	电视墙能力 XML 描述
MATRIXDECODER_ABILITY_V41	解码器能力集	无	NET_DVR_MATRIX_ABILITY_V41
DECODER_ABILITY	获取解码器 xml 能力集	解码器能力集获取输入描述	解码器能力集 XML 描述

注: 能力集结构体和 XML 描述请参见《设备网络 SDK 使用手册.chm》。

[返回目录](#)



## 5.5 显示通道配置和控制

### 5.5.1 获取显示通道信息 **NET\_DVR\_MatrixGetDisplayCfg\_V41**

**函 数：** BOOL NET\_DVR\_MatrixGetDisplayCfg\_V41(LONG IUserID, DWORD dwDispChanNum, LPNET\_DVR\_MATRIX\_VOUTCFG lpVoutCfg)

**参 数：** [in] IUserID 用户 ID 号，NET\_DVR\_Login\_V30 返回值  
[in] dwDispChanNum 显示通道，从能力集获取  
[out] lpVoutCfg 显示通道参数，详见结构：NET\_DVR\_MATRIX\_VOUTCFG

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：**

[返回目录](#)

### 5.5.2 显示通道配置 **NET\_DVR\_MatrixSetDisplayCfg\_V41**

**函 数：** BOOL NET\_DVR\_MatrixSetDisplayCfg\_V41(LONG IUserID, DWORD dwDispChanNum, LPNET\_DVR\_MATRIX\_VOUTCFG lpDisplayCfg)

**参 数：** [in] IUserID 用户 ID 号，NET\_DVR\_Login\_V30 返回值  
[in] dwDispChanNum 显示通道，从能力集获取  
[in] lpDisplayCfg 显示通道参数，详见结构体：NET\_DVR\_MATRIX\_VOUTCFG

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：**

[返回目录](#)

### 5.5.3 显示通道控制 **NET\_DVR\_MatrixDiaplayControl**

**函 数：** BOOL NET\_DVR\_MatrixDiaplayControl(LONG IUserID, DWORD dwDispChanNum, DWORD dwDispChanCmd, DWORD dwCmdParam)

**参 数：** [in] IUserID 用户 ID 号，NET\_DVR\_Login\_V30 返回值  
[in] dwDispChanNum 显示通道，从能力集获取  
[in] dwDispChanCmd 显示通道命令，具体定义见表 5.6  
[in] dwCmdParam 命令参数，保留

表 5.6 显示通道控制命令

dwDispChanCmd 宏定义	宏定义值	含义	dwCmdParam 取值
DISP_CMD_ENLARGE_WINDOW	1	显示通道放大某个窗口	子窗口号
DISP_CMD_RENEW_WINDOW	2	显示通道窗口还原	子窗口号

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：**

[返回目录](#)

## 5.6 参数配置

### 5.6.1 获取设备的配置信息 **NET\_DVR\_GetDVRConfig**

函 数: BOOL NET\_DVR\_GetDVRConfig(LONG UserID, DWORD dwCommand, LONG IChannel, LPVOID lpOutBuffer, DWORD dwOutBufferSize, LPDWORD lpBytesReturned)

参 数: [in] UserID 用户 ID 号, NET\_DVR\_Login\_V30 的返回值  
[in] dwCommand 设备配置命令, 详见表 5.7  
[in] IChannel 通道号, 不同的命令取值不同, 详见表 5.8。如果该参数无效, 置为 0xFFFFFFFF 即可  
[out] lpOutBuffer 接收数据的缓冲指针, 详见表 5.8  
[in] dwOutBufferSize 接收数据的缓冲长度(以字节为单位), 不能为 0  
[out] lpBytesReturned 实际收到的数据长度指针, 不能为 NULL

表 5.7 参数获取命令

dwCommand 宏定义	dwCommand 含义	宏定义值
NET_DVR_GET_DEVICECFG_V40	获取设备参数	1100
NET_DVR_GET_NETCFG_V30	获取网络参数	1000
NET_DVR_GET_TIMECFG	获取时间参数	118
NET_DVR_GET_USERCFG_V40	获取用户参数	6187
NET_DVR_GET_EXCEPTIONCFG_V40	获取异常参数	6177
NET_DVR_GET_NETCFG_MULTI	获取多网卡配置参数	1161
NET_DVR_GET_NETWORK_BONDING	获取多网卡 BONDING 参数	1254
NET_DVR_GET_NETCFG_OTHER	获取多路解码器 DNS 网络参数	244
NET_DVR_MATRIX_BIGSCREENCFG_GET	获取大屏拼接参数	1140
NET_DVR_GET_AUTO_REBOOT_CFG	获取自动重启参数(DS-65xxD)	1710
NET_DVR_GET_DEV_WORK_MODE	获取设备工作模式	9108
NET_DVR_GET_DEC_VCA_CFG	获取解码器智能报警参数	9130
NET_DVR_GET_WIN_ROAM_SWITCH_CFG	获取解码器窗口漫游开关参数	9224

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明: 不同的获取功能对应不同的结构体和命令号, 如表 5.8 所示。

表 5.8 获取设备参数

dwCommand 宏定义	通道号	lpOutBuffer 对应结构体
NET_DVR_GET_DEVICECFG_V40	无效	NET_DVR_DEVICECFG_V40
NET_DVR_GET_NETCFG_V30	无效	NET_DVR_NETCFG_V30
NET_DVR_GET_TIMECFG	无效	NET_DVR_TIME
NET_DVR_GET_USERCFG_V40	组号, 从 0 开始, 每组 32 个用户	NET_DVR_USER_V40
NET_DVR_GET_EXCEPTIONCFG_V40	组号, 从 0 开始, 每组 32 种异常	NET_DVR_EXCEPTION_V40

NET_DVR_GET_NETCFG_MULTI	无效	NET_DVR_NETCFG_MULTI
NET_DVR_GET_NETWORK_BONDING	无效	NET_DVR_NETWORK_BONDING
NET_DVR_GET_NETCFG_OTHER	无效	NET_DVR_NETCFG_OTHER
NET_DVR_MATRIX_BIGSCREENCFG_GET	大屏序号, 0~n(从能力集获取)	NET_DVR_BIGSCREENCFG
NET_DVR_GET_AUTO_REBOOT_CFG	无效	NET_DVR_AUTO_REBOOT_CFG
NET_DVR_GET_DEV_WORK_MODE	无效	NET_DVR_DEV_WORK_MODE
NET_DVR_GET_DEC_VCA_CFG	解码通道号	NET_DVR_VCA_ALARM_CFG
NET_DVR_GET_WIN_ROAM_SWITCH_CFG	无效	NET_DVR_WIN_ROAM_SWITCH_CFG

[返回目录](#)

## 5.6.2 设置设备的配置信息 **NET\_DVR\_SetDVRConfig**

函 数: BOOL NET\_DVR\_SetDVRConfig(LONG lUserID, DWORD dwCommand, LONG lChannel, LPVOID lpInBuffer, DWORD dwInBufferSize)

参 数: [in] lUserID 用户 ID 号, NET\_DVR\_Login\_V30 的返回值  
[in] dwCommand 设备配置命令, 参见表 5.9  
[in] lChannel 通道号, 不同的命令取值不同, 详见表 5.10。如果该参数无效, 置为 0xFFFFFFFF 即可  
[in] lpInBuffer 输入数据的缓冲指针, 详见表 5.10  
[in] dwInBufferSize 输入数据的缓冲长度(以字节为单位)  
表 5.9 参数设置命令

dwCommand 宏定义	dwCommand 含义	宏定义值
NET_DVR_SET_DEVICECFG_V40	设置设备参数	1101
NET_DVR_SET_NETCFG_V30	设置网络参数	1001
NET_DVR_SET_TIMECFG	设置时间参数	119
NET_DVR_SET_USERCFG_V40	设置用户参数	6188
NET_DVR_SET_EXCEPTIONCFG_V40	设置异常参数	6178
NET_DVR_SET_NETCFG_MULTI	设置多网卡配置参数	1162
NET_DVR_SET_NETWORK_BONDING	设置多网卡 BONDING 参数	1255
NET_DVR_SET_NETCFG_OTHER	设置多路解码器 DNS 网络参数	245
NET_DVR_MATRIX_BIGSCREENCFG_SET	设置大屏拼接参数	1141
NET_DVR_SET_AUTO_REBOOT_CFG	设置自动重启参数(DS-65xxD)	1171
NET_DVR_SET_DEV_WORK_MODE	设置设备工作模式	9109
NET_DVR_SET_DEC_VCA_CFG	设置解码器智能报警参数	9131
NET_DVR_SET_WIN_ROAM_SWITCH_CFG	设置解码器窗口漫游开关参数	9225

