

DS_8000/7000/6000 系列网络硬盘录像机、IP 设备客户端软件开发包编程手册

版本 2.20(Build 080606)

2008-06-06

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。本手册的内容将做定期的更新，恕不另行通知；更新的内容将会在本手册的新版本中加入。我们随时会改进或更新本手册中描述的产品或程序。

目 录

第一章 简介	1
1.1 SDK 修改记录.....	1
1.2 组成.....	1
1.3 注意事项.....	2
1.3.1 基本注意事项.....	2
1.3.2 显卡注意事项.....	2
1.3.3 硬解码注意事项.....	3
第二章 编程指南	4
2.1 编程说明.....	4
2.2 函数调用顺序.....	5
第三章 数据结构	8
3.1 客户端数据结构.....	8
3.1.1 设备类型定义.....	8
3.1.2 显示模式.....	8
3.1.3 发送模式.....	9
3.1.4 设备信息结构.....	9
3.1.5 图像预览参数.....	9
3.1.6 文件信息结构.....	9
3.1.7 上传报警信息.....	10
3.1.8 报警输出状态.....	10
3.1.9 硬件解码预览图像参数.....	10
3.1.10 日志信息.....	11
3.1.11 设备的状态信息.....	13
3.1.12 ATM DVR 上传交易信息	14
3.2 设备端参数数据结构.....	14
3.2.1 设备系统配置结构.....	15
3.2.2 网络配置结构.....	15
3.2.3 通道图象配置结构.....	16
3.2.4 压缩参数配置结构.....	19
3.2.5 录像参数配置结构.....	21
3.2.6 解码器参数配置结构.....	21
3.2.7 串口配置结构.....	24
3.2.8 报警输入配置结构.....	24
3.2.9 报警输出配置结构.....	25
3.2.10 本地预览参数配置结构.....	25
3.2.11 视频输出配置结构	25
3.2.12 用户参数配置结构.....	26
3.2.13 异常参数配置结构.....	28
3.2.14 帧信息配置结构.....	28
3.2.15 编码端字符叠加配置结构.....	29
3.2.16 6001D/F 配置结构	29
3.2.17 HS 设备辅助输出配置结构.....	31
3.2.18 网络应用参数（NTP/DDNS/EMAIL）	32

3.2.19 NFS 配置	33
3.2.20 IP 快球配置	33
3.2.21 多路解码器参数	34
第四章 函数说明	36
4.1 客户端函数	36
4.1.1 初始化	36
4.1.2 设置显示模式	39
4.1.3 启动/停止监听程序	39
4.1.4 获取错误代码	40
4.1.5 用户注册	41
4.1.6 图像预览	42
4.1.6.1 视频参数	42
4.1.6.2 叠加字符和图像	43
4.1.6.3 播放控制	43
4.1.6.4 声音控制	44
4.1.6.5 捕获	45
4.1.6.6 抓图	46
4.1.7 云台控制	46
4.1.8 回放与下载	52
4.1.8.1 查找文件	52
4.1.8.2 回放	53
4.1.8.3 播放控制	54
4.1.8.4 捕获	55
4.1.8.5 获取 OSD 时间	56
4.1.8.6 抓图	56
4.1.8.7 下载	56
4.1.9 报警	57
4.1.10 语音对讲和转发	58
4.1.10.1 语音对讲	58
4.1.10.2 语音转发	59
4.1.11 语音广播	60
4.1.12 音频编解码	60
4.1.12.1 音频解码	60
4.1.12.2 音频编码	61
4.1.13 透明通道	61
4.1.14 远程控制本地显示	62
4.1.15 远程手动录像	62
4.1.16 解码卡	63
4.1.17 日志	65
4.1.18 获取设备状态	66
4.1.19 JPEG 抓图	66
4.1.20 多显卡支持	67
4.1.21 多路解码器	68
4.2 设备端配置函数	70
4.2.1 参数配置	70
4.2.2 恢复默认值	74

4.2.3 保存参数.....	74
4.2.4 重启/关闭设备.....	75
4.2.5 远程升级.....	75
4.2.6 远程格式化硬盘.....	75
4.2.7 配置交易信息.....	76
4.2.8 6001D/F 配置函数	76
4.2.9 IP 快球配置函数	79

第一章 简介

1.1 SDK 修改记录

版本编号	修改页码和行号	简要说明 (修改内容)	日期
V2.20	P8.L36-44	增加新的设备类型定义	2008-5-12
V2.20	P23.L36-50	增加新的云台协议标号, 自 88 到 102	2008-5-12
V2.20	P32.L33-43	增加扩展 DDNS 参数配置	2008-5-12
V2.20	P13.L26-27	增加新的设备硬盘状态和远程格式化硬盘状态	2008-5-12
V2.20	P19.L34	事件触发参数增加使能位, 复用码流类型字节	2008-5-12
V2.20	P20.L20	增加压缩参数扩展	2008-5-13
V2.20	P69.L15	增加抓 JPEG 图像参数定义	2008-5-13
V2.20	P78.L4	新增获取 SDK 的 build 号的接口	2008-5-13
V2.20	P36.L14	多路解码器新增参数结构	2008-5-13

1.2 组成

客户端软件开发包是 DS_8000HC 系列网络硬盘录像机、DS_6000HC、DS_6100HC 系列网络视频服务器以及 DS_7000、DS_7100 系列产品、IP 设备包括 IP 模块、IP 快球、IP 摄像机的配套产品, 本手册详细介绍了此软件开发包中各个函数所实现的功能以及使用方法、各个函数之间的调用关系。

客户端软件开发包主要由客户端操作和设备端操作两大部分组成:

客户端操作: 图像预览、文件回放和下载、云台控制、布防/撤防、语音对讲、日志管理、解码卡等功能。

设备端操作: 远程升级、远程重启/关闭、格式化硬盘、参数配置(系统配置、通道配置、串口配置、报警配置、用户配置)等功能。

本开发包包括以下文件: HCNetsdk.lib, HCNetsdk.dll, HCNetsdk.h, PlayM4.lib, PlayM4.dll, DsSdk.lib, DsSdk.dll, DataType.h, DecodeCardSdk.h。其中 PlayM4.lib, PlayM4.dll 是解码库, DsSdk.lib, DsSdk.dll, DataType.h, DecodeCardSdk.h 是硬解码的库函数。

1.3 注意事项

1.3.1 基本注意事项

- 本软件开发包同时兼容 ATM 专用设备 (DS8002AH 和 DS8002AHL) 和 DS8000HC, DS6000 系列的网络硬盘录像机。
- 同时支持 TCP、UDP、RTP、MULTICAST 网络传输模式。
- 本客户端可以采用软解码和硬解码 (采用 MD 系列的解码卡) 进行网络预览图像, 2 种模式都支持以下码流格式: QCIF、CIF、2CIF、4CIF、DCIF。
- ATM 专用设备 (DS8002AH 和 DS8002AHL) 只支持 CIF 的分辨率。
- 网络预览时, 设备的每个通道最多支持 6 路同时连接, 每台设备最多支持 24 路同时连接。
- 本开发包同时支持 512 个用户注册, 同时支持 100 路网络预览和网络回放(回放和下载的总数), 同时支持 512 路报警上传, 解码卡最多支持 64 路输出。以上均为理论数值, 实际情况还得看设备的运行情况和运行客户端的计算机 CPU 能力。
- 本软件开发包同时兼容 DS6100 系列、DS7000、DS7100 系列产品。
- 本软件开发包同时兼容 IP 模块、IP 快球、IP 摄像机。

1.3.2 显卡注意事项

此软件开发包的图像显示部分采用的是我们公司自己的播放器, 播放器的显示部分, 主要采用了 DirectDraw 技术。目前使用了 off_screen 画面和 Blt 操作。我们在画面的放大缩小方面主要依靠显卡的支持, 因为通过软件来缩放, 会占用大量 CPU 资源。为此, 我们提供了测试系统是否支持的一个接口 NET_DVR_IsSupport (), 用户可以用它测试在你的系统中播放器是否能正常工作, 另外, 显示属性里的颜色设置应该设置为 16 位或 32 位的, 对于 24 位的, 我们是不支持的。

我们测试过的几款显卡有以下几种 (Win2000 下)

显卡型号	显存 (M)	是否支持颜色转换	是否支持缩小	是否支持放大
ATI Rage128	32	是	是	是
ATI Radeon LE	32	是	是	是
ATI Radeon 7200	64	是	是	是
nVidia TNT2 Model64	16 和 32	是	是	是
nVidia TNT2 Pro	32	是	是	是
Geforce2 Mx, Mx200, Mx400	32	是	是	是
Geforce4 Mx420, Mx440	32	是	是	是

Sis630	16	否	否	否
Sis305	32	是	否	否

注意: nVidia 公司的显卡需要更新最新的驱动, 老的驱动可能不支持缩小功能; 其他没有测试过的显卡, 如果在使用过程中发现不支持某些播放需要的功能, 不妨安装该显卡的最新的驱动

程序试试。

1.3.3 硬解码注意事项

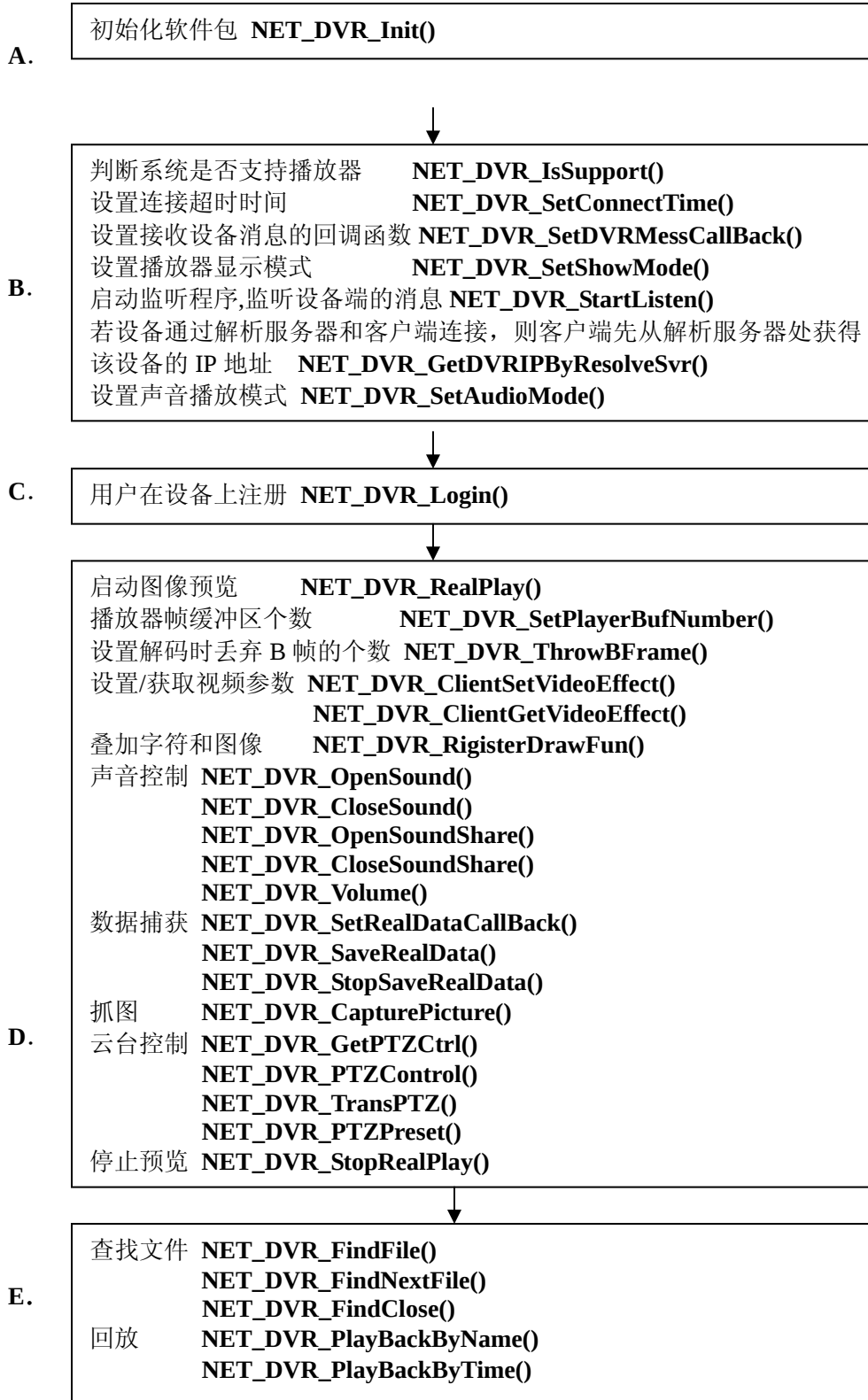
在此开发包中，网络预览图像时支持软件解码和硬件解码，硬件解码采用的是 DS400XMD 系列的解码卡，**此解码卡的具体功能请参见相关解码卡的功能书**，此解码卡具有视频矩阵功能，并且可以解 4CIF 的图像。由于功能的增加此解码卡较原来的 D 卡增加了一些 API 函数，增加的函数不通过客户端用户可以直接调用，这样以后修改解码卡的库函数时就不用修改客户端的 SDK。客户端和用户应用程序同时对这个解码卡的库函数操作，要注意对解码卡的初始化和释放函数只能做一次，一般都是由客户端来初始化解码卡，解码通道可以重复打开，现在返回的解码句柄其实就是解码通道号。