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明: 不同的获取功能对应不同的结构体和命令号, 如表 5.10 所示。

表 5.10 设置设备参数

dwCommand 宏定义	通道号	lpInBuffer 对应结构体
NET_DVR_SET_DEVICECFG_V40	无效	NET_DVR_DEVICECFG_V40
NET_DVR_SET_NETCFG_V30	无效	NET_DVR_NETCFG_V30
NET_DVR_SET_TIMECFG	无效	NET_DVR_TIME
NET_DVR_SET_USERCFG_V40	组号, 从 0 开始, 每组 32 个用户	NET_DVR_USER_V40
NET_DVR_SET_EXCEPTIONCFG_V40	组号, 从 0 开始, 每组 32 种异常	NET_DVR_EXCEPTION_V40
NET_DVR_SET_NETCFG_MULTI	无效	NET_DVR_NETCFG_MULTI
NET_DVR_SET_NETWORK_BONDING	无效	NET_DVR_NETWORK_BONDING
NET_DVR_SET_NETCFG_OTHER	无效	NET_DVR_NETCFG_OTHER
NET_DVR_MATRIX_BIGSCREENCFG_SET	大屏序号, 0~n(从能力集获取)	NET_DVR_BIGSCREENCFG
NET_DVR_SET_AUTO_REBOOT_CFG	无效	NET_DVR_AUTO_REBOOT_CFG
NET_DVR_SET_DEV_WORK_MODE	无效	NET_DVR_DEV_WORK_MODE
NET_DVR_SET_DEC_VCA_CFG	解码通道号	NET_DVR_VCA_ALARM_CFG
NET_DVR_SET_WIN_ROAM_SWITCH_CFG	无效	NET_DVR_WIN_ROAM_SWITCH_CFG

[返回目录](#)

### 5.6.3 批量获取配置信息 NET\_DVR\_GetDeviceConfig

- 函 数: BOOL NET\_DVR\_GetDeviceConfig(LONG lUserID, DWORD dwCommand, DWORD dwCount, LPVOID lpInBuffer, DWORD dwInBufferSize, LPVOID lpStatusList, LPVOID lpOutBuffer, DWORD dwOutBufferSize)
- 参 数:
- [in] lUserID 用户 ID 号, NET\_DVR\_Login\_V30 的返回值
  - [in] dwCommand 设备配置命令, 详见表 5.11
  - [in] dwCount 要设置的子设备个数
  - [in] lpInBuffer 配置条件缓冲区, 详见表 5.12
  - [in] dwInBufferSize 缓冲区长度
  - [out] lpStatusList 错误信息列表, 和要查询的监控点一一对应, 例如 lpStatusList[2] 就对应 lpInBuffer[2], 由用户分配内存, 每个错误信息为 4 个字节, 参数值: 0- 成功, 大于 0-失败
  - [out] lpOutBuffer 设备返回的参数内容 (详见表 5.12), 和要查询的监控点一一对应。如果某个监控点对应的 lpStatusList 信息为大于 0 值, 对应 lpOutBuffer 的内容就是无效的
  - [in] dwOutBufferSize 输出缓冲区大小

表 5.11 参数批量获取命令

dwCommand 宏定义	dwCommand 含义	宏定义值
NET_DVR_MATRIX_WALL_GET	获取电视墙中屏幕参数	9002
NET_DVR_WALLWIN_GET	电视墙中获取窗口参数	9003
NET_DVR_WALLSCENEPARAM_GET	电视墙中获取场景参数	9007

返回值: TRUE 表示成功, 但不代表每一个配置都成功, 哪一个成功, 对应查看 IpStatusList[n]值; FALSE 表示全部失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说明: 该接口是带有发送数据的批量获取监控点配置信息的通用接口。IpInBuffer 指定需要获取的 dwCount 个监控点信息, IpOutBuffer 保存获取到的 dwCount 个监控点的配置信息。全部获取时 dwCount 置为 0xffffffff, IpInBuffer 置为 NULL, dwInBufferSize 置为 0, IpStatusList 置为 NULL。不同的获取功能对应不同的结构体和命令号, 如表 5.12 所示。

表 5.12 批量获取参数

dwCommand 宏定义	IpInBuffer 对应结构体	IpOutBuffer 对应结构体
NET_DVR_MATRIX_WALL_GET	dwCount 个 (4 字节) 屏幕输出号	dwCount 个 NET_DVR_SINGLEWALLPARAM
NET_DVR_WALLWIN_GET	dwCount 个 (4 字节) 窗口号	dwCount 个 NET_DVR_WALLWINCFG
NET_DVR_WALLSCENEPARAM_GET	dwCount 个 (4 字节) 场景号	dwCount 个 NET_DVR_WALLSCENECFG

[返回目录](#)

## 5.6.4 批量设置配置信息 **NET\_DVR\_SetDeviceConfig**

函数: BOOL NET\_DVR\_SetDeviceConfig(LONG IUserID,DWORD dwCommand,DWORD dwCount,LPVOID IpInBuffer,DWORD dwInBufferSize,LPVOID IpStatusList, LPVOID IpInParamBuffer,DWORD dwInParamBufferSize)

参数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 的返回值  
[in] dwCommand 设备配置命令, 详见表 5.13  
[in] dwCount 要设置的子设备个数  
[in] IpInBuffer 配置条件缓冲区, 详见表 5.14  
[in] dwInBufferSize 缓冲区长度  
[out] IpStatusList 错误信息列表, 和要查询的监控点一一对应, 例如 IpStatusList[2] 就对应 IpInBuffer[2], 由用户分配内存, 每个错误信息为 4 个字节, 参数值: 0- 成功, 大于 0-失败  
[in] IpInParamBuffer 需要设置给设备的参数内容 (详见表 5.14), 和要查询的监控点一一对应。如果某个监控点对应的 IpStatusList 信息为大于 0 值, 表示对应的 IpInBuffer 设置失败, 为 0 则设置成功  
[in] dwInParamBufferSize 设置内容缓冲区大小

表 5.13 批量设置命令

dwCommand 宏定义	dwCommand 含义	宏定义值
NET_DVR_MATRIX_WALL_SET	设置电视墙中屏幕参数	9001
NET_DVR_WALLWIN_SET	电视墙中设置窗口参数	9004
NET_DVR_WALLSCENEPARAM_SET	电视墙中设置场景参数	9008

返回值: TRUE 表示成功, 但不代表每一个配置都成功, 哪一个成功, 对应查看 IpStatusList[n]值; FALSE 表示全部失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说明: 该接口是带有发送数据的批量设置监控点配置信息的通用接口。IpInBuffer 指定需要设置的信息, IpOutBuffer 保存将要设置的 dwCount 个监控点的配置信息。不同的获取功能对应不同的结构体和命令号, 如表 5.14 所示。

表 5.14 批量设置参数

dwCommand 宏定义	lpInBuffer 对应结构体	lpInParamBuffer 对应结构体
NET_DVR_MATRIX_WALL_SET	dwCount 个（4 字节）屏幕输出号	dwCount 个 NET_DVR_SINGLEWALLPARAM
NET_DVR_WALLWIN_SET	dwCount 个（4 字节）窗口号	dwCount 个 NET_DVR_WALLWINCFG
NET_DVR_WALLSCENEPARAM_SET	dwCount 个（4 字节）场景号	dwCount 个 NET_DVR_WALLSCENECFG

[返回目录](#)

## 5.7 解码通道相关

### 5.7.1 获取解码通道配置信息 **NET\_DVR\_MatrixGetDecChanCfg**

函 数: BOOL NET\_DVR\_MatrixGetDecChanCfg(LONG IUserID, DWORD dwDecChan, LPNET\_DVR\_MATRIX\_DECCHAN\_CONTROL lpInter)

参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 返回值  
[in] dwDecChan 解码通道号  
[out] lpInter 解码通道缩放控制参数, 详见结构体:  
NET\_DVR\_MATRIX\_DECCHAN\_CONTROL

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

### 5.7.2 配置解码通道 **NET\_DVR\_MatrixSetDecChanCfg**

函 数: BOOL NET\_DVR\_MatrixSetDecChanCfg(LONG IUserID, DWORD dwDecChan, LPNET\_DVR\_MATRIX\_DECCHAN\_CONTROL lpInter)