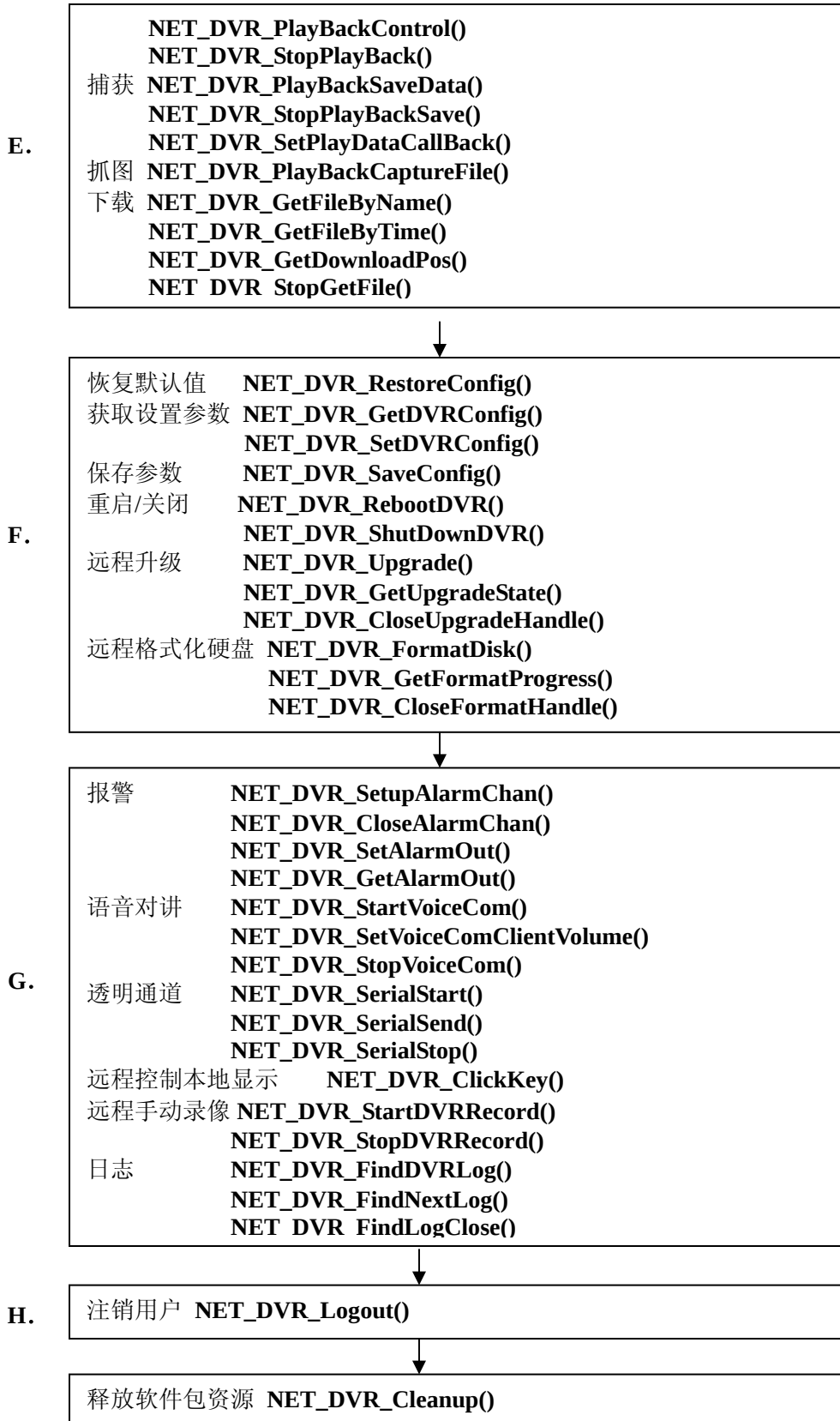
第二章 编程指南

2.1 编程说明

- **初始化:** 使用客户端软件包首先对系统进行初始化 `NET_DVR_Init()`, 退出该软件包则调用 `NET_DVR_Cleanup()` 释放资源。
- **系统检测:** 检测系统函数 `NET_DVR_IsSupport()` 可以在 `NET_DVR_Init()` 之前调用, 如果系统不支持则无法正常显示图像。
- **用户注册:** 用户在访问设备之前必须通过 `NET_DVR_Login()` 函数到设备上注册, 注册成功后返回该用户的 ID 值。此 ID 值就像一把钥匙一样, 之后该用户都必须通过此 ID 号访问该设备, 每台设备允许同时有 **16** 个不同用户注册。退出对设备的操作则通过 `NET_DVR_Logout()` 函数在设备上注销此 ID 号, 该用户就无权访问设备了。
- **主动上传:** 软件包中所有需主动上传的信息 (异常除外) 都通过 `NET_DVR_SetDVRMessCallBack()` 函数设置的回调函数上传 (具体回调内容参见函数说明)。软件包中的异常信息则通过 `NET_DVR_SetDVRMessage()` 函数设置的消息窗口接收。
- **网络延时和流畅度:** 软解码时, 通过 `NET_DVR_SetPlayerBufNumber()` 函数调节播放的延时和流畅程度, 如果 `dwBufNum` 值越大, 播放的流畅性越好, 相应的延时比较大, `dwBufNum` 值越小, 播放的延时很小, 但是当网络不太顺畅的时候, 会有丢帧现象, 感觉播放不会很流畅。一般设置的帧缓冲大于等于 6 帧时, 音频预览才会正常, 如果不需要音频预览, 只需要视频实时性则这个值可以设置的更小。此函数要紧跟在 `NET_DVR_RealPlay()` 后使用, 在图像播放之后设置则不起作用! 硬解码时则通过相应的硬解码的函数来调整延时和流畅程度。
- **云台控制:** 云台控制分为两种模式, 一种是通过图像预览返回的句柄进行控制, 一种是没有预览限制, 通过用户 ID 号进行云台控制。在控制云台之前, 须先获取云台的控制权, 返回该用户当前是否有权限控制云台, 这样做的目的是为了防止优先级不同的用户同时控制云台造成的混乱。(具体参见后面的函数说明)
- **透明通道:** 可以分别选择 485 和 232 串口作为透明通道功能, 其中要把 232 串口作为透明通道使用, 必须先在 232 串口的配置信息 (结构 `NET_DVR_RS232CFG`) 中将工作模式选为透明通道, 这样 232 串口才可作为透明通道使用。
- **回放和下载:** 在调用回放和下载命令后, 回放和下载并没有立即执行, 要调用开始播放 (`NET_DVR_PLAYSTART`) 来启动当前操作, 用 `NET_DVR_PlayBackControl()` 函数实现。在调用开始播放之前, 可以设置保存数据 `NET_DVR_PlayBackSaveData()` 和回调函数 `NET_DVR_SetPlayDataCallBack()`。

2.2 函数调用顺序





I.

以上为软件解码的函数调用流程，其中 D、E、F、G 四大部分内容是没有先后调用顺序的。
F 模块主要完成对设备的配置工作，如果采用硬件解码，则只需把其中的 D 模块用以下的代替：

初始化解码卡	NET_DVR_InitDevice_Card()
初始化解码卡 OVERLAY 表面	NET_DVR_InitDDraw_Card()
解码卡预览图像	NET_DVR_RealPlay_Card()
	NET_DVR_ResetPara_Card()
	NET_DVR_OpenSound_Card()
	NET_DVR_CloseSound_Card()
	NET_DVR_SetVolume_Card()
	NET_DVR_AudioPreview_Card()
数据捕获	NET_DVR_SetRealDataCallBack()
	NET_DVR_SaveRealData()
	NET_DVR_StopSaveRealData()
云台控制	NET_DVR_GetPTZCtrl()
	NET_DVR_PTZControl()
	NET_DVR_TransPTZ()
	NET_DVR_PTZPreset()
停止预览	NET_DVR_StopRealPlay()
刷新解码卡使用的 OVERLAY 表面	NET_DVR_RefreshSurface_Card()
清除解码卡使用的 OVERLAY 表面	NET_DVR_ClearSurface_Card()
恢复解码卡使用的 OVERLAY 表面	NET_DVR_RestoreSurface_Card()
释放解码卡使用的 OVERLAY 表面	NET_DVR_ReleaseDDraw_Card()
释放解码卡资源	NET_DVR_ReleaseDevice_Card()

第三章 数据结构

3.1 客户端数据结构

3.1.1 设备类型定义

```

#define DVR          1      /*对尚未定义的 dvr 类型返回 NETRET_DVR*/
#define ATMDVR       2      /*atm dvr*/
#define DVS          3      /*DVS*/
#define DEC          4      /* 6001D */
#define ENC_DEC       5      /* 6001F */
#define DVR_HC        6      /*8000HC*/
#define DVR_HT        7      /*8000HT*/
#define DVR_HF        8      /*8000HF*/
#define DVR_HS        9      /* 8000HS DVR(no audio) */
#define DVR_HTS       10     /* 8016HTS DVR(no audio) */
#define DVR_HB        11     /* HB DVR(SATA HD) */
#define DVR_HCS       12     /* 8000HCS DVR */
#define DVS_A         13     /* 带 ATA 硬盘的 DVS */
#define DVR_HC_S      14     /* 8000HC-S */
#define DVR_HT_S      15     /* 8000HT-S */
#define DVR_HF_S      16     /* 8000HF-S */
#define DVR_HS_S      17     /* 8000HS-S */
#define ATMDVR_S      18     /* ATM-S */
#define DVR_7000H     19     /*7000 DVR*/
#define DEC_MAT       20     /*多路解码器*/
#define DVR_MOBILE    21     /* mobile DVR */
#define DVR_HD_S      22     /* 8000HD-S */
#define DVR_HDL_S     23     /* 8000HDL-S */
#define DVR_HCL_S     24     /* 8000HCL-S */
#define DVR_HL_S      25     /* 8000HL-S */
#define DVS_HW        26     /* 6000HW */
#define IPCAM         30     /*IP 摄像机*/
#define IPDOME        40     /*IP 快球*/
#define IPMOD         50     /*IP 模块*/

```

3.1.2 显示模式

```
enum{NORMALMODE = 0,OVERLAYMODE};
```

NORMALMODE: 可以同时显示多窗口，但是对显卡有一定的要求；

OVERLAYMODE: 只能同时显示一个窗口, 但是对显卡基本没有要求;

3.1.3 发送模式

```
enum{PTOPTCPMODE =0, PTOUDPMODE, MULTIMODE, RTPMODE, AUDIODETACH,
RESERVEDMODE};
PTOPTCPMODE: TCP 方式
PTOUDPMODE: UDP 方式
MULTIMODE: 多播方式
RTPMODE: RTP方式
AUDIODETACH: 音视频分开模式
RESERVEDMODE: 保留模式
```

3.1.4 设备信息结构

```
typedef struct {
    BYTE sSerialNumber[SERIALNO_LEN]; /* 序列号 */
    BYTE byAlarmInPortNum; /* DVR 报警输入个数 */
    BYTE byAlarmOutPortNum; /* DVR 报警输出个数 */
    BYTE byDiskNum; /* DVR 硬盘个数 */
    BYTE byDVRType; /* DVR 类型*/
    BYTE byChanNum; /* DVR 通道个数 */
    BYTE byStartChan; /* 起始通道号,例如 DVS-1,DVR - 1 */
}NET_DVR_DEVICEINFO, *LPNET_DVR_DEVICEINFO;
```

3.1.5 图像预览参数

```
typedef struct{
    LONG IChannel; /* 通道号 */
    LONG ILinkMode; /* 最高位(31)为 0 表示主码流, 为 1 表示子码流, 0—30 位表示码流连接方式: 0: TCP 方式,1: UDP 方式,2: 多播方式,3 - RTP 方式, 4—音视频分开 */
    HWND hPlayWnd; /* 播放窗口的句柄 */
    char* sMultiCastIP; /* 多播组地址 */
}NET_DVR_CLIENTINFO, *LPNET_DVR_CLIENTINFO;
如果将 hPlayWnd 参数设置为 NULL, 则客户端收到数据后不解码, 但是可以录像。
```

3.1.6 文件信息结构

```
typedef struct{
    DWORD dwYear; /* 年 */
    DWORD dwMonth; /* 月 */
    DWORD dwDay; /* 日 */
}
```

```

    DWORD dwHour;      /* 时 */
    DWORD dwMinute;    /* 分 */
    DWORD dwSecond;    /* 秒 */
} NET_DVR_TIME,*LPNET_DVR_TIME;

typedef struct
{
    char sFileName[100]; /* 文件名 */
    NET_DVR_TIME struStartTime; /* 文件的开始时间 */
    NET_DVR_TIME struStopTime; /* 文件的结束时间 */
    DWORD dwFileSize; /* 文件的大小 */
}NET_DVR_FIND_DATA,*LPNET_DVR_FIND_DATA;

```

3.1.7 上传报警信息

```

typedef struct{
    DWORD dwAlarmType; /*0-信号量报警,1-硬盘满,2-信号丢失, 3-移动侦测, 4-
    硬盘未格式化, 5-读写硬盘出错,6-遮挡报警,7-制式不匹配, 8-非法访问*/
    DWORD dwAlarmInputNumber; /*报警输入端口*/
    DWORD dwAlarmOutputNumber[MAX_ALARMOUT]; /*报警输入端口对应的输出端口,
    哪一位为 1 表示对应哪一个输出*/
    DWORD dwAlarmRelateChannel[MAX_CHANNUM]; /*报警输入端口对应的录像通道,
    哪一位为 1 表示对应哪一路录像,dwAlarmRelateChannel[0]对应第 1 个通道*/
    DWORD dwChannel[MAX_CHANNUM]; /*dwAlarmType 为 2 或 3 时, 表示哪个
    通道, dwChannel[0]位对应第 0 个通道*/
    DWORD dwDiskNumber[MAX_DISKNUM]; /*dwAlarmType 为 1, 4, 5 时,表示哪个
    硬盘*/
}NET_DVR_ALARMINFO,*LPNET_DVR_ALARMINFO;

```

注：当 dwAlarmType 为 5 时，dwDiskNumber[]数组全部为 0，则表示当前未接硬盘。

3.1.8 报警输出状态

```

typedef struct{
    BYTE Output[MAX_ALARMOUT]; /* 报警输出的状态 0:无效 1:有效 */
}NET_DVR_ALARMOUTSTATUS,*LPNET_DVR_ALARMOUTSTATUS;

```

3.1.9 硬件解码预览图像参数

```

typedef struct{
    long lToScreen; /* 是否输出到显示器屏幕上, 1-是, 0-否 */
    long lToVideoOut; /* 是否输出到监视器上, 1-是, 0-否 */
    long lLeft; /* 输出位置的左上点的横坐标, 相对与父窗口而言, lToScreen 为 1 时//
    需要指定 */

```

```

    long lTop;      /* 输出位置的左上点的纵坐标，相对与父窗口而言，lToScreen 为 1 时//
                    需要指定 */
    long lWidth;    /* 输出图象的宽度，lToScreen 为 1 时需要指定 */
    long lHeight;   /* 输出图象的高度，lToScreen 为 1 时需要指定 */
    long lReserved; /* 保留 */
} NET_DVR_DISPLAY_PARA,*LP NET_DVR_DISPLAY_PARA;

typedef struct{
    LONG lChannel;      /* 通道号 */
    LONG lLinkMode; /*最高位(31)为 0 表示主码流，为 1 表示子码流，0—30 位表示码流连接
                    方式：0: TCP 方式,1: UDP 方式,2: 多播方式,3 - RTP 方式，4—电话线，
                    5—128k 宽带，6—256k 宽带，7—384k 宽带，8—512k 宽带*/
    char* sMultiCastIP; /* 多播组地址 */
    NET_DVR_DISPLAY_PARA struDisplayPara;
}NET_DVR_CARDINFO,*LPNET_DVR_CARDINFO;

```

3.1.10 日志信息

```

/* 报警 */
//主类型
#define MAJOR_ALARM                0x1
//次类型
#define MINOR_ALARM_IN             0x1    /* 报警输入 */
#define MINOR_ALARM_OUT           0x2    /* 报警输出 */
#define MINOR_MOTDET_START        0x3    /* 移动侦测报警开始 */
#define MINOR_MOTDET_STOP        0x4    /* 移动侦测报警结束 */
#define MINOR_HIDE_ALARM_START   0x5    /* 遮挡报警开始 */
#define MINOR_HIDE_ALARM_STOP    0x6    /* 遮挡报警结束 */

/* 异常 */
//主类型
#define MAJOR_EXCEPTION            0x2
//次类型
#define MINOR_VI_LOST             0x21   /* 信号丢失 */
#define MINOR_ILLEGAL_ACCESS      0x22   /* 非法访问 */
#define MINOR_HD_FULL             0x23   /* 硬盘满 */
#define MINOR_HD_ERROR            0x24   /* 硬盘错误 */
#define MINOR_DCD_LOST            0x25   /* MODEM 掉线 */
#define MINOR_IP_CONFLICT         0x26   /* IP 地址冲突 */
#define MINOR_NET_BROKEN          0x27   /* 网线断 */

/* 操作 */
//主类型
#define MAJOR_OPERATION            0x3
//次类型
#define MINOR_START_DVR           0x41   /* 开机 */
#define MINOR_STOP_DVR           0x42   /* 关机 */

```

```
#define MINOR_STOP_ABNORMAL      0x43    /* 非法关机 */

#define MINOR_LOCAL_LOGIN        0x50    /* 本地登陆 */
#define MINOR_LOCAL_LOGOUT       0x51    /* 本地注销登陆 */
#define MINOR_LOCAL_CFG_PARM     0x52    /* 本地配置参数 */
#define MINOR_LOCAL_PLAYBYFILE   0x53    /* 本地按文件回放 */
#define MINOR_LOCAL_PLAYBYTIME   0x54    /* 本地按时间回放 */
#define MINOR_LOCAL_START_REC    0x55    /* 本地开始录像 */
#define MINOR_LOCAL_STOP_REC     0x56    /* 本地停止录像 */
#define MINOR_LOCAL_PTZCTRL      0x57    /* 本地云台控制 */
#define MINOR_LOCAL_PREVIEW      0x58    /* 本地预览 */
#define MINOR_LOCAL_MODIFY_TIME  0x59    /* 本地修改时间 */
#define MINOR_LOCAL_UPGRADE      0x5a/* 本地升级 */
#define MINOR_LOCAL_COPYFILE     0x5b    /* 本地备份文件 */

#define MINOR_REMOTE_LOGIN       0x70    /* 远程登录 */
#define MINOR_REMOTE_LOGOUT      0x71    /* 远程注销登陆 */
#define MINOR_REMOTE_START_REC   0x72    /* 远程开始录像 */
#define MINOR_REMOTE_STOP_REC    0x73    /* 远程停止录像 */
#define MINOR_START_TRANS_CHAN   0x74    /* 开始透明传输 */
#define MINOR_STOP_TRANS_CHAN    0x75    /* 停止透明传输 */
#define MINOR_REMOTE_GET_PARM    0x76    /* 远程获得参数 */
#define MINOR_REMOTE_CFG_PARM    0x77    /* 远程配置参数 */
#define MINOR_REMOTE_GET_STATUS  0x78/* 远程获得状态 */
#define MINOR_REMOTE_ARM         0x79    /* 远程布防 */
#define MINOR_REMOTE_DISARM      0x7a/* 远程撤防 */
#define MINOR_REMOTE_REBOOT      0x7b    /* 远程重启 */
#define MINOR_START_VT           0x7c/* 开始语音对讲 */
#define MINOR_STOP_VT            0x7d    /* 停止语音对讲 */
#define MINOR_REMOTE_UPGRADE     0x7e/* 远程升级 */
#define MINOR_REMOTE_PLAYBYFILE  0x7f/* 远程按文件回放 */
#define MINOR_REMOTE_PLAYBYTIME  0x80    /* 远程按时间回放 */
#define MINOR_REMOTE_PTZCTRL     0x81    /* 远程云台控制 */
```

//参数类型:

当日志的主类型为 MAJOR_OPERATION=03, 次类型为 MINOR_LOCAL_CFG_PARM=0x52 或者 MINOR_REMOTE_GET_PARM=0x76 或者 MINOR_REMOTE_CFG_PARM=0x77 时, dwParaType:参数类型有效, 其含义如下:

```
#define PARA_VIDEOOUT 0x1
#define PARA_IMAGE    0x2
#define PARA_ENCODE   0x4
#define PARA_NETWORK  0x8
#define PARA_ALARM     0x10
#define PARA_EXCEPTION 0x20
#define PARA_DECODER   0x40    /*解码器*/
#define PARA_RS232     0x80
#define PARA_PREVIEW   0x100
#define PARA_SECURITY  0x200
#define PARA_DATETIME  0x400
```

```
#define PARA_FRAMETYPE 0x800    /*帧格式*/

typedef struct {
    NET_DVR_TIME strLogTime;
    DWORD dwMajorType;    /* 主类型 1-报警; 2-异常; 3-操作; 0xff-全部*/
    DWORD dwMinorType;    /* 次类型 */
    BYTE sPanelUser[MAX_NAMELEN]; /* 操作面板的用户名 */
    BYTE sNetUser[MAX_NAMELEN]; /* 网络操作的用户名 */
    char sRemoteHostAddr[16]; /* 远程主机地址 */
    DWORD dwParaType; /* 参数类型 */
    DWORD dwChannel; /* 通道号 */
    DWORD dwDiskNumber; /* 硬盘号 */
    DWORD dwAlarmInPort; /* 报警输入端口 */
    DWORD dwAlarmOutPort; /* 报警输出端口 */
}NET_DVR_LOG,*LPNET_DVR_LOG;
```

3.1.11 设备的状态信息

```
typedef struct {
    DWORD dwVolume; /* 硬盘的容量 */
    DWORD dwFreeSpace; /* 硬盘的剩余空间 */
    DWORD dwHardDiskStatic; /* 硬盘的状态: 0-活动, 1-休眠, 2-不正常, 3-休眠硬盘出
错 */
}NET_DVR_DISKSTATE,*LPNET_DVR_DISKSTATE;

typedef struct {
    BYTE byRecordStatic; /* 通道是否在录像,0-不录像,1-录像 */
    BYTE bySignalStatic; /* 连接的信号状态,0-正常,1-信号丢失 */
    BYTE byHardwareStatic; /* 通道硬件状态,0-正常,1-异常,例如DSP死掉 */
    char reservedData; /* 保留 */
    DWORD dwBitRate; /* 实际码率 */
    DWORD dwLinkNum; /* 客户端连接的个数 */
    DWORD dwClientIP[MAX_LINK]; /* 客户端的IP地址 */
}NET_DVR_CHANNELSTATE,*LPNET_DVR_CHANNELSTATE;

typedef struct{
    DWORD dwDeviceStatic; /* 设备的状态,0-正常,1-CPU占用率太高,超过85%,2-硬件错误,
例//如串口死掉 */
    NET_DVR_DISKSTATE struHardDiskStatic[MAX_DISKNUM]; /* 硬盘状态 */
    NET_DVR_CHANNELSTATE struChanStatic[MAX_CHANNUM]; /* 通道的状态 */
    BYTE byAlarmInStatic[MAX_ALARMIN]; /* 报警端口的状态,0-没有报警,1-有报警 */
    BYTE byAlarmOutStatic[MAX_ALARMOUT]; /* 报警输出端口的状态,0-没有输出,1-
有报警//输出 */
    DWORD dwLocalDisplay; /* 本地显示状态,0-正常,1-不正常 */
}NET_DVR_WORKSTATE,*LPNET_DVR_WORKSTATE;
```