参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 返回值  
[in] dwDecChan 解码通道号  
[in] lpInter 解码通道缩放控制参数, 详见结构体:  
NET\_DVR\_MATRIX\_DECCHAN\_CONTROL

返回值: TRUE 表示成功, FALSE 表示失败。获取错误码调用 [NET\\_DVR\\_GetLastError](#)。

说 明:

[返回目录](#)

### 5.7.3 获取当前解码通道状态 **NET\_DVR\_MatrixGetDecChanStatus**

函 数: BOOL NET\_DVR\_MatrixGetDecChanStatus(LONG IUserID, DWORD dwDecChanNum, LPNET\_DVR\_MATRIX\_DEC\_CHAN\_STATUS lpInter)

参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 返回值  
[in] dwDecChanNum 解码通道  
[out] lpInter 解码通道状态信息, 详见结构体:

## NET\_DVR\_MATRIX\_DEC\_CHAN\_STATUS

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说明:** 获取当前解码通道状态, 包括解码状态、码流传输速率等。

[返回目录](#)

### 5.7.4 获取解码通道开关 **NET\_DVR\_MatrixGetDecChanEnable**

函数: BOOL NET\_DVR\_MatrixGetDecChanEnable(LONG IUserID, DWORD dwDecChanNum, LPDWORD lpdwEnable)

参数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 返回值  
[in] dwDecChanNum 解码通道  
[out] lpdwEnable 0 表示关闭, 1 表示打开

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说明:** 解码通道的解码开关, 用于控制解码通道的解码过程, 设置此开关为关时, 无论当前解码通道是处在动态解码还是轮巡过程, 都将停止解码, 生效后该通道处于黑屏状态; 设置开关为开时, 将恢复先前过程。此功能可与轮巡, 轮巡开关等配合使用。

[返回目录](#)

### 5.7.5 设置解码通道开关 **NET\_DVR\_MatrixSetDecChanEnable**

函数: BOOL NET\_DVR\_MatrixSetDecChanEnable (LONG IUserID, DWORD dwDecChanNum, DWORD dwEnable)

参数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 返回值  
[in] dwDecChanNum 解码通道  
[in] dwEnable 0 表示关闭, 1 表示打开

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说明:** 解码通道的解码开关, 用于控制解码通道的解码过程, 设置此开关为关时, 无论当前解码通道是处在动态解码还是轮巡过程, 都将停止解码, 生效后该通道处于黑屏状态; 设置开关为开时, 将恢复先前过程。此功能可与轮巡, 轮巡开关等配合使用。

[返回目录](#)

## 5.8 主动解码

### 5.8.1 启动动态解码 **NET\_DVR\_MatrixStartDynamic\_V41**

函数: BOOL NET\_DVR\_MatrixStartDynamic\_V41(LONG IUserID, DWORD dwDecChanNum, LPNET\_DVR\_PU\_STREAM\_CFG\_V41 lpDynamicInfo);

参数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 返回值  
[in] dwDecChanNum 解码通道



返回值: [in] lpDynamicInfo 动态解码参数, 详见结构体: NET\_DVR\_PU\_STREAM\_CFG\_V41  
TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说明:** 配置多路解码器某一解码通道连接前端设备的某一通道, 持续进行解码, 直到调用停止动态解码接口或者关闭解码通道解码开关。解码过程中若因出现网络中断等原因造成解码中断的情况, 多路解码器将自动向前端设备发起重连, 直到连接成功或者停止解码接口被调用, 其间该解码通道处于黑屏状态。

[返回目录](#)

## 5.8.2 停止动态解码 NET\_DVR\_MatrixStopDynamic

函数: BOOL NET\_DVR\_MatrixStopDynamic(LONG IUserID, DWORD dwDecChanNum)  
参数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 返回值  
[in] dwDecChanNum 解码通道  
返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说明:**

[返回目录](#)

## 5.8.3 获取轮巡解码通道 NET\_DVR\_MatrixGetLoopDecChanInfo\_V41

函数: BOOL NET\_DVR\_MatrixGetLoopDecChanInfo\_V41(LONG IUserID, DWORD dwDecChanNum, LPNET\_DVR\_MATRIX\_LOOP\_DECINFO\_V41 lpOuter)  
参数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 返回值  
[in] dwDecChanNum 解码通道  
[out] lpInter 轮巡的解码通道参数, 详见结构体:  
NET\_DVR\_MATRIX\_LOOP\_DECINFO\_V41  
返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说明:**

[返回目录](#)

## 5.8.4 设置轮巡解码通道 NET\_DVR\_MatrixSetLoopDecChanInfo\_V41

函数: BOOL NET\_DVR\_MatrixSetLoopDecChanInfo\_V41(LONG IUserID, DWORD dwDecChanNum, LPNET\_DVR\_MATRIX\_LOOP\_DECINFO\_V41 lpInter)  
参数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 返回值  
[in] dwDecChanNum 解码通道  
[in] lpInter 轮巡的解码通道参数, 详见结构体:  
NET\_DVR\_MATRIX\_LOOP\_DECINFO\_V41  
返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说明:** 设置多路解码器某一解码通道的轮巡解码参数, 每个解码通道最多可连接 64 路前端设备通道进



行循环解码，循环周期可以设置，参数设置成功后，若连接信息中的启用标志置为启用，则该通道即进入轮巡状态，开始循环解码，若全部置为不启用，则不进行循环解码。对循环解码的控制可以配合使用解码轮巡开关来实现，详细参看设置获取解码轮巡开关部分。

[返回目录](#)

### 5.8.5 获取解码通道轮巡开关 **NET\_DVR\_MatrixGetLoopDecChanEnable**

**函 数：** BOOL NET\_DVR\_MatrixGetLoopDecChanEnable(LONG lUserID, DWORD dwDecChanNum, LPDWORD lpdwEnable)

**参 数：** [in] lUserID                      用户 ID 号，NET\_DVR\_Login\_V30 返回值  
           [in] dwDecChanNum            解码通道  
           [out] lpdwEnable            0 表示关闭，1 表示打开

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 轮巡开关用于控制轮巡过程的启停，而不是控制解码的启停，当设置解码器当前正在轮巡的解码通道的轮巡开关为关时，该解码通道停止循环，停留在当前所连接的前端 dvr 通道继续解码，退为动态解码，当设置轮巡开关为开时，解码器恢复该解码通道的循环。

[返回目录](#)

### 5.8.6 设置解码通道轮巡开关 **NET\_DVR\_MatrixSetLoopDecChanEnable**

**函 数：** BOOL NET\_DVR\_MatrixSetLoopDecChanEnable(LONG lUserID, DWORD dwDecChanNum, DWORD dwEnable)

**参 数：** [in] lUserID                      用户 ID 号，NET\_DVR\_Login\_V30 返回值  
           [in] dwDecChanNum            解码通道  
           [in] dwEnable                0 表示关闭，1 表示打开

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 轮巡开关用于控制轮巡过程的启停，而不是控制解码的启停，当设置解码器当前正在轮巡的解码通道的轮巡开关为关时，该解码通道停止循环，停留在当前所连接的前端 dvr 通道继续解码，退为动态解码，当设置轮巡开关为开时，解码器恢复该解码通道的循环。

[返回目录](#)

### 5.8.7 获取所有解码通道轮巡开关 **NET\_DVR\_MatrixGetLoopDecEnable**

**函 数：** BOOL NET\_DVR\_MatrixGetLoopDecEnable(LONG lUserID, LPDWORD lpdwEnable)

**参 数：** [in] lUserID                      用户 ID 号，NET\_DVR\_Login\_V30 返回值  
           [out] lpdwEnable            按位表示，0 表示关闭，1 表示打开

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** lpdwEnable 按位表示，例如 lpdwEnable&0x1=1 和 lpdwEnable&0x4=1，其他位为 0，表示第 1、3 解码通道开启轮巡，其他解码通道关闭轮巡。

[返回目录](#)

### 5.8.8 获取当前解码通道信息 **NET\_DVR\_MatrixGetDecChanInfo\_V41**

**函 数：** BOOL NET\_DVR\_MatrixGetDecChanInfo\_V41(LONG IUserID, DWORD dwDecChanNum, LPNET\_DVR\_MATRIX\_DEC\_CHAN\_INFO\_V41 lpOuter)

**参 数：** [in] IUserID 用户 ID 号，NET\_DVR\_Login\_V30 的返回值  
[in] dwDecChanNum 解码通道  
[out] lpOuter 解码通道信息，详见结构体：  
NET\_DVR\_MATRIX\_DEC\_CHAN\_INFO\_V41