3.1.12 ATM DVR 上传交易信息

```
typedef struct{
    USHORT m_Year;          /* 年 */
    USHORT m_Month;         /* 月 */
    USHORT m_Day;           /* 日 */
    USHORT m_Hour;           /* 时 */
    USHORT m_Minute;         /* 分 */
    USHORT m_Second;         /* 秒 */
    BYTE DeviceName[24];     /* 设备名称 */
    DWORD dwChannelNumer;    /* 通道号 */
    BYTE CardNumber[32];     /* 卡号 */
    char cTradeType[12];     /* 交易类型 */
    DWORD dwCash;            /* 交易金额 */
}NET_DVR_TRADEINFO,*LPNET_DVR_TRADEINFO;
```

注：所有信息保存后需要重启设备。

3.2 设备端参数数据结构

```
#define NAME_LEN            32
#define SERIALNO_LEN       48
#define MACADDR_LEN        6
#define MAX_ETHERNET       2
#define PATHNAME_LEN       128
#define PASSWD_LEN         16
#define MAX_CHANNUM        16
#define MAX_ALARMOUT       4
#define MAX_TIMESEGMENT    4
#define MAX_PRESET         128
#define MAX_DAYS           7
#define PHONENUMBER_LEN    32
#define MAX_DISKNUM        16
#define MAX_WINDOW         16
#define MAX_VGA            1
#define MAX_USERNUM        16
#define MAX_EXCEPTIONNUM   16
#define MAX_LINK           6
#define MAXCARD_NUM        40
#define MAX_ALARMIN        16
#define MAX_SERIALNUM      2
#define CARDNUM_LEN        20
#define MAX_VIDEOOUT       2
#define MAX_DISPLAY_REGION 16
#define MAX_NAMELEN        16
#define MAX_RIGHT          32
#define MAX_SHELTERNUM     4
#define MAX_DECPOOLNUM     4
```

```
#define MAX_DECNUM          4
#define MAX_TRANSPARENTNUM 2
#define MAX_STRINGNUM      4
#define MAX_AUXOUT          4
#define MAX_HD_COUNT       24
#define MAX_CYCLE_CHAN     16
#define MAX_NFS_DISK        8
```

3.2.1 设备系统配置结构

```
typedef struct{
    DWORD dwSize;
    BYTE sDVRName[NAME_LEN]; /* DVR 名称 */
    DWORD dwDVRID; /* DVR ID,用于遥控器 */
    DWORD dwRecycleRecord; /* 是否循环录像,0:不是; 1:是 */
//以下不能更改
    BYTE sSerialNumber[SERIALNO_LEN]; /* 序列号 */
    DWORD dwSoftwareVersion; /* 软件版本号,高 16 位是主版本,低 16 位是次版本 */
    DWORD dwSoftwareBuildDate; /* 软件生成日期,0xYYYYMMDD */
    DWORD dwDSPSoftwareVersion; /* DSP 软件版本,高 16 位是主版本,低 16 位是次版本
*/
    DWORD dwDSPSoftwareBuildDate; /* DSP 软件生成日期,0xYYYYMMDD */
    DWORD dwPanelVersion; /* 前面板版本,高 16 位是主版本,低 16 位是次版本 */
    DWORD dwHardwareVersion; /* 硬件版本,高 16 位是主版本,低 16 位是次版本 */
    BYTE byAlarmInPortNum; /* DVR 报警输入个数 */
    BYTE byAlarmOutPortNum; /* DVR 报警输出个数 */
    BYTE byRS232Num; /* DVR 232 串口个数 */
    BYTE byRS485Num; /* DVR 485 串口个数 */
    BYTE byNetworkPortNum; /* 网络口个数 */
    BYTE byDiskCtrlNum; /* DVR 硬盘控制器个数 */
    BYTE byDiskNum; /* DVR 硬盘个数 */
    BYTE byDVRType; /* DVR 类型, 1:DVR 2:ATM DVR 3:DVS 等等 */
    BYTE byChanNum; /* DVR 通道个数 */
    BYTE byStartChan; /* 起始通道号,例如 DVS-1,DVR - 1 */
    BYTE byDecordChans; /* DVR 解码路数 */
    BYTE byVGA Num; /* VGA 口的个数 */
    BYTE byUSBNum; /* USB 口的个数 */
    char reservedData[3]; /* 保留 */
}NET_DVR_DEVICECFG,*LPNET_DVR_DEVICECFG;
```

注：当设备恢复默认参数或者升级完成后需要重启设备。

3.2.2 网络配置结构

```
typedef struct {
    char sDVRIP[16]; /* DVR IP 地址 */
```

```

char sDVRIPMask[16];          /* DVR IP 地址掩码 */
DWORD dwNetInterface;        /* 10M/100M 自适应,索引 */
                                //1-10MBase - T
                                //2-10MBase-T 全双工
                                //3-100MBase - TX
                                //4-100M 全双工
                                //5-10M/100M 自适应
WORD wDVRPort;               /* 端口号 */
BYTE byMACAddr[MACADDR_LEN]; /* 服务器的物理地址 */
}NET_DVR_ETHERNET;
typedef struct{
    DWORD dwSize;
    NET_DVR_ETHERNET struEtherNet[MAX_ETHERNET]; /* 以太网口 */
    char sManageHostIP[16]; /* 远程管理主机地址 */
    WORD wManageHostPort; /* 远程管理主机端口号 */
    char sIPServerIP[16]; /* IP Server 服务器地址 */
    char sMultiCastIP[16]; /* 多播组地址 */
    char sGatewayIP[16]; /* 网关地址 */
    char sNFSIP[16]; /* NAS 主机 IP 地址 */
    BYTE sNFSDirectory[PATHNAME_LEN]; /* NAS 目录 */
    DWORD dwPPPOE; /* 0-不启用,1-启用 */
    BYTE sPPPoEUser[NAME_LEN]; /* PPPoE 用户名 */
    char sPPPoEPassword[PASSWD_LEN]; /* PPPoE 密码 */
    char sPPPoEIP[16]; /* PPPoE IP 地址(只读) */
    WORD wHttpPort; /* http 端口号 */
}NET_DVR_NETCFG,*LPNET_DVR_NETCFG;

//多路解码器专用
typedef struct tagNETCFG_OTHER{
    DWORD dwSize;
    char sFirstDNSIP[16];
    char sSecondDNSIP[16];
    char sRes[32];
}NET_DVR_NETCFG_OTHER,*LPNET_DVR_NETCFG_OTHER;

```

注：在网络配置参数设置后需要重启设备。

3.2.3 通道图象配置结构

```

typedef struct
{
    DWORD dwHandleType; /*处理方式,处理方式的"或"结果*/
                                /*0x00: 无响应*/
                                /*0x01: 监视器上警告*/
                                /*0x02: 声音警告*/
                                /*0x04: 上传中心*/
                                /*0x08: 触发报警输出*/
                                /*0x10: Jpeg 抓图并上传 EMail(中性功能)*/

```

```

/*0x20: 触发抓图*/ //(网吧 JPEG 定制)
BYTE byRelAlarmOut[MAX_ALARMOUT]; /* 报警触发的输出通道,报警触发的输出,为
1 表示触发该输出 */
}NET_DVR_HANDLEEXCEPTION;

typedef struct{
    BYTE byMotionScope[18][22]; /*侦测区域,共有 22*18 个小宏块,为 1 表示改宏块是移动侦
测区域,0-表示不是*/
    BYTE byMotionSensive; /*移动侦测灵敏度, 0 - 5,越高越灵敏,0xff 关闭*/
    BYTE byEnableHandleMotion; /* 是否处理移动侦测 */
    NET_DVR_HANDLEEXCEPTION strMotionHandleType; /* 处理方式 */
    NET_DVR_SCHEDULETIME struAlarmTime[MAX_DAYS][MAX_TIMESEGMENT]; /*布防时
间 */
    BYTE byRelRecordChan[MAX_CHANNUM]; //报警触发的录像通道,为 1 表示触发该通道
}NET_DVR_MOTION,*LPNET_DVR_MOTION;

//遮挡报警 区域大小 704*576
typedef struct{
    DWORD dwEnableHideAlarm; /* 是否启动遮挡报警 ,0-否,1-低灵敏度 2-
中灵敏度 3-高灵敏度*/
    WORD wHideAlarmAreaTopLeftX; /* 遮挡区域的 x 坐标 */
    WORD wHideAlarmAreaTopLeftY; /* 遮挡区域的 y 坐标 */
    WORD wHideAlarmAreaWidth; /* 遮挡区域的宽 */
    WORD wHideAlarmAreaHeight; /*遮挡区域的高*/
    NET_DVR_HANDLEEXCEPTION strHideAlarmHandleType; /* 处理方式 */
    NET_DVR_SCHEDULETIME struAlarmTime[MAX_DAYS][MAX_TIMESEGMENT]; /*布防时
间 */
}NET_DVR_HIDEALARM,*LPNET_DVR_HIDEALARM;

//信号丢失报警
typedef struct{
    BYTE byEnableHandleVILost; /* 是否处理信号丢失报警 */
    NET_DVR_HANDLEEXCEPTION strVILostHandleType; /* 处理方式 */
    NET_DVR_SCHEDULETIME struAlarmTime[MAX_DAYS][MAX_TIMESEGMENT]; /*布防时
间 */
}NET_DVR_VILOST,*LPNET_DVR_VILOST;

typedef struct
{
    DWORD dwSize; /* 此结构的大小 */
    BYTE sChanName[NAME_LEN]; /* 通道名称 */
    DWORD dwVideoFormat; /* 只读 视频制式 1-NTSC 2-PAL */
    BYTE byBrightness; /*亮度,0-255*/
    BYTE byContrast; /*对比度,0-255*/
    BYTE bySaturation; /*饱和度,0-255 */
    BYTE byHue; /*色调,0-255*/
    //显示通道名
    DWORD dwShowChanName; /*预览的图象上是否显示通道名称,0-不显示,1-显示区
域大小 704*576*/

```

```

WORD wShowNameTopLeftX;      /* 通道名称显示位置的 x 坐标 */
WORD wShowNameTopLeftY;      /* 通道名称显示位置的 y 坐标 */
//信号丢失报警
NET_DVR_VILOST struVILost;
//移动侦测
NET_DVR_MOTION struMotion;
//遮挡报警
NET_DVR_HIDEALARM strHideAlarm;
//遮挡 区域大小 704*576
DWORD dwEnableHide;          /* 是否启动遮挡 ,0-否,1-是*/
WORD wHideAreaTopLeftX;      /* 遮挡区域的 x 坐标 */
WORD wHideAreaTopLeftY;      /* 遮挡区域的 y 坐标 */
WORD wHideAreaWidth;          /* 遮挡区域的宽 */
WORD wHideAreaHeight;         /* 遮挡区域的高 */
//OSD
DWORD dwShowOsd;              /* 预览的图象上是否显示 OSD,0-不显示,1-显示 */
WORD wOSDTopLeftX;           /* OSD 的 x 坐标 */
WORD wOSDTopLeftY;           /* OSD 的 y 坐标 */
BYTE byOSDType;               /* OSD 类型(主要是年月日格式) */
/* 0: XXXX-XX-XX 年月日 */
/* 1: XX-XX-XXXX 月日年 */
/* 2: XXXX 年 XX 月 XX 日 */
/* 3: XX 月 XX 日 XXXX 年 */
/* 4: XX-XX-XXXX 日月年 */
/* 5: XX 日 XX 月 XXXX 年 */
BYTE byDispWeek;              /* 是否显示星期 */
BYTE byOSDAttrib;             /* OSD 属性:透明, 闪烁 */
/* 1: 透明,闪烁 */
/* 2: 透明,不闪烁 */
/* 3: 闪烁,不透明 */
/* 4: 不透明,不闪烁 */
}NET_DVR_PICCFG,*LPNET_DVR_PICCFG;

typedef struct{
    WORD wHideAreaTopLeftX;      /* 遮挡区域的 x 坐标 */
    WORD wHideAreaTopLeftY;      /* 遮挡区域的 y 坐标 */
    WORD wHideAreaWidth;          /* 遮挡区域的宽 */
    WORD wHideAreaHeight;         /* 遮挡区域的高 */
}NET_DVR_SHELTER,*LP NET_DVR_SHELTER;

typedef struct
{
    DWORD dwSize;
    BYTE sChanName[NAME_LEN];
    DWORD dwVideoFormat; /* 只读 视频制式 1-NTSC 2-PAL */
    BYTE byBrightness; /* 亮度,0-255 */
    BYTE byContrast; /* 对比度,0-255 */
    BYTE bySaturation; /* 饱和度,0-255 */
    BYTE byHue; /* 色调,0-255 */

```

```

//显示通道名
DWORD dwShowChanName; // 预览的图象上是否显示通道名称,0-不显示,1-显示 区域大小 704*576
WORD wShowNameTopLeftX;          /* 通道名称显示位置的 x 坐标 */
WORD wShowNameTopLeftY;          /* 通道名称显示位置的 y 坐标 */
//信号丢失报警
NET_DVR_VILOST struVILost;
//移动侦测
NET_DVR_MOTION struMotion;
//遮挡报警
NET_DVR_HIDEALARM struHideAlarm;
//遮挡 区域大小 704*576
DWORD dwEnableHide;          /* 是否启动遮挡 ,0-否,1-是*/
NET_DVR_SHELTER struShelter[MAX_SHELTERNUM];
//OSD
DWORD dwShowOsd; // 预览的图象上是否显示 OSD,0-不显示,1-显示 区域大小 704*576
WORD wOSDTopLeftX;          /* OSD 的 x 坐标 */
WORD wOSDTopLeftY;          /* OSD 的 y 坐标 */
BYTE byOSDType;          /* OSD 类型(主要是年月日格式) */
/* 0: XXXX-XX-XX 年月日 */
/* 1: XX-XX-XXXX 月日年 */
/* 2: XXXX 年 XX 月 XX 日 */
/* 3: XX 月 XX 日 XXXX 年 */
BYTE byDispWeek;          /* 是否显示星期 */
BYTE byOSDAttrib;          /* OSD 属性:透明, 闪烁 */
/* 1: 透明,闪烁 */
/* 2: 透明,不闪烁 */
/* 3: 闪烁,不透明 */
/* 4: 不透明,不闪烁 */
}NET_DVR_PICCFG_EX,*LPNET_DVR_PICCFG_EX;

```

注：修改移动侦测报警、遮挡报警和视频丢失报警的布防时间段后需要重启设备。

3.2.4 压缩参数配置结构

```

typedef struct{
    BYTE byStreamType;          /* 码流类型 0-视频流,1-复合流 最高位用来表示是否使事件触发参数设置后即生效, 0-设置但不生效, 1-设置即生效*/
    BYTE byResolution;          /* 分辨率 0-DCIF 1-CIF, 2-QCIF, 3-4CIF (ATM DVR 不支持 4CIF) , 4-2CIF */
    BYTE byBitrateType;          /* 码率类型 0: 变码率, 1: 定码率*/
    BYTE byPicQuality;          /* 图象质量 0-最好 1-次好 2-较好 3-一般 4-较差 5-差*/
    DWORD dwVideoBitrate;          /* 视频码率 0-保留 1-16K(保留) 2-32K 3-48k 4-64K 5-80K 6-96K 7-128K 8-160k 9-192K 10-224K 11-256K 12-320K 13-384K 14-448K 15-512K 16-640K 17-768K 18-896K 19-1024K 20-1280K 21-1536K 22-1792K 23-2048K

```

最高位(31 位)置成 1 表示是自定义码流, 0-30 位表示码流值。最小值 16k 最大值 8192k*/

```
    DWORD dwVideoFrameRate; /* 帧率 0-全部; 1-1/16; 2-1/8; 3-1/4; 4-1/2; 5-1; 6-2; 7-4; 8-6; 9-8; 10-10; 11-12; 12-16; 13-20*/
}NET_DVR_COMPRESSION_INFO,*LPNET_DVR_COMPRESSION_INFO;
```

```
typedef struct{
    DWORD dwSize; /* 此结构的大小 */
    NET_DVR_COMPRESSION_INFO struRecordPara; /* 录像/事件触发录像 */
    NET_DVR_COMPRESSION_INFO struNetPara; /* 网传/保留 */
}NET_DVR_COMPRESSIONCFG,*LPNET_DVR_COMPRESSIONCFG;
```

注：当命令是 NET_DVR_GET_EVENTCOMPCFG 和 NET_DVR_SET_EVENTCOMPCFG 时，结构 NET_DVR_COMPRESSIONCFG 定义如下

```
typedef struct{
    DWORD dwSize; /* 此结构的大小 */
    NET_DVR_COMPRESSION_INFO struRecordPara; /*事件触发录像 */
    NET_DVR_COMPRESSION_INFO struNetPara; /*保留 */
}NET_DVR_COMPRESSIONCFG,*LPNET_DVR_COMPRESSIONCFG;
```

```
typedef struct
{
    BYTE byStreamType; /*码流类型 0-视频流,1-复合流
    BYTE byResolution; /*分辨率 0-DCIF 1-CIF, 2-QCIF, 3-4CIF, 4-2CIF 5（保留）16-VGA（640*480）17-UXGA（1600*1200）18-SVGA（800*600）19-HD720p（1280*720）20-SXGA（1280*1024）
    BYTE byBitrateType; /*码率类型 0:变码率，1:定码率
    BYTE byPicQuality; /*图象质量 0-最好 1-次好 2-较好 3-一般 4-较差 5-差
    DWORD dwVideoBitrate; /*视频码率 0-保留 1-16K(保留) 2-32K 3-48k 4-64K 5-80K 6-96K 7-128K 8-160k 9-192K 10-224K 11-256K 12-320K
    // 13-384K 14-448K 15-512K 16-640K 17-768K 18-896K 19-1024K 20-1280K 21-1536K 22-1792K 23-2048K
    //最高位(31 位)置成 1 表示是自定义码流, 0-30 位表示码流值(MIN-32K MAX-8192K)。
    DWORD dwVideoFrameRate; /*帧率 0-全部; 1-1/16; 2-1/8; 3-1/4; 4-1/2; 5-1; 6-2; 7-4; 8-6; 9-8; 10-10; 11-12; 12-16; 13-20; //2.0 还支持 14-15; 15-18; 16-22;
    WORD wIntervalFrameI; /*I 帧间隔
    //2006-08-11 增加单 P 帧的配置接口，可以改善实时流延时问题
    BYTE byIntervalBPFrame; /*0-BBP 帧; 1-BP 帧; 2-单 P 帧 BP 帧设置暂不支持
    BYTE byENumber; /*E 帧数量
}NET_DVR_COMPRESSION_INFO_EX,*LPNET_DVR_COMPRESSION_INFO_EX;
```

```
typedef struct
{
    DWORD dwSize;
    NET_DVR_COMPRESSION_INFO_EX struRecordPara; //录像
    NET_DVR_COMPRESSION_INFO_EX struNetPara; //网传
}NET_DVR_COMPRESSIONCFG_EX,*LPNET_DVR_COMPRESSIONCFG_EX;
```

注：在网传（子码流）中的分辨率只能设置成 CIF 和 QCIF。在设置码流类型(Stream Type)后需

要重启设备；部分设备(8 路和 16 路的 7000 以及 8000HS_S)修改分辨率后也需要重启设备。

3.2.5 录像参数配置结构

```
typedef struct{
    //开始时间
    BYTE byStartHour;
    BYTE byStartMin;
    //结束时间
    BYTE byStopHour;
    BYTE byStopMin;
}NET_DVR_SCHEDULE,*LPNET_DVR_SCHEDULE;

typedef struct{
    NET_DVR_SCHEDULE struRecordTime;
    BYTE byRecordType; /* 0:定时录像, 1:移动侦测, 2:报警录像, 3:动测|报警, 4:动测&报
警 5:命令触发, 6: 手动录像*/
    char reservedData[3]; /* 保留 */
}NET_DVR_RECORDSCHED,*LPNET_DVR_RECORDSCHED;

typedef struct {
    WORD wAllDayRecord; /* 是否全天录像 */
    BYTE byRecordType; /* 录像类型 0:定时录像, 1:移动侦测, 2:报警录像,
3:动测|报警, 4:动测&报警 5:命令触发, 6: 手动录像*/
    char reservedData;
}NET_DVR_RECORDDAY;

typedef struct {
    DWORD dwSize; /* 此结构的大小 */
    DWORD dwRecord; /*是否录像 0-否 1-是*/
    NET_DVR_RECORDDAY struRecAllDay[MAX_DAYS];
    NET_DVR_RECORDSCHED struRecordSched[MAX_DAYS][MAX_TIMESEGMENT];
    DWORD dwRecordTime; /* 录像时间长度 */
    DWORD dwPreRecordTime; /* 预录时间 0-不预录 1-5 秒 2-10 秒 3-15 秒 4-20 秒 5-25 秒
6-30 秒 7-0xffffffff(尽可能预录) */
}NET_DVR_RECORD,*LPNET_DVR_RECORD;
```

注：在修改 sata1 的用途（录像或者备份）、开启/关闭录像以及设置录像时间段都需要重启设备。

3.2.6 解码器参数配置结构

```
typedef struct{
    DWORD dwSize; /* 此结构的大小 */
    DWORD dwBaudRate; /* 波特率(bps), 0-50, 1-75, 2-110, 3-150, 4-300, 5-600,
6-1200, 7-2400, 8-4800, 9-9600, 10-19200, 11-38400, 12-57600, 13-76800, 14
-115.2k; */
    BYTE byDataBit; /* 数据有几位 0-5 位, 1-6 位, 2-7 位, 3-8 位; */
```

```

BYTE byStopBit;      /* 停止位 0—1 位, 1—2 位; */
BYTE byParity;       /* 校验 0—无校验, 1—奇校验, 2—偶校验; */
BYTE byFlowcontrol;  /* 0—无, 1—软流控,2-硬流控 */
WORD wDecoderType;   /* 解码器类型, 见下表*/
WORD wDecoderAddress; /*解码器地址:0 - 255*/
BYTE bySetPreset[MAX_PRESET]; /* 预置点是否设置,0-没有设置,1-设置*/
BYTE bySetCruise[MAX_PRESET]; /* 巡航是否设置: 0-没有设置,1-设置 */
BYTE bySetTrack[MAX_PRESET]; /* 轨迹是否设置,0-没有设置,1-设置*/
}NET_DVR_DECODERCFG,*LPNET_DVR_DECODERCFG;

```

```

/* PTZ type */
#define YOULI 0
#define LILIN_1016 1
#define LILIN_820 2
#define PELCO_P 3
#define DM_QUICKBALL 4
#define HD600 5
#define JC4116 6
#define PELCO_DWX 7
#define PELCO_D 8
#define VCOM_VC_2000 9
#define NETSTREAMER 10
#define SAE 11
#define SAMSUNG 12
#define KALATEL_KTD_312 13
#define CELOTEX 14
#define TLPELCO_P 15
#define TL_HHX2000 16
#define BBV 17
#define RM110 18
#define KC3360S 19
#define ACES 20
#define ALSON 21
#define INV3609HD 22
#define HOWELL 23
#define TC_PELCO_P 24
#define TC_PELCO_D 25
#define AUTO_M 26
#define AUTO_H 27
#define ANTEN 28
#define CHANGLIN 29
#define DELTADOME 30
#define XYM_12 31
#define ADR8060 32
#define EVI 33
#define Demo_Speed 34
#define DM_PELCO_D 35
#define ST_832 36
#define LC_D2104 37
#define HUNTER 38
#define A01 39
#define TECHWIN 40

```

#define WEIHAN	41
#define LG	42
#define D_MAX	43
#define PANASONIC	44
#define KTD_348	45
#define INFINOVA	46
#define LILIN	47
#define IDOME_IVIEW_LCU	48
#define DENNARD_DOME	49
#define PHILIPS	50
#define SAMPLE	51
#define PLD	52
#define PARCO	53
#define HY	54
#define NAIJIE	55
#define CAT_KING	56
#define YH_06	57
#define SP9096X	58
#define M_PANEL	59
#define M_MV2050	60
#define SAE_QUICK	61
#define PEARMAIN	62
#define NKO8G	63
#define DAHUA	64
#define TX_CONTROL_232	65
#define VCL_SPEED_DOME	66
#define ST_2C160	67
#define TDWY	68
#define TWHC	69
#define USNT	70
#define KALLATE_NVD2200PS	71
#define VIDO_B01	72
#define LG_MULTIX	73
#define ENKEL	74
#define YT_PELCOD	75
#define HIKVISION	76
#define PE60	77
#define LiAo	78
#define NK16	79
#define DaLi	80
#define HN_4304	81
#define VIDEOTEC	82
#define HNDCB	83
#define Lion_2007	84
#define LG_LVC_C372	85
#define Gold_Video	86
#define NVD1600PS	87
#define TC615P	88
#define NANWANG	89
#define NANWANG_1602	90
#define SieMens	91
#define WVCS850	92
#define PHILIPS_2	93

```
#define  PHLIPS_3          94
#define  AD                95
#define  TYCO_AD          96
#define  VICON            97
#define  TKC676           98
#define  YAAN_NEW         99
#define  DL_NVS_1Z        100
#define  I3_Z1200         101
#define  I3_Z2200         102
```