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 获取当前解码通道信息，包括前端设备通道信息、取流模式等。

[返回目录](#)

### 5.8.9 远程控制文件回放解码 **NET\_DVR\_RemoteControl**

**函 数：** BOOL NET\_DVR\_RemoteControl(LONG IUserID, DWORD dwCommand, LPVOID lpInBuffer, DWORD dwInBufferSize)

**参 数：** [in] IUserID 用户 ID 号，NET\_DVR\_Login\_V30 的返回值  
[in] dwCommand 控制命令，详见表 5.15  
[in] lpInBuffer 输入参数，具体内容跟控制命令相关，详见表 5.15  
[in] dwInBufferSize 输入参数长度

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 不同的控制功能对应不同的命令号，同时 lpInBuffer 对应不同的结构体，如表 5.15 所示。

表 5.15 回放解码控制

dwCommand 宏定义	宏定义值	控制功能	lpInBuffer 对应结构体
NET_DVR_DEC_PLAY_REMOTE_FILE	9032	解码播放远程文件	NET_DVR_MATRIX_DEC_REMOTE_PLAY_EX

[返回目录](#)

### 5.8.10 远程回放文件解码配置 **NET\_DVR\_MatrixSetRemotePlay**

**函 数：** BOOL NET\_DVR\_MatrixSetRemotePlay(LONG IUserID, DWORD dwDecChanNum, LPNET\_DVR\_MATRIX\_DEC\_REMOTE\_PLAY lpInter)

**参 数：** [in] IUserID 用户 ID 号，NET\_DVR\_Login\_V30 的返回值  
[in] dwDecChanNum 解码通道  
[in] lpInter 远程回放文件参数，详见结构体：  
NET\_DVR\_MATRIX\_DEC\_REMOTE\_PLAY

**返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 调用该接口完成配置之后还需要调用 NET\_DVR\_MatrixSetRemotePlayControl (NET\_DVR\_PLAYSTART) 开启回放。

[返回目录](#)

### 5.8.11 远程文件回放控制 **NET\_DVR\_MatrixSetRemotePlayControl**

函 数: BOOL NET\_DVR\_MatrixSetRemotePlayControl(LONG IUserID, DWORD dwDecChanNum, DWORD dwControlCode, DWORD dwInValue, DWORD \*lpOutValue)

参 数:

[in] IUserID	用户 ID 号, NET_DVR_Login_V30 的返回值
[in] dwDecChanNum	解码通道
[in] dwControlCode	控制命令, 具体定义见表 5.16
[in] dwInValue	设置参数
[out] lpOutValue	获取到的参数指针

表 5.16 回放控制命令

dwControlCode 宏定义	宏定义值	含义
NET_DVR_PLAYSTART	1	开始播放
NET_DVR_PLAYSTOP	2	停止播放
NET_DVR_PLAYPAUSE	3	暂停播放
NET_DVR_PLAYRESTART	4	恢复播放
NET_DVR_PLAYFAST	5	快放
NET_DVR_PLAYSLOW	6	慢放
NET_DVR_PLAYNORMAL	7	正常速度播放
NET_DVR_PLAYSTARTAUDIO	9	打开声音
NET_DVR_PLAYSTOPAUDIO	10	关闭声音
NET_DVR_PLAYSETPOS	12	改变文件回放的进度

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明: dwInValue 和 lpOutValue 参数根据不同的命令号决定是否输入和输出, 如当进行 NET\_DVR\_PLAYSETPOS 命令操作时, 需要对 dwInValue 参数进行赋值。

[返回目录](#)

### 5.8.12 获取回放状态 **NET\_DVR\_MatrixGetRemotePlayStatus**

函 数: BOOL NET\_DVR\_MatrixGetRemotePlayStatus(LONG IUserID, DWORD dwDecChanNum, LPNET\_DVR\_MATRIX\_DEC\_REMOTE\_PLAY\_STATUS lpOuter)

参 数:

[in] IUserID	用户 ID 号, NET_DVR_Login_V30 的返回值
[in] dwDecChanNum	解码通道
[out] lpOuter	回放状态, 详见结构体: NET_DVR_MATRIX_DEC_REMOTE_PLAY_STATUS

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明: 多路解码器远程连接前端设备完成按文件名/时间回放文件, 并能够获取相应回放状态, 进行回

放控制等。按时间回放不支持进度控制；由于回放控制命令是转发实现，存在一定的延迟，因此，回放控制命令不宜过于频繁调用，具体视网络状况而定，当把获取的状态作为客户端处理依据时应考虑网络转发的延迟因素；按时间回放时，获取回放状态所得到的文件信息是当前播放的单个片段的信息，并非整个时间范围内全部片段的信息，判断播放是否结束使用 NET\_DVR\_MATRIX\_DEC\_REMOTE\_PLAY\_STATUS 结构中的 dwCurDataType 成员。

[返回目录](#)

## 5.9 被动解码

### 5.9.1 启动被动解码 NET\_DVR\_MatrixStartPassiveDecode

函 数： LONG NET\_DVR\_MatrixStartPassiveDecode(LONG IUserID, DWORD dwDecChanNum, LPNET\_DVR\_MATRIX\_PASSIVEMODE lpPassiveMode)

参 数： [in] IUserID 用户 ID 号，NET\_DVR\_Login\_V30 的返回值  
[in] dwDecChanNum 解码通道  
[in] lpPassiveMode 被动解码参数，详见结构体：NET\_DVR\_MATRIX\_PASSIVEMODE

返回值： -1 表示失败，其他值作为 NET\_DVR\_MatrixSendData 等函数的参数。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明：

[返回目录](#)

### 5.9.2 向被动解码通道发送数据 NET\_DVR\_MatrixSendData

函 数： BOOL NET\_DVR\_MatrixSendData(LONG IPassiveHandle, char \*pSendBuf, DWORD dwBufSize)

参 数： [in] IPassiveHandle NET\_DVR\_MatrixStartPassiveDecode 的返回值  
[in] pSendBuf 发送数据缓冲区  
[in] dwBufSize 发送缓冲区大小，需小于 256K 字节，推荐 30K 字节

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明：

[返回目录](#)

### 5.9.3 停止被动解码 NET\_DVR\_MatrixStopPassiveDecode

函 数： BOOL NET\_DVR\_MatrixStopPassiveDecode(LONG IPassiveHandle)

参 数： [in] IPassiveHandle NET\_DVR\_MatrixStartPassiveDecode 的返回值

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明：

[返回目录](#)

### 5.9.4 获取被动解码状态 **NET\_DVR\_MatrixGetPassiveDecodeStatus**

函 数: LONG NET\_DVR\_MatrixGetPassiveDecodeStatus(LONG lPassiveHandle)  
 参 数: [in] lPassiveHandle NET\_DVR\_MatrixStartPassiveDecode 的返回值  
 返回值: -1 表示失败, 1-发送成功, 2-暂停发送, 3-恢复发送, 4-错误, 5-心跳信息。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。  
 说 明:

[返回目录](#)

### 5.9.5 被动解码播放控制 **NET\_DVR\_MatrixPassiveDecodeControl**

函 数: BOOL NET\_DVR\_MatrixPassiveDecodeControl( LONG lUserID, DWORD dwDecChanNum, LPNET\_DVR\_PASSIVEDECODE\_CONTROL lpInter)  
 参 数: [in] lUserID 用户 ID 号, NET\_DVR\_Login\_V30 的返回值  
 [in] dwDecChanNum 解码通道号  
 [in] lpInter 解码控制参数, 详见结构体: NET\_DVR\_PASSIVEDECODE\_CONTROL  
 返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。  
 说 明:

[返回目录](#)

## 5.10 LOGO 上传和显示控制

### 5.10.1 LOGO 上传 **NET\_DVR\_UploadLogo**

函 数: BOOL NET\_DVR\_UploadLogo(LONG lUserID,DWORD dwDecChanNum, LPNET\_DVR\_DISP\_LOGOCFG lpDispLogoCfg, char \*sLogoBuffer)  
 参 数: [in] lUserID NET\_DVR\_Login\_V30 的返回值  
 [in] dwDecChanNum 解码通道号  
 [in] lpDispLogoCfg LOGO 的参数, 详见结构体: NET\_DVR\_DISP\_LOGOCFG  
 [in] sLogoBuffer LOGO 数据缓冲区, 最大 100K, 图像的宽和高必须是 32 的倍数  
 返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。  
 说 明:

[返回目录](#)

### 5.10.2 LOGO 显示控制 **NET\_DVR\_LogoSwitch**

函 数: BOOL NET\_DVR\_LogoSwitch(LONG lUserID, DWORD dwDecChan, DWORD dwLogoSwitch)  
 参 数: [in] lUserID NET\_DVR\_Login\_V30 的返回值  
 [in] dwDecChan 解码通道号  
 [in] dwLogoSwitch 开关命令, 宏定义详见表 5.17

表 5.17 LOGO 控制命令

宏定义	宏定义值	含义
NET_DVR_SHOWLOGO	1	显示 LOGO
NET_DVR_HIDELOGO	2	隐藏 LOGO

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明：

[返回目录](#)

## 5.11 场景操作

### 5.11.1 场景切换控制 **NET\_DVR\_MatrixSceneControl**

函 数： BOOL NET\_DVR\_MatrixSceneControl(LONG IUserID, DWORD dwSceneNum, DWORD dwCmd, DWORD dwCmdParam)

参 数： [in] IUserID NET\_DVR\_Login\_V30 的返回值

[in] dwSceneNum 场景号，能力集中获取

[in] dwCmd 命令类型：

- 1- 场景模式切换（如果要切换的是当前场景，则不进行切换）；
- 2- 初始化场景（将此场景的配置清空，如果是当前场景，则同时对当前场景进行清屏操作）；
- 3- 强制切换（无论是否是当前场景，强制切换）；
- 4- 保存场景

[in] dwCmdParam 命令参数，保留

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明：

[返回目录](#)

### 5.11.2 获取当前正在使用的场景模式 **NET\_DVR\_MatrixGetCurrentSceneMode**

函 数： BOOL NET\_DVR\_MatrixGetCurrentSceneMode(LONG IUserID, DWORD \*dwSceneNum)

参 数： [in] IUserID NET\_DVR\_Login\_V30 的返回值

[out] dwSceneNum 场景号

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明：

[返回目录](#)

## 5.12 远程控制

### 5.12.1 远程控制 **NET\_DVR\_RemoteControl**

- 函 数：** BOOL NET\_DVR\_RemoteControl(LONG IUserID, DWORD dwCommand, LPVOID lpInBuffer, DWORD dwInBufferSize)
- 参 数：** [in]IUserID 用户 ID 号，NET\_DVR\_Login\_V30 的返回值  
[in]dwCommand 控制命令，详见表 5.18  
[in]lpInBuffer 输入参数，具体内容跟控制命令相关，详见表 5.18  
[in]dwInBufferSize 输入参数长度
- 返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。
- 说 明：** 不同的控制功能对应不同的命令号，同时 lpInBuffer 对应不同的结构体，如表 5.18 所示。

表 5.18 窗口控制命令

dwCommand 宏定义	宏定义值	控制功能	lpInBuffer 对应结构体
NET_DVR_CLOSE_ALL_WND	9016	电视墙关闭所有窗口	NULL
NET_DVR_SWITCH_WIN_TOP	9017	窗口置顶	4 字节，窗口号
NET_DVR_SWITCH_WIN_BOTTOM	9018	窗口置底	4 字节，窗口号

[返回目录](#)

## 5.13 透明通道

### 5.13.1 获取透明通道信息 **NET\_DVR\_MatrixGetTranInfo\_V30**

- 函 数：** BOOL NET\_DVR\_MatrixGetTranInfo\_V30(LONG IUserID, LPNET\_DVR\_MATRIX\_TRAN\_CHAN\_CONFIG\_V30 lpTranInfo)
- 参 数：** [in] IUserID NET\_DVR\_Login\_V30 的返回值  
[out] lpTranInfo 透明通道参数，详见结构体：  
NET\_DVR\_MATRIX\_TRAN\_CHAN\_CONFIG\_V30
- 返回值：** TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。
- 说 明：** 该透明通道配置是配置解码器与前端设备之间建立网络透明通道，而不是客户端与解码器之间建立透明通道，多路解码器本身不支持与客户端建立 232 透明通道和 485 透明通道，多路解码器本地串口只能作为串口控制台接入（通过 RS232）或是控制键盘等输入性设备接入（通过 RS232/RS485）。

[返回目录](#)

### 5.13.2 设置透明通道参数 **NET\_DVR\_MatrixSetTranInfo\_V30**

- 函 数：** BOOL NET\_DVR\_MatrixSetTranInfo\_V30(LONG IUserID, LPNET\_DVR\_MATRIX\_TRAN\_CHAN\_CONFIG\_V30 lpTranInfo)



**参 数:** [in] IUserID NET\_DVR\_Login\_V30 的返回值  
[in] IpTranInfo 透明通道参数, 详见结构体:  
NET\_DVR\_MATRIX\_TRAN\_CHAN\_CONFIG\_V30

**返回值:** TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:** 目前多路解码器支持最多建立 64 条透明通道, 包括 232 透明通道和 485 透明通道, 但其中只有最多一条 232 全双工透明通道和最多一条 485 全双工透明通道, 可以不设置全双工透明通道。

[返回目录](#)

## 5.14 设备状态

### 5.14.1 获取解码设备状态 NET\_DVR\_MatrixGetDeviceStatus\_V41

**函 数:** BOOL NET\_DVR\_MatrixGetDeviceStatus\_V41(LONG IUserID,  
LPNET\_DVR\_DECODER\_WORK\_STATUS\_V41 lpDecoderCfg)

**参 数:** [in] IUserID NET\_DVR\_Login\_V30 的返回值  
[out] lpDecoderCfg 设备状态参数, 详见结构体:  
NET\_DVR\_DECODER\_WORK\_STATUS\_V41

**返回值:** TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:**

[返回目录](#)

### 5.14.2 获取设备运行状态 NET\_DVR\_GetDeviceStatus

**函 数:** BOOL NET\_DVR\_GetDeviceStatus(LONG IUserID, DWORD dwCommand, DWORD dwCount, LPVOID lpInBuffer, DWORD dwInBufferSize, LPVOID lpStatusList, LPVOID lpOutBuffer, DWORD dwOutBufferSize)

**参 数:** [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 返回值  
[in] dwCommand 设备配置命令, 详见表 5.19  
[in] dwCount 要设置的个数, 设为 1  
[in] lpInBuffer 配置条件缓冲区, 详见表 5.20  
[in] dwInBufferSize 缓冲区长度  
[out] lpStatusList 错误信息列表, 和要查询的监控点一一对应, 例如  
lpStatusList[2]就对应 lpInBuffer[2], 由用户分配内存, 每个  
错误信息为 4 个字节(1 个 32 位无符号整数值), 参数值:  
0- 成功, 大于 0- 失败  
[out] lpOutBuffer 设备返回的参数内容(详见表 5.20), 和要查询的监控  
点一一对应。如果某个监控点对应的 lpStatusList 信息为大  
于 0 值, 对应 lpOutBuffer 的内容就是无效的  
[in] dwOutBufferSize 输出缓冲区大小



表 5.19 状态获取命令

dwCommand 宏定义	dwCommand 含义	宏定义值
NET_DVR_GET_DEVICE_RUN_STATUS	获取设备窗口状态	1618
NET_DVR_GET_DEC_CHAN_STATUS	获取解码通道解码状态	9113
NET_DVR_GET_DISP_CHAN_STATUS	获取显示通道状态	9114
NET_DVR_GET_ALARMIN_STATUS	获取报警输入状态	9115
NET_DVR_GET_ALARMOUT_STATUS	获取报警输出状态	9116
NET_DVR_GET_AUDIO_CHAN_STATUS	获取语音对讲状态	9117

返回值： TRUE 表示成功，但不代表每一个配置都成功，哪一个成功，对应查看 IpStatusList[n]值；FALSE 表示全部失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 该接口是带有发送数据的批量获取设备状态信息的通用接口。全部获取时 dwCount 置为 0xffffffff，IpInBuffer 置为 NULL，dwInBufferSize 置为 0，IpStatusList 置为 NULL。IpOutBuffer 前面 4 个字节为个数(N)，后面为设备返回的 N 个信息内容（按通道号 1~N 排列），如果设置的 IpOutBuffer 缓冲区不足，仅返回部分信息，可以根据返回的个数（前 4 字节的值）重新获取。  
不同的获取功能对应不同的结构体和命令号，如表 5.20 所示。