3.2.7 串口配置结构

```
typedef struct{
    char sRemoteIP[16];          /* 远端 IP 地址 */
    char sLocalIP[16];           /* 本地 IP 地址 */
    char sLocalIPMask[16];       /* 本地 IP 地址掩码 */
    BYTE sUsername[NAME_LEN];    /* 用户名 */
    BYTE sPassword[PASSWD_LEN]; /* 密码 */
    BYTE byPPPMode;              /* PPP 模式, 0—主动, 1—被动 */
    BYTE byRedial;               /* 是否回拨 : 0-否,1-是 */
    BYTE byRedialMode;           /* 回拨模式,0-由拨入者指定,1-预置回拨号码 */
    BYTE byDataEncrypt;          /* 数据加密,0-否,1-是 */
    DWORD dwMTU;                 /* MTU */
    char sTelephoneNumber[PHONENUMBER_LEN]; /* 电话号码 */
}NET_DVR_PPPCFG,*LPNET_DVR_PPPCFG;

typedef struct{
    DWORD dwSize;                /* 此结构的大小 */
    DWORD dwBaudRate;            /* 波特率(bps), 0—50, 1—75, 2—110, 3—150, 4—300,
5—600, 6—1200, 7—2400, 8—4800, 9—9600, 10—19200, 11—38400, 12—57600, 13—
76800, 14—115.2k; */
    BYTE byDataBit;              /* 数据有几位 0—5 位, 1—6 位, 2—7 位, 3—8 位; */
    BYTE byStopBit;              /* 停止位 0—1 位, 1—2 位; */
    BYTE byParity;               /* 校验 0—无校验, 1—奇校验, 2—偶校验; */
    BYTE byFlowcontrol;          /* 0—无, 1—软流控,2-硬流控 */
    DWORD dwWorkMode;            /* 工作模式, 0—窄带传输 (232 串口用于 PPP 拨号), 1
—控制台 (232 串口用于参数控制), 2—透明通道 */
    NET_DVR_PPPCFG struPPPConfig;
}NET_DVR_RS232CFG,*LPNET_DVR_RS232CFG;
```

注：修改 RS232 串口使用方式后需要重启设备。

3.2.8 报警输入配置结构

```
typedef struct{
    DWORD dwSize;                /* 此结构的大小 */
```

```

BYTE sAlarmInName[NAME_LEN]; /* 名称 */
BYTE byAlarmType;           /* 报警器类型,0: 常开,1: 常闭 */
BYTE byAlarmInHandle;       /* 是否处理 */
NET_DVR_HANDLEEXCEPTION struAlarmHandleType; /* 处理方式 */
NET_DVR_SCHEDULETIME struAlarmTime[MAX_DAYS][MAX_TIMESEGMENT]; //布防时间
BYTE byRelRecordChan[MAX_CHANNUM]; /* 报警触发的录像通道,为 1 表示触发
该通道 */
BYTE byEnablePreset[MAX_CHANNUM]; /* 是否调用预置点 */
BYTE byPresetNo[MAX_CHANNUM]; /* 调用的云台预置点序号,一个报警输
入可以调用多个通道的云台预置点, 0xff 表示不调用预置点。*/
BYTE byEnableCruise[MAX_CHANNUM]; /* 是否调用巡航 */
BYTE byCruiseNo[MAX_CHANNUM]; /* 巡航 */
BYTE byEnablePtzTrack[MAX_CHANNUM]; /* 是否调用轨迹 */
BYTE byPTZTrack[MAX_CHANNUM]; /* 调用的云台的轨迹序号 */
}NET_DVR_ALARMINCFG,*LPNET_DVR_ALARMINCFG;

```

注：修改报警器类型和布防时间段后需要重启设备。

3.2.9 报警输出配置结构

```

typedef struct{
    DWORD dwSize; /* 此结构的大小 */
    BYTE sAlarmOutName[NAME_LEN]; /* 名称 */
    DWORD dwAlarmOutDelay; /* 输出保持时间(-1 为无限, 手动关闭) */
    NET_DVR_SCHEDULETIME struAlarmOutTime[MAX_DAYS][MAX_TIMESEGMENT]; /* 报警
输出激活时间段 */
}NET_DVR_ALARMOUTCFG,*LPNET_DVR_ALARMOUTCFG;

```

注：修改激活时间段后需要重启设备。

3.2.10 本地预览参数配置结构

```

typedef struct{
    DWORD dwSize; /* 此结构的大小 */
    BYTE byPreviewNumber; /* 预览数目,0-1 画面,1-4 画面,2-9 画面,3-16 画面,0xff:最大画面
*/
    BYTE byEnableAudio; /* 是否声音预览,0-不预览,1-预览 */
    WORD wSwitchTime; /* 切换时间,0-不切换,1-5s,2-10s,3-20s,4-60s,5-120s,6-300s */
    BYTE bySwitchSeq[MAX_WINDOW]; /* 切换顺序,如果 lSwitchSeq[i]为 0xff 表示不用*/
}NET_DVR_PREVIEWCFG,*LPNET_DVR_PREVIEWCFG;

```

3.2.11 视频输出配置结构

```

typedef struct {

```

```

    WORD wResolution;          /* 分辨率 */
    WORD wFreq;                 /* 刷新频率 */
    DWORD dwBrightness;        /* 亮度 */
}NET_DVR_VGAPARA;

typedef struct{
    WORD wDisplayLogo;          /* 显示视频通道号 */
    WORD wDisplayOsd;           /* 显示时间 */
}NET_DVR_MATRIXPARA;

typedef struct{
    BYTE byVideoFormat;         /* 输出制式,0-PAL,1-N */
    BYTE byMenuAlphaValue;      /* 菜单与背景图象对比度 */
    WORD wScreenSaveTime;       /* 屏幕保护时间 */
    WORD wVOffset;              /* 视频输出偏移 */
    WORD wBrightness;           /* 视频输出亮度 */
    BYTE byStartMode;           /* 启动后视频输出模式(0:菜单,1:预览) */
    char reservedData;
}NET_DVR_VOOUT;

typedef struct {
    DWORD dwSize;
    NET_DVR_VOOUT struVOOut[MAX_VIDEOOUT];
    NET_DVR_VGAPARA struVGAPara[MAX_VGA]; /* VGA 参数 */
    NET_DVR_MATRIXPARA struMatrixPara;    /* MATRIX 参数 */
}NET_DVR_VIDEOOUT,*LPNET_DVR_VIDEOOUT;

```

3.2.12 用户参数配置结构

```

typedef struct{
    BYTE sUserName[NAME_LEN];    /* 用户名 最大 16 位 */
    BYTE sPassword[PASSWD_LEN]; /* 密码 */
    DWORD dwLocalRight[MAX_RIGHT]; /* 权限 */
                                   /*数组 0: 本地控制云台*/
                                   /*数组 1: 本地手动录象*/
                                   /*数组 2: 本地回放*/
                                   /*数组 3: 本地设置参数*/
                                   /*数组 4: 本地查看状态、日志*/
                                   /*数组 5: 本地高级操作(升级, 格式化, 重启, 关机)*/
    DWORD dwRemoteRight[MAX_RIGHT]; /* 权限 */
                                   /*数组 0: 远程控制云台*/
                                   /*数组 1: 远程手动录象*/
                                   /*数组 2: 远程回放*/
                                   /*数组 3: 远程设置参数*/
                                   /*数组 4: 远程查看状态、日志*/
                                   /*数组 5: 远程高级操作(升级, 格式化, 重启, 关机)*/
                                   /*数组 6: 远程发起语音对讲*/

```

```

/*数组 7: 远程预览*/
/*数组 8: 远程请求报警上传、报警输出*/
/*数组 9: 远程控制, 本地输出*/
/*数组 10: 远程控制串口*/
char sUserIP[16]; /* 用户 IP 地址(为 0 时表示允许任何地址) */
BYTE byMACAddr[MACADDR_LEN]; /* 物理地址 */
}NET_DVR_USER_INFO,*LPNET_DVR_USER_INFO;

typedef struct{
    DWORD dwSize;
    NET_DVR_USER_INFO struUser[MAX_USERNUM];
}NET_DVR_USER,*LPNET_DVR_USER;

```

DVR 用户参数扩展, 本地、远程回放, 远程预览权限细化到通道。

```

typedef struct{
    BYTE sUserName[NAME_LEN]; /* 用户名 */
    BYTE sPassword[PASSWD_LEN]; /* 密码 */
    DWORD dwLocalRight[MAX_RIGHT]; /* 权限 */
    /*数组 0: 本地控制云台*/
    /*数组 1: 本地手动录象*/
    /*数组 2: 本地回放*/
    /*数组 3: 本地设置参数*/
    /*数组 4: 本地查看状态、日志*/
    /*数组 5: 本地高级操作(升级, 格式化, 重启, 关机)*/
    DWORD dwLocalPlaybackRight; /* 本地可以回放的通道 bit0 -- channel 1*/
    DWORD dwRemoteRight[MAX_RIGHT]; /* 权限 */
    /*数组 0: 远程控制云台*/
    /*数组 1: 远程手动录象*/
    /*数组 2: 远程回放 */
    /*数组 3: 远程设置参数*/
    /*数组 4: 远程查看状态、日志*/
    /*数组 5: 远程高级操作(升级, 格式化, 重启, 关机)*/
    /*数组 6: 远程发起语音对讲*/
    /*数组 7: 远程预览*/
    /*数组 8: 远程请求报警上传、报警输出*/
    /*数组 9: 远程控制, 本地输出*/
    /*数组 10: 远程控制串口*/
    DWORD dwNetPreviewRight; /* 远程可以预览的通道 bit0 -- channel 1*/
    DWORD dwNetPlaybackRight; /* 远程可以回放的通道 bit0 -- channel 1*/
    char sUserIP[16]; /* 用户 IP 地址(为 0 时表示允许任何地址) */
    BYTE byMACAddr[MACADDR_LEN]; /* 物理地址 */
}NET_DVR_USER_INFO_EX,*LPNET_DVR_USER_INFO_EX;

typedef struct{
    DWORD dwSize;
    NET_DVR_USER_INFO_EX struUser[MAX_USERNUM];
}NET_DVR_USER_EX,*LPNET_DVR_USER_EX;

```

3.2.13 异常参数配置结构

```
typedef struct{
    DWORD dwSize;
    NET_DVR_HANDLEEXCEPTION struExceptionHandleType[MAX_EXCEPTIONNUM];
    /*数组 0-盘满,1- 硬盘出错,2-网线断,3-局域网内 IP 地址冲突,4-非法访问, 5-输入/输出视频
    制式不匹配*/
}NET_DVR_EXCEPTION,*LPNET_DVR_EXCEPTION;
```

3.2.14 帧信息配置结构

```
//ATM类型
#define NCR      0
#define DIEBOLD 1
#define WINCOR_NIXDORF 2
#define SIEMENS 3
#define OLIVETTI 4
#define FUJITSU 5
#define HITACHI 6
#define SMI      7
#define IBM      8
#define BULL 9
#define YiHua 10
#define LiDe 11
#define GDYT 12
#define Mini_Banl 13
#define GuangLi 14
#define DongXin 15
#define ChenTong 16
#define NanTian 17
#define XiaoXing 18
#define GZYY 19
#define QHTLT 20
#define DRS918 21
#define KALATEL 22
#define NCR_2 23
#define NXS 24

typedef struct{
    BYTE code[12]; /* 代码 */
}NET_DVR_FRAMEYPECODE;

typedef struct{
    DWORD dwSize;
    char sATMIP[16]; /* ATM IP 地址 */
    DWORD dwATMType; /* ATM 类型 */
    DWORD dwInputMode; /* 输入方式 */
}
```

```

    DWORD dwFrameSignBeginPos;          /* 报文标志位的起始位置*/
    DWORD dwFrameSignLength;            /* 报文标志位的长度 */
    BYTE  byFrameSignContent[12];        /* 报文标志位的内容 */
    DWORD dwCardLengthInfoBeginPos;      /* 卡号长度信息的起始位置 */
    DWORD dwCardLengthInfoLength;        /* 卡号长度信息的长度 */
    DWORD dwCardNumberInfoBeginPos;      /* 卡号信息的起始位置 */
    DWORD dwCardNumberInfoLength;        /* 卡号信息的长度 */
    DWORD dwBusinessTypeBeginPos;        /* 交易类型的起始位置 */
    DWORD dwBusinessTypeLength;          /* 交易类型的长度 */
    NET_DVR_FRAMETYPECODE frameTypeCode[10];/* 类型 */
}NET_DVR_FRAMEFORMAT,*LPNET_DVR_FRAMEFORMAT;

```

3.2.15 编码端字符叠加配置结构

```

typedef struct
{
    DWORD dwShowString;                /* 预览的图象上是否显示字符,0-不显示,1-显示 区域
                                         704*576,单个字符的大小为 32*32 */
    WORD wStringSize;                  /* 该行字符的长度，不能大于 44 个字符 */
    WORD wShowStringTopLeftX;          /* 字符显示位置的 x 坐标 */
    WORD wShowStringTopLeftY;          /* 字符名称显示位置的 y 坐标 */
    char sString[44];                  /* 要显示的字符内容 */
}NET_DVR_SHOWSTRINGINFO,*LPNET_DVR_SHOWSTRINGINFO;

```

```

typedef struct
{
    DWORD dwSize;
    NET_DVR_SHOWSTRINGINFO struStringInfo[MAX_STRINGNUM]; /*要显示的字符内
                                                             容 */
}NET_DVR_SHOWSTRING,*LPNET_DVR_SHOWSTRING;

```

```

typedef struct tagSHOWSTRING_EX
{
    DWORD dwSize;
    NET_DVR_SHOWSTRINGINFO struStringInfo[MAX_STRINGNUM_EX]; /* 要显示的字符内容 */
}NET_DVR_SHOWSTRING_EX,*LPNET_DVR_SHOWSTRING_EX;

```

3.2.16 6001D/F 配置结构

```

/*连接的通道配置*/
typedef struct {
    char sDVRIP[16];                  /* DVR IP 地址 */
    WORD wDVRPort;                    /* 端口号 */
    BYTE sUserName[NAME_LEN];          /* 用户名 */
    BYTE sPassword[PASSWD_LEN];        /* 密码 */
}

```

```

    BYTE byChannel;          /* 通道号 */
    BYTE byLinkMode;         /* 连接模式 */
    BYTE byLinkType;         /* 连接类型 0—主码流 1—子码流 */
}NET_DVR_DECCHANINFO, *LPNET_DVR_DECCHANINFO;

/*每个解码通道的配置*/
typedef struct {
    BYTE byPoolChans;        /*每路解码通道上的循环通道数量, 最多 4 通道 0 表示没有解码*/
    NET_DVR_DECCHANINFO struchanConInfo[MAX_DECPOOLNUM];
    BYTE byEnablePoll;       /*是否轮巡 0-否 1-是*/
    BYTE byPoolTime;         /*轮巡时间 0-保留 1-10 秒 2-15 秒 3-20 秒 4-30 秒 5-45 秒 6-1 分钟 7-2 分钟 8-5 分钟 */
}NET_DVR_DECINFO, *LPNET_DVR_DECINFO;

/*整个设备解码配置*/
typedef struct {
    DWORD dwSize;
    DWORD dwDecChanNum;      /*解码通道的数量*/
    NET_DVR_DECINFO struDecInfo[MAX_DECNUM];
}NET_DVR_DECCFG, *LPNET_DVR_DECCFG;

/* 解码设备透明通道设置 */
typedef struct {
    DWORD dwEnableTransPort; /* 是否启动透明通道 0—不启用 1—启用*/
    char sDecoderIP[16];     /* DVR IP 地址 */
    WORD wDecoderPort;       /* 端口号 */
    WORD wDVRTransPort;      /* 配置前端 DVR 是从 485/232 输出, 1 表示 232 串口, 2 表示 485 串口 */
    char cReserve[4];
}NET_DVR_PORTINFO, *LPNET_DVR_PORTINFO;

typedef struct {
    DWORD dwSize;
    NET_DVR_PORTINFO struTransPortInfo[MAX_TRANSPARENTNUM]; /* 数组 0 表示 232 数组 1 表示 485 */
}NET_DVR_PORTCFG, *LPNET_DVR_PORTCFG;

/* 控制网络文件回放 */
typedef struct {
    DWORD dwSize;
    char sDecoderIP[16];     /* DVR IP 地址 */
    WORD wDecoderPort;       /* 端口号 */
    WORD wLoadMode;          /* 回放下载模式 1—按名字 2—按时间 */
    union
    {
        BYTE byFile[100];   /* 回放的文件名 */
        struct
        {
            DWORD dwChannel;

```

```

        BYTE sUserName[NAME_LEN];/*请求视频用户名*/
        BYTE sPassword[PASSWD_LEN]; /* 密码 */
        NET_DVR_TIME struStartTime; /* 按时间回放的开始时间 */
        NET_DVR_TIME struStopTime; /* 按时间回放的结束时间 */
    }bytime;
    }mode_size;
}NET_DVR_PLAYREMOTEFILE,*LPNET_DVR_PLAYREMOTEFILE;

/*当前设备解码连接状态*/
typedef struct {
    DWORD dwWorkType; /*工作方式：1：轮巡、2：动态连接解码、3：文件回放下载 4：
按时间回放下载*/
    char sDVRIP[16]; /*连接的设备 ip*/
    WORD wDVRPort; /*连接端口号*/
    BYTE byChannel; /* 通道号 */
    BYTE byLinkMode; /* 连接模式 */
    DWORD dwLinkType; /*连接类型 0—主码流 1—子码流*/
    union
    {
        struct
        {
            BYTE sUserName[NAME_LEN];/*请求视频用户名*/
            BYTE sPassword[PASSWD_LEN]; /* 密码 */
            char cReserve[52];
        }userInfo;
        struct
        {
            BYTE fileName[100];
        }fileInfo;
        struct
        {
            DWORD dwChannel;
            BYTE sUserName[NAME_LEN]; /*请求视频用户名*/
            BYTE sPassword[PASSWD_LEN]; /* 密码 */
            NET_DVR_TIME struStartTime; /* 按时间回放的开始时间 */
            NET_DVR_TIME struStopTime; /* 按时间回放的结束时间 */
        }timeInfo;
    }objectInfo;
}NET_DVR_DECCHANSTATUS,*LPNET_DVR_DECCHANSTATUS;

typedef struct {
    DWORD dwSize;
    NET_DVR_DECCHANSTATUS struDecState[MAX_DECNUM];
}NET_DVR_DECSTATUS,*LPNET_DVR_DECSTATUS;

```

3.2.17 HS 设备辅助输出配置结构

```

typedef struct tagAUXOUT{
    DWORD dwSize;

```

```

    DWORD dwAlarmOutChan; /* 选择报警弹出大画面的输出通道: 0:主输出/1:辅 1/2:辅 2/3:辅
3/4:辅 4 */
    DWORD dwAlarmChanSwitchTime; /* 报警通道切换时间: 1:1 秒 - 10:10 秒 */
    DWORD dwAuxSwitchTime[MAX_AUXOUT]; /* 辅助输出切换时间: 0- 不切
换,1-5s,2-10s,3-20s,4-30s,5-60s,6-120s,7-300s */
    BYTE byAuxOrder[MAX_AUXOUT][MAX_WINDOW]; /* 辅助输出预览顺序, 0xff 表
示相应的窗口不预览 */
}NET_DVR_AUXOUTCFG, *LPNET_DVR_AUXOUTCFG;

```

3.2.18 网络应用参数（NTP/DDNS/EMAIL）

```

typedef struct tagNTPPARA{
    BYTE sNTPServer[64]; /* Domain Name or IP addr of NTP server */
    WORD wInterval; /* adjust time interval(hours) */
    BYTE byEnableNTP; /* enable NPT client 0-no, 1-yes*/
    signed char cTimeDifferenceH; /* 与国际标准时间的时差 -12 ... +13 */
    signed char cTimeDifferenceM;
    BYTE res[11];
}NET_DVR_NTPPARA, *LPNET_DVR_NTPPARA;

typedef struct tagDDNSPARA{
    BYTE sUsername[NAME_LEN]; /* DDNS账号用户名/密码 */
    BYTE sPassword[PASSWD_LEN];
    BYTE sDomainName[64]; /* 域名 */
    BYTE byEnableDDNS; /*是否应用 0-否, 1-是*/
    BYTE res[15];
}NET_DVR_DDNSPARA, *LPNET_DVR_DDNSPARA;

typedef struct tagNET_DDNSPARA_EX{
    BYTE byHostIndex; /* 0-Hikvision DNS(保留) 1-Dyndns 2-
PeanutHull(花生壳) */
    BYTE byEnableDDNS; /*是否应用DDNS 0-否, 1-是*/
    WORD wDDNSPort; /* DDNS端口号 */
    BYTE sUsername[NAME_LEN]; /* DDNS用户名*/
    BYTE sPassword[PASSWD_LEN]; /* DDNS密码 */
    BYTE sDomainName[MAX_DOMAIN_NAME]; /* 设备配备的域名地址 */
    BYTE sServerName[MAX_DOMAIN_NAME]; /* DDNS 对应的服务器地址, 可以是IP地址
或域名 */
    BYTE byRes[16];
}NET_DVR_DDNSPARA_EX, *LPNET_DVR_DDNSPARA_EX;

typedef struct tagNETAPPCFG{
    DWORD dwSize;
    char sDNSIp[16]; /* DNS服务器地址 */
    NET_DVR_NTPPARA struNtpClientParam; /* NTP参数 */
    NET_DVR_DDNSPARA struDDNSClientParam; /* DDNS参数 */
    BYTE res[464]; /* 保留 */
}NET_DVR_NETAPPCFG, *LPNET_DVR_NETAPPCFG;

```

```
typedef struct tagEMAILCFG
{
    /* 12 bytes */
    DWORD dwSize;
    char    sUserName[32];
    char    sPassWord[32];
    char    sFromName[32];          /* Sender */
    char    sFromAddr[48];          /* Sender address *///字符串中的第一个字符和最后一个字符不能是"@",并且字符串中要有"@"字符
    char    sToName1[32];           /* Receiver1 */
    char    sToName2[32];           /* Receiver2 */
    char    sToAddr1[48];           /* Receiver address1 */
    char    sToAddr2[48];           /* Receiver address2 */
    char    sEmailServer[32];       /* Email server address */
    BYTE    byServerType;           /* Email server type: 0-SMTP*/
    BYTE    byUseAuthen;            /* Email server authentication method: 1-enable, 0-disable */
    BYTE    byAttachment;           /* enable attachment */
    BYTE    byMailinterval;         /* mail interval 0-2s, 1-3s, 2-4s. 3-5s*/
} NET_DVR_EMAILCFG, *LPNET_DVR_EMAILCFG;
```

注：修改DDNS、NTP和EMAIL参数后需要重启设备。

3.2.19 NFS 配置

```
typedef struct tag_NFSCFG{
    DWORD    dwSize;
    struct{
        char sNfsHostIPAddr[16];
        BYTE sNfsDirectory[PATHNAME_LEN];          // PATHNAME_LEN = 128
    }struNfsDiskParam[MAX_NFS_DISK]; // MAX_NFS_DISK=8
}NET_DVR_NFSCFG, *LPNET_DVR_NFSCFG;
```

注：修改 NAS 参数后需要重启设备。

3.2.20 IP 快球配置

```
typedef struct tagPOINT_FRAME
{
    int xTop; // 方框起始点的x坐标
    int yTop; // 方框结束点的y坐标
    int xBottom; // 方框结束点的x坐标
    int yBottom; //方框结束点的y坐标
    int bCounter; //保留
}NET_DVR_POINT_FRAME, *LPNET_DVR_POINT_FRAME;
```

```
typedef struct tagCRUISE_RET{
    struct{
        BYTE    PresetNum; //预置点
    }
};
```

```

        BYTE    Dwell;           //停留时间
        BYTE    Speed;          //速度
        BYTE    Reserve; //保留
    }struCruisePoint[32];        //最大支持 32 个巡航点
}NET_DVR_CRUISE_RET, *LPNET_DVR_CRUISE_RET;