表 5.20 获取设备状态

dwCommand 宏定义	IpInBuffer 对应结构体	IpOutBuffer 对应结构体
NET_DVR_GET_DEVICE_RUN_STATUS	dwCount 个 NET_DVR_WALLWIN_INFO	dwCount 个 NET_DVR_WALL_WIN_STATUS
NET_DVR_GET_DEC_CHAN_STATUS	dwCount 个 4 字节解码通道号	dwCount 个 NET_DVR_MATRIX_CHAN_STATUS
NET_DVR_GET_DISP_CHAN_STATUS	dwCount 个 4 字节显示通道号	dwCount 个 NET_DVR_DISP_CHAN_STATUS_V41
NET_DVR_GET_ALARMIN_STATUS	dwCount 个 4 字节报警输入通道号	dwCount 个 4 字节状态值(0-没有报警，1-有报警)
NET_DVR_GET_ALARMOUT_STATUS	dwCount 个 4 字节报警输出通道号	dwCount 个 4 字节状态值(0-没有报警，1-有报警)
NET_DVR_GET_AUDIO_CHAN_STATUS	dwCount 为 1，4 字节语音对讲通道号	1 个 4 字节状态(0-未开启，1-开启)

[返回目录](#)

## 5.15 布防、撤防

### 设置报警等信息上传的回调函数

#### 5.15.1 注册报警信息回调函数 **NET\_DVR\_SetDVRMessageCallBack\_V31**

函 数： BOOL NET\_DVR\_SetDVRMessageCallBack\_V31(MSGCallBack\_V31 fMessageCallBack, void \*pUser)

参 数： [in]fMessageCallBack 报警信息回调函数

[in]pUser 用户数据

```
typedef BOOL (CALLBACK * MSGCallBack_V31)(LONG ICommand, NET_DVR_ALARMER *pAlarmer,
char *pAlarmInfo, DWORD dwBufLen, void *pUser)
```

[out]ICommand 上传的消息类型，详见表 5.21

[out]pAlarmer 报警设备信息，详见结构体：NET\_DVR\_ALARMER

[out]pAlarmInfo	报警信息，详见表 5.22
[out]dwBufLen	报警信息缓存大小
[out]pUser	用户数据

表 5.21 报警信息类型

ICommand 宏定义	宏定义值	含义
COMM_ALARM_DEC_VCA	0x5010	解码器智能解码报警上传

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说明： 该接口中回调函数的第一个参数（ICommand）和第三个参数（pAlarmInfo）是密切关联的，其关系见表 5.22。

表 5.22 报警信息结构

消息类型（ICommand）	上传内容	pAlarmInfo 对应的结构体
COMM_ALARM_DEC_VCA	解码器智能解码报警信息	NET_DVR_DEC_VCA_ALARM

[返回目录](#)

## 布防撤防

### 5.15.2 建立报警上传通道，获取报警等信息 **NET\_DVR\_SetupAlarmChan\_V41**

函数： LONG NET\_DVR\_SetupAlarmChan\_V41(LONG IUserID, LPNET\_DVR\_SETUPALARM\_PARAM lpSetupParam)

参数： [in] IUserID NET\_DVR\_Login\_V40 的返回值  
[in] lpSetupParam 报警布防参数，详见结构体：NET\_DVR\_SETUPALARM\_PARAM

返回值： -1 表示失败，其他值作为 NET\_DVR\_CloseAlarmChan\_V30 函数的句柄参数。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说明： 使用该接口支持上传 V3.0 以上版本支持的设备的报警结构。  
启动布防前，需要调用注册回调函数的接口（如 [NET\\_DVR\\_SetDVRMessageCallBack\\_V30](#)）才能获取到上传的报警等信息。

[返回目录](#)

### 5.15.3 撤销报警上传通道 **NET\_DVR\_CloseAlarmChan\_V30**

函数： BOOL NET\_DVR\_CloseAlarmChan\_V30(LONG IAlarmHandle)

参数： [in] IAlarmHandle NET\_DVR\_SetupAlarmChan\_V41 的返回值

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说明：

[返回目录](#)

## 5.16 监听报警

### 5.16.1 启动监听，接收设备主动上传的报警等信息 **NET\_DVR\_StartListen\_V30**

函 数： LONG NET\_DVR\_StartListen\_V30(char \*sLocalIP, WORD wLocalPort, MSGCallBack DataCallback, void\* pUserData = NULL)

参 数： [in]sLocalIP                      PC 机本地 IP 地址，可以置为 NULL  
 [in]wLocalPort                      PC 本地监听端口号。由用户设置，必须和设备端设置的一致  
 [in]DataCallback                      回调函数  
 [in]pUserData                      用户数据

```
typedef void(CALLBACK *MSGCallBack)(LONG ICommand,NET_DVR_ALARMER *pAlarmer,char
*pAlarmInfo,DWORD dwBufLen,void *pUser)
```

[out]ICommand                      上传的消息类型，详见表 5.23  
 [out]pAlarmer                      报警设备信息，详见结构体：NET\_DVR\_ALARMER  
 [out]pAlarmInfo                      报警信息，详见表 5.24  
 [out]dwBufLen                      报警信息缓存大小  
 [out]pUser                      用户数据

表 5.23 报警信息类型

ICommand 宏定义	宏定义值	含义
COMM_ALARM_DEC_VCA	0x5010	解码器智能解码报警上传

返回值： -1 表示失败，其他值作为 NET\_DVR\_CloseAlarmChan\_V30 函数的句柄参数。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明： 该接口中回调函数的第一个参数（ICommand）和第三个参数（pAlarmInfo）是密切关联的，其关系见表 5.24。

- SDK 最大能支持 512 路监听。
- 要使 PC 能够收到设备主动发过来的报警等信息，必须将设备的网络配置中的“远程管理主机地址”或者“远程报警主机地址”设置成 PC 机的 IP 地址（与接口中的 sLocalIP 参数一致），“远程管理主机端口号”或者“远程报警主机端口号”设置成 PC 机的监听端口号（与接口中的 wLocalPort 参数一致）。
- 该接口中的回调函数优先级高于其他回调函数，即设置了该接口中的回调函数，其他回调函数将接收不到报警信息。

表 5.24 报警信息结构

消息类型（ICommand）	上传内容	pAlarmInfo 对应的结构体
COMM_ALARM_DEC_VCA	解码器智能解码报警信息	NET_DVR_DEC_VCA_ALARM

[返回目录](#)

### 5.16.2 停止监听（支持多线程） **NET\_DVR\_StopListen\_V30**

函 数： BOOL NET\_DVR\_StopListen\_V30(LONG IListenHandle)

参 数： [in]IListenHandle                      监听句柄，NET\_DVR\_StartListen\_V30 的返回值

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通

过错误码判断出错原因。

**说 明：**

[返回目录](#)

## 5.17 IPC 协议列表获取

### 5.17.1 获取设备支持的 IPC 协议表 **NET\_DVR\_GetIPCProtoList**

函 数： BOOL NET\_DVR\_GetIPCProtoList(LONG IUserID, LPNET\_DVR\_IPC\_PROTO\_LIST lpProtoList)

参 数： [in]IUserID                                NET\_DVR\_Login\_V30 的返回值  
[out]lpProtoList                                IPC 协议列表结构，详见结构体：NET\_DVR\_IPC\_PROTO\_LIST

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 该接口用于获取当前设备所支持的外接 IPC 的协议。

[返回目录](#)

## 5.18 设备维护管理

### 远程升级

#### 5.18.1 设置远程升级时网络环境 **NET\_DVR\_SetNetworkEnvironment**

函 数： BOOL NET\_DVR\_SetNetworkEnvironment(DWORD dwEnvironmentLevel)

参 数： [in] dwEnvironmentLevel                网络环境级别

```
enum{
    LOCAL_AREA_NETWORK = 0,  //局域网环境
    WIDE_AREA_NETWORK      //广域网环境
}
```

返回值： TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

**说 明：** 接口中的网络环境级别参数分为两类，  
**LOCAL\_AREA\_NETWORK** 表示局域网环境(网络环境好，通讯流畅)；  
**WIDE\_AREA\_NETWORK** 表示广域网环境(网络环境差，易阻塞)。  
在调用远程升级接口之前，可以通过此接口适应不同的升级环境。

[返回目录](#)

#### 5.18.2 远程升级 **NET\_DVR\_Upgrade**

函 数： LONG NET\_DVR\_Upgrade(LONG IUserID, char \*sFileName)

参 数： [in] IUserID                                NET\_DVR\_Login\_V30 的返回值

[in] sFileName                      升级的文件路径（包括文件名）。路径长度和操作系统有关，sdk 不做限制，windows 默认路径长度小于等于 256 字节（包括文件名在内）。

返回值:    -1 表示失败，其他值作为 NET\_DVR\_GetUpgradeState 等函数的参数。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明:

[返回目录](#)

### 5.18.3 获取远程升级的进度 **NET\_DVR\_GetUpgradeProgress**

函 数:    int NET\_DVR\_GetUpgradeProgress(LONG IUpgradeHandle)

参 数:    [in] IUpgradeHandle                      NET\_DVR\_Upgrade 的返回值

返回值:    -1 表示失败，0~100 表示升级进度。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明:

[返回目录](#)

### 5.18.4 获取远程升级的状态 **NET\_DVR\_GetUpgradeState**

函 数:    int NET\_DVR\_GetUpgradeState(LONG IUpgradeHandle)

参 数:    [in] IUpgradeHandle                      NET\_DVR\_Upgrade 的返回值

返回值:    -1 表示失败，其他值定义：1- 升级成功；2- 正在升级；3- 升级失败；4- 网络断开，状态未知；5- 升级文件语言版本不匹配。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明:

[返回目录](#)

### 5.18.5 关闭远程升级句柄，释放资源 **NET\_DVR\_CloseUpgradeHandle**

函 数:    BOOL NET\_DVR\_CloseUpgradeHandle(LONG IUpgradeHandle)

参 数:    [in] IUpgradeHandle                      NET\_DVR\_Upgrade 的返回值

返回值:    TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明:

[返回目录](#)

## 恢复设备默认参数

### 5.18.6 恢复设备默认参数 **NET\_DVR\_RestoreConfig**

函 数:    BOOL NET\_DVR\_RestoreConfig(LONG IUserID)

**参 数:** [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 的返回值

**返回值:** TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:**

[返回目录](#)

### 5.18.7 完全恢复出厂默认参数 **NET\_DVR\_RemoteControl**

**函 数:** BOOL NET\_DVR\_RemoteControl(LONG IUserID, DWORD dwCommand, LPVOID lpInBuffer, DWORD dwInBufferSize)

**参 数:** [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 返回值  
[in] dwCommand 控制命令, 详见表 5.25  
[in] lpInBuffer 输入参数, 跟控制命令相关, 详见列表  
[in] dwInBufferSize 输入参数长度

**返回值:** TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:** 不同的控制功能对应不同的命令号, 同时 lpInBuffer 对应不同的结构体, 如表 5.25 所示。

表 5.25 远程控制命令

dwCommand 宏定义	宏定义值	控制功能	lpInBuffer 对应结构体
NET_DVR_COMPLETE_RESTORE_CTRL	3420	设置完全恢复出厂值	NET_DVR_COMPLETE_RESTORE_INFO

[返回目录](#)

## 导入/导出配置文件

### 5.18.8 导出配置文件 **NET\_DVR\_GetConfigFile\_V30**

**函 数:** BOOL NET\_DVR\_GetConfigFile\_V30(LONG IUserID, char \*sOutBuffer, DWORD dwOutSize, DWORD \*pReturnSize)

**参 数:** [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 的返回值  
[out] sOutBuffer 存放配置参数的缓冲区  
[in] dwOutSize 缓冲区大小  
[out] pReturnSize 实际获得的缓冲区大小

**返回值:** TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

**说 明:** 当 sOutBuffer = NULL、dwOutSize = 0 且 pReturnSize != NULL 时用于获取参数配置文件的所需的缓冲区长度; 当 sOutBuffer != NULL 且 dwOutSize != 0 时用于获取参数配置文件的所需的缓冲区内内容。

[返回目录](#)

### 5.18.9 导出配置文件 **NET\_DVR\_GetConfigFile**

函 数: BOOL NET\_DVR\_GetConfigFile(LONG IUserID, char \*sFileName)

参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 的返回值  
[in] sFileName 存放保存配置文件的文件路径 (二进制文件)

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

### 5.18.10 导入配置文件 **NET\_DVR\_SetConfigFile\_EX**

函 数: BOOL NET\_DVR\_SetConfigFile\_EX(LONG IUserID, char \*sInBuffer, DWORD dwInSize)

参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 的返回值  
[in] sInBuffer 存放配置参数的缓冲区  
[in] dwInSize 缓冲区大小

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

### 5.18.11 导入配置文件 **NET\_DVR\_SetConfigFile**

函 数: BOOL NET\_DVR\_SetConfigFile(LONG IUserID, char \*sFileName)

参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 的返回值  
[in] sFileName 存放保存配置文件的文件路径 (二进制文件)

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

## 5.19 关机和重启

### 5.19.1 重启设备 **NET\_DVR\_RebootDVR**

函 数: BOOL NET\_DVR\_RebootDVR(LONG IUserID)

参 数: [in] IUserID 用户 ID 号, NET\_DVR\_Login\_V30 的返回值

返回值: TRUE 表示成功, FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码, 通过错误码判断出错原因。

说 明:

[返回目录](#)

### 5.19.2 关闭设备 **NET\_DVR\_ShutDownDVR**

函 数:     BOOL   NET\_DVR\_ShutDownDVR(LONG lUserID)

参 数:     [in] lUserID                    用户 ID 号，NET\_DVR\_Login\_V30 的返回值

返回值:    TRUE 表示成功，FALSE 表示失败。接口返回失败请调用 [NET\\_DVR\\_GetLastError](#) 获取错误码，通过错误码判断出错原因。

说 明:

[返回目录](#)



## 6 错误代码及说明

错误名称	错误值	说明
NET_DVR_NOERROR	0	没有错误。
NET_DVR_PASSWORD_ERROR	1	用户名密码错误。注册时输入的用户名或者密码错误。
NET_DVR_NOENOUGHPRI	2	权限不足。该注册用户没有权限执行当前对设备的操作，可以与远程用户参数配置做对比。
NET_DVR_NOINIT	3	SDK 未初始化。
NET_DVR_CHANNEL_ERROR	4	通道号错误。设备没有对应的通道号。
NET_DVR_OVER_MAXLINK	5	连接到设备的用户个数超过最大。
NET_DVR_VERSIONNOMATCH	6	版本不匹配。SDK 和设备的版本不匹配。
NET_DVR_NETWORK_FAIL_CONNECT	7	连接设备失败。设备不在线或网络原因引起的连接超时等。
NET_DVR_NETWORK_SEND_ERROR	8	向设备发送失败。
NET_DVR_NETWORK_RECV_ERROR	9	从设备接收数据失败。
NET_DVR_NETWORK_RECV_TIMEOUT	10	从设备接收数据超时。
NET_DVR_NETWORK_ERRORDATA	11	传送的数据有误。发送给设备或者从设备接收到的数据错误，如远程参数配置时输入设备不支持的值。
NET_DVR_ORDER_ERROR	12	调用次序错误。
NET_DVR_OPERNOPERMIT	13	无此权限。
NET_DVR_COMMANDTIMEOUT	14	设备命令执行超时。
NET_DVR_ERRORSERIALPORT	15	串口号错误。指定的设备串口号不存在。
NET_DVR_ERRORALARMPORT	16	报警端口错误。指定的设备报警输出端口不存在。
NET_DVR_PARAMETER_ERROR	17	参数错误。SDK 接口中给入的输入或输出参数为空。
NET_DVR_CHAN_EXCEPTION	18	设备通道处于错误状态
NET_DVR_NODISK	19	设备无硬盘。当设备无硬盘时，对设备的录像文件、硬盘配置等操作失败。
NET_DVR_ERRORDISKNUM	20	硬盘号错误。当对设备进行硬盘管理操作时，指定的硬盘号不存在时返回该错误。
NET_DVR_DISK_FULL	21	设备硬盘满。
NET_DVR_DISK_ERROR	22	设备硬盘出错
NET_DVR_NOSUPPORT	23	设备不支持。
NET_DVR_BUSY	24	设备忙。
NET_DVR_MODIFY_FAIL	25	设备修改不成功。
NET_DVR_PASSWORD_FORMAT_ERROR	26	密码输入格式不正确
NET_DVR_DISK_FORMATING	27	硬盘正在格式化，不能启动操作。
NET_DVR_DVRNORESOURCE	28	设备资源不足。
NET_DVR_DVROPRATEFAILED	29	设备操作失败。
NET_DVR_OPENHOSTSOUND_FAIL	30	语音对讲、语音广播操作中采集本地音频或打开音频输出失败。
NET_DVR_DVRVOICEOPENED	31	设备语音对讲被占用。
NET_DVR_TIMEINPUTERROR	32	时间输入不正确。
NET_DVR_NOSPECFILE	33	回放时设备没有指定的文件。
NET_DVR_CREATEFILE_ERROR	34	创建文件出错。本地录像、保存图片、获取配置文件和远

		程下载录像时创建文件失败。
NET_DVR_FILEOPENFAIL	35	打开文件出错。设置配置文件、设备升级、上传审讯文件时打开文件失败。
NET_DVR_OPERNOTFINISH	36	上次的操作还没有完成
NET_DVR_GETPLAYTIMEFAIL	37	获取当前播放的时间出错。
NET_DVR_PLAYFAIL	38	播放出错。
NET_DVR_FILEFORMAT_ERROR	39	文件格式不正确。
NET_DVR_DIR_ERROR	40	路径错误
NET_DVR_ALLOC_RESOURCE_ERROR	41	SDK 资源分配错误。
NET_DVR_AUDIO_MODE_ERROR	42	声卡模式错误。当前打开声音播放模式与实际设置的模式不符出错。
NET_DVR_NOENOUGH_BUF	43	缓冲区太小。接收设备数据的缓冲区或存放图片缓冲区不足。
NET_DVR_CREATESOCKET_ERROR	44	创建 SOCKET 出错。
NET_DVR_SETSOCKET_ERROR	45	设置 SOCKET 出错。
NET_DVR_MAX_NUM	46	个数达到最大。分配的注册连接数、预览连接数超过 SDK 支持的最大数。
NET_DVR_USERNOTEXIST	47	用户不存在。注册的用户 ID 已注销或不可用。
NET_DVR_WRITEFLASHERROR	48	写 FLASH 出错。设备升级时写 FLASH 失败。
NET_DVR_UPGRADEFAIL	49	设备升级失败。网络或升级文件语言不匹配等原因升级失败。
NET_DVR_CARDHAVEINIT	50	解码卡已经初始化过。
NET_DVR_PLAYERFAILED	51	调用播放库中某个函数失败。
NET_DVR_MAX_USERNUM	52	登录设备的用户数达到最大。
NET_DVR_GETLOCALIPANDMACFAIL	53	获得本地 PC 的 IP 地址或物理地址失败。
NET_DVR_NOENCODEING	54	设备该通道没有启动编码。
NET_DVR_IPMISMATCH	55	IP 地址不匹配。
NET_DVR_MACMISMATCH	56	MAC 地址不匹配。
NET_DVR_UPGRADELANGMISMATCH	57	升级文件语言不匹配。
NET_DVR_MAX_PLAYERPORT	58	播放器路数达到最大。
NET_DVR_NOSPACEBACKUP	59	备份设备中没有足够空间进行备份。
NET_DVR_NODEVICEBACKUP	60	没有找到指定的备份设备。
NET_DVR_PICTURE_BITS_ERROR	61	图像素位数不符，限 24 色。
NET_DVR_PICTURE_DIMENSION_ERROR	62	图片高*宽超限，限 128*256。
NET_DVR_PICTURE_SIZ_ERROR	63	图片大小超限，限 100K。
NET_DVR_LOADPLAYERSDKFAILED	64	载入当前目录下 Player Sdk 出错。
NET_DVR_LOADPLAYERSDKPROC_ERROR	65	找不到 Player Sdk 中某个函数入口。
NET_DVR_LOADDSSDKFAILED	66	载入当前目录下 DsSdk 出错。
NET_DVR_LOADDSSDKPROC_ERROR	67	找不到 DsSdk 中某个函数入口。
NET_DVR_DSSDK_ERROR	68	调用硬解码库 DsSdk 中某个函数失败。
NET_DVR_VOICEMONOPOLIZE	69	声卡被独占。
NET_DVR_JOINMULTICASTFAILED	70	加入多播组失败。
NET_DVR_CREATEDIR_ERROR	71	建立日志文件目录失败。

NET_DVR_BINDSOCKET_ERROR	72	绑定套接字失败。
NET_DVR_SOCKETCLOSE_ERROR	73	socket 连接中断，此错误通常是由于连接中断或目的地不可达。
NET_DVR_USERID_ISUSING	74	注销时用户 ID 正在进行某操作。
NET_DVR_SOCKETLISTEN_ERROR	75	监听失败。
NET_DVR_PROGRAM_EXCEPTION	76	程序异常。
NET_DVR_WRITEFILE_FAILED	77	写文件失败。本地录像、远程下载录像、下载图片等操作时写文件失败。
NET_DVR_FORMAT_READONLY	78	禁止格式化只读硬盘。
NET_DVR_WITHSAMEUSERNAME	79	远程用户配置结构中存在相同的用户名。
NET_DVR_DEVICETYPE_ERROR	80	导入参数时设备型号不匹配。
NET_DVR_LANGUAGE_ERROR	81	导入参数时语言不匹配。
NET_DVR_PARAVERSION_ERROR	82	导入参数时软件版本不匹配。
NET_DVR_IPCHAN_NOTALIVE	83	预览时外接 IP 通道不在线。
NET_DVR_RTSP_SDK_ERROR	84	加载标准协议通讯库 StreamTransClient 失败。
NET_DVR_CONVERT_SDK_ERROR	85	加载转封装库失败。
NET_DVR_IPC_COUNT_OVERFLOW	86	超出最大的 IP 接入通道数。
NET_DVR_MAX_ADD_NUM	87	添加录像标签或者其他操作超出最多支持的个数。
NET_DVR_PARAMMODE_ERROR	88	图像增强仪，参数模式错误（用于硬件设置时，客户端进行软件设置时错误值）。
NET_DVR_CODESPITTER_OFFLINE	89	码分器不在线。
NET_DVR_BACKUP_COPYING	90	设备正在备份。
NET_DVR_CHAN_NOTSUPPORT	91	通道不支持该操作。
NET_DVR_CALLINEINVALID	92	高度线位置太集中或长度线不够倾斜。
NET_DVR_CALCANCELCONFLICT	93	取消标定冲突，如果设置了规则及全局的实际大小尺寸过滤。
NET_DVR_CALPOINTOUTRANGE	94	标定点超出范围。
NET_DVR_FILTERRECTINVALID	95	尺寸过滤器不符合要求。
NET_DVR_DDNS_DEVOFFLINE	96	设备没有注册到 ddns 上。
NET_DVR_DDNS_INTER_ERROR	97	DDNS 服务器内部错误。
NET_DVR_FUNCTION_NOT_SUPPORT_OS	98	此功能不支持该操作系统。
NET_DVR_DEC_CHAN_REBIND	99	解码通道绑定显示输出次数受限。
NET_DVR_INTERCOM_SDK_ERROR	100	加载当前目录下的语音对讲库失败。
NET_DVR_NO_CURRENT_UPDATEFILE	101	没有正确的升级包
NET_DVR_ALIAS_DUPLICATE	150	别名重复（EasyDDNS 的配置）
解码器错误码		
NET_ERR_MAX_WIN_OVERLAP	951	达到最大窗口重叠数
NET_ERR_STREAMID_CHAN_BOTH_VALID	952	stream ID 和通道号同时有效
NET_ERR_NO_ZERO_CHAN	953	设备无零通道
NEED_RECONNECT	955	需要重定向（转码子系统使用）
NET_ERR_NO_STREAM_ID	956	流 ID 不存在
NET_DVR_TRANS_NOT_START	957	转码未启动
NET_ERR_MAXNUM_STREAM_ID	958	流 ID 数达到上限
NET_ERR_WORKMODE_MISMATCH	959	工作模式不匹配

NET_ERR_MODE_IS_USING	960	已工作在当前模式
NET_ERR_DEV_PROGRESSING	961	设备正在处理中
NET_ERR_PASSIVE_TRANSCODING	962	正在被动转码
能力集错误码		
XML_ABILITY_NOTSUPPORT	1000	不支持能力节点获取。
XML_ANALYZE_NOENOUGH_BUF	1001	输出内存不足。
XML_ANALYZE_FIND_LOCALXML_ERROR	1002	无法找到对应的本地 xml。
XML_ANALYZE_LOAD_LOCALXML_ERROR	1003	加载本地 xml 出错。
XML_NANLYZE_DVR_DATA_FORMAT_ERROR	1004	设备能力数据格式错误。
XML_ANALYZE_TYPE_ERROR	1005	能力集类型错误。
XML_ANALYZE_XML_NODE_ERROR	1006	XML 能力节点格式错误。
XML_INPUT_PARAM_ERROR	1007	输入的能力 XML 节点值错误。
XML_VERSION_MISMATCH	1008	XML 版本不匹配。