```

3.2.21 多路解码器参数

```

typedef struct tagDVRCHANINFO{
    char        sDVRIP[16];      /* DVR IP 地址 */
    WORD        wDVRPort;        /* 端口号 */
    BYTE        byChannel;       /* 通道号 */
    BYTE        byTransProtocol; /* 传输协议类型 */
    BYTE        byTransMode;     /* 传输码流模式 0—主码流 1—子码流*/
    BYTE        byRes[3];
    BYTE        sUserName[NAME_LEN]; /* 监控主机登陆帐号 */
    BYTE        sPassword[PASSWD_LEN]; /* 监控主机密码 */
}NET_DVR_MATRIX_DECINFO, *LPNET_DVR_MATRIX_DECINFO;

//启动/停止动态解码
typedef struct tagDYNAMIC_TEST{
    DWORD dwSize;
    NET_DVR_MATRIX_DECINFO struDecChanInfo; /* 动态解码通道信息 */
}NET_DVR_MATRIX_DYNAMIC_DEC, *LPNET_DVR_MATRIX_DYNAMIC_DEC;

/*连接的通道配置*/
typedef struct tagMATRIX_CHAN_INFO
{
    DWORD dwEnable; /* 是否启用 0—否 1—启用*/
    NET_DVR_MATRIX_DECINFO struDecChanInfo; /* 轮循解码通道信息 */
}NET_DVR_MATRIX_CHAN_INFO, *LPNET_DVR_MATRIX_CHAN_INFO;

/*每个解码通道的配置*/
typedef struct tagLOOPDECINFO
{
    DWORD dwSize;
    DWORD dwPoolTime; /*轮巡时间 */
    NET_DVR_MATRIX_CHAN_INFO struchanConInfo[MAX_CYCLE_CHAN];
}NET_DVR_MATRIX_LOOP_DECINFO, *LPNET_DVR_MATRIX_LOOP_DECINFO;

typedef struct tagDEC_MATRIX_CHAN_INFO
{
    DWORD dwSize;
    NET_DVR_MATRIX_DECINFO struDecChanInfo; /* 解码通道信息 */
    DWORD dwDecState; /* 0-动态解码 1—循环解码 2—按文件回放 3—按时间回放 */
    NET_DVR_TIME StartTime; /* 按时间回放开始时间 */
    NET_DVR_TIME StopTime; /* 按时间回放停止时间 */
    char sFileName[128]; /* 按文件回放文件名 */
}

```

```

}NET_DVR_MATRIX_DEC_CHAN_INFO,*LPNET_DVR_MATRIX_DEC_CHAN_INFO;

typedef struct  tagMATRIX_DEC_CHAN_STATUS
{
    DWORD    dwSize;
    DWORD    dwIsLinked;          /* 解码通道状态 0—休眠 1—正在连接 2—已连接 3-
正在解码 */
    DWORD    dwStreamCpRate;      /* Stream copy rate, X kbits/second */
    char     cRes[64];           /* 保留 */
}NET_DVR_MATRIX_DEC_CHAN_STATUS,*LPNET_DVR_MATRIX_DEC_CHAN_STATUS;

```

第四章 函数说明

4.1 客户端函数

4.1.1 初始化

BOOL NET_DVR_Init()

功能：初始化 SDK

返回值：TRUE 表示成功，FALSE 表示失败。

BOOL NET_DVR_Cleanup()

功能：释放 SDK 资源

返回值：TRUE 表示成功，FALSE 表示失败。

int NET_DVR_IsSupport()

功能：判断运行客户端的机器是否支持

返回值：//1—9 位分别表示以下信息（位与是 TRUE）表示支持；

```
#define NET_DVR_SUPPORT_DDRAW      0x01 //支持DIRECTDRAW，如果不支持，则播
                                     放器不能工作；
#define NET_DVR_SUPPORT_BLT        0x02 //显卡支持BLT操作，如果不支持，则播
                                     放器不能工作；
#define NET_DVR_SUPPORT_BLTFOURCC  0x04 //显卡BLT支持颜色转换，如果不支持，
                                     播放器会用软件方法作RGB转换；
#define NET_DVR_SUPPORT_BLTSHRINKX 0x08 //显卡BLT支持X轴缩小；如果不支持，
                                     系统会用软件方法转换；
#define NET_DVR_SUPPORT_BLTSHRINKY 0x10 //显卡BLT支持Y轴缩小；如果不支持，
                                     系统会用软件方法转换；
#define NET_DVR_SUPPORT_BLTSTRETCHX 0x20 //显卡BLT支持X轴放大；如果不支持，
                                     系统会用软件方法转换；
#define NET_DVR_SUPPORT_BLTSTRETCHY 0x40 //显卡BLT支持Y轴放大；如果不支持，
                                     系统会用软件方法转换；
#define NET_DVR_SUPPORT_SSE        0x80 //CPU支持SSE指令，Intel Pentium3以
                                     上支持SSE指令；
#define NET_DVR_SUPPORT_MMX        0x100 //CPU支持MMX指令集，Intel Pentium3
                                     以上支持SSE指令；
```

BOOL NET_DVR_SetConnectTime(DWORD dwWaitTime,DWORD dwTryTimes)

功能：设置连接超时时间和连接尝试次数

参数说明：

dwWaitTime: 超时时间, 单位:毫秒 (>300, <60*1000)

dwTryTimes: 连接尝试次数(暂时保留)

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_SetDVRMessage(UINT nMessage,HWND hWnd)

功能: 设置接收硬盘录像机消息的窗口句柄

参数说明:

nMessage: 消息

hWnd: 接收消息的窗口句柄

//异常类型

```
#define EXCEPTION_EXCHANGE          0x8000 //用户交互时异常
#define EXCEPTION_AUDIOEXCHANGE     0x8001 //语音对讲异常
#define EXCEPTION_ALARM              0x8002 //报警异常
#define EXCEPTION_PREVIEW            0x8003 //网络预览异常
#define EXCEPTION_SERIAL              0x8004 //透明通道异常
#define EXCEPTION_RECONNECT          0x8005 //预览时重连
```

注: 消息处理函数如下:

```
void OnMyCommand(WPARAM wParam, LPARAM lParam)
{
    char tempbuf[256];
    DWORD dwType = wParam; //异常类型
    LONG lIndex = lParam; //出现异常的相应类型的句柄
    ZeroMemory(tempbuf,256);
    switch(dwType)
    {
        case EXCEPTION_AUDIOEXCHANGE://语音对讲时网络异常
            sprintf(tempbuf,"语音对讲时网络异常!!!");
            TRACE("%s",tempbuf);
            //TODO: 关闭语音对讲
            break;
        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_PREVIEW: //网络预览时异常
            sprintf(tempbuf,"网络预览时网络异常!!!");
            TRACE("%s",tempbuf);
            //TODO: 关闭网络预览
            break;
        case EXCEPTION_SERIAL: //透明通道传输时异常
            sprintf(tempbuf,"透明通道传输时网络异常!!!");
            TRACE("%s",tempbuf);
            //TODO: 关闭透明通道
            break;
        case EXCEPTION_RECONNECT: //预览时重连
            break;
        default:
```

```

        break;
    }
}

```

BOOL NET_DVR_SetDVRMessCallBack(BOOL (CALLBACK *fMessCallBack)(LONG lCommand,char *sDVRIP,char *pBuf,DWORD dwBufLen))

功能：设置接收硬盘录像机消息的回调函数（以 IP 地址区分设备）

参数说明：

fMessCallBack：消息回调函数，像硬盘录像机主动发起的请求；以及通过报警上传通道收到的报警信息，都通过此函数回调。

lCommand：消息的类型，定义如下：

```

#define COMM_ALARM          0x1100  //上传报警信息
#define COMM_TRADEINFO      0x1500  //ATMDVR 上传交易信息

```

sDVRIP：服务器的 IP 地址

pBuf：存放信息的缓冲区，不同的类型分别指向不同的结构

dwBufLen：缓冲区的大小

返回值：TRUE 表示成功，FALSE 表示失败。

BOOL NET_DVR_SetDVRMessCallBack_EX(BOOL (CALLBACK *fMessCallBack_EX)(LONG lCommand,LONG lUserID,char *pBuf,DWORD dwBufLen))

功能：设置接收硬盘录像机消息的回调函数(以 ID 号区分设备)

参数说明：

fMessCallBack_EX：消息回调函数，像硬盘录像机主动发起的请求；以及通过报警上传通道收到的报警信息，都通过此函数回调。

lCommand：消息的类型

lUserID：用户登录 ID，NET_DVR_Login 的返回值

pBuf：存放信息的缓冲区，不同的类型分别指向不同的结构

dwBufLen：缓冲区的大小

返回值：TRUE 表示成功，FALSE 表示失败。

NET_DVR_API BOOL __stdcall NET_DVR_SetDVRMessCallBack_NEW(BOOL (CALLBACK *fMessCallBack_NEW)(LONG lCommand,char *sDVRIP,char *pBuf,DWORD dwBufLen, WORD dwLinkDVRPort));

功能:注册接收硬盘录像机消息的回调函数(以 DVRIP 和连接 DVR 端口回调)

返回值:

TRUE 表示成功,FALSE 表示失败

NET_DVR_API BOOL __stdcall NET_DVR_SetDVRMessageCallBack(BOOL (CALLBACK *fMessageCallBack)(LONG lCommand,char *sDVRIP,char *pBuf,DWORD dwBufLen, DWORD dwUser), DWORD dwUser);

功能:注册接收硬盘录像机消息的回调函数(带有用户数据)

返回值:

TRUE 表示成功,FALSE 表示失败

DWORD NET_DVR_GetSDKVersion()

功能：获取 SDK 版本号

返回值：2 个高字节表示主版本，2 个低字节表示次版本，比如 0x00020001 代表版本 2.1

BOOL NET_DVR_GetDVRIPByResolveSvr(char *sServerIP, WORD wServerPort BYTE *sDVRName,WORD wDVRNameLen,BYTE *sDVRSerialNumber,WORD

wDVRSerialLen,char* sGetIP)

功能：通过解析服务器, 获取硬盘录像机的动态 IP 地址

参数说明：

sServerIP: 解析服务器的 IP 地址

wServerPort: 解析服务器的端口号, 我们提供的解析服务器端口号为 7071

sDVRName: 硬盘录像机的名称, 可以为 NULL

wDVRNameLen: 硬盘录像机名称的长度

sDVRSerialNumber: 硬盘录像机的序列号, 可以为 NULL

wDVRSerialLen: 硬盘录像机序列号的长度

sGetIP: 保存获取到的 IP 的指针

返回值: TRUE 表示成功, FALSE 表示失败。

注: 硬盘录像机的名称和硬盘录像机的序列号不能同时为空。

4.1.2 设置显示模式

BOOL NET_DVR_SetShowMode(DWORD dwShowType,COLORREF colorKey)

功能：设置播放器显示模式, 在播放之前设置。播放器有两种显示模式：普通模式和 OVERLAY 方式, 使用 OVERLAY 模式的优点是：大部分显卡都支持 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, 可以大大降低 CPU 利用率并提高画面质量(相对于软件实现缩放、颜色转换)。缺点是：同时只能播放一路图象, 不能实现大规模集中监控。

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, 它将自动改成普通的模式, 并不返回 FALSE, 一些常用的播放器, 例如我们卡的预览都可能使用了 OVERLAY 表面, 同样, 如果我们的 SDK 中使用了 OVERLAY 表面, 那么其他的程序将不能再使用 OVERLAY 表面。

参数说明：

dwShowType: 显示模式

colorKey: 用户设置的透明色, 在 OVERLAY 模式下需要设置, 透明色相当于一层透视膜, 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。colorKey 是一个 32 位的值 0x00rrggbb, 最高字节为 0, 后三个字节分别表示 r、g、b 的值。

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.3 启动/停止监听程序

BOOL NET_DVR_StartListen (char *sLocalIP,WORD wLocalPort)

功能：启动监听程序, 监听硬盘录像机发起的请求,接收硬盘录像机的信息

参数说明：

sLocalIP: PC 机的 IP 地址, 如果为 NULL, SDK 将自动获取 PC 机的 IP 地址, 如果 PC 机有多个 IP 地址, 可以指定一个 IP 地址进行监听。

WLocalPort: 本地监听端口号, 由用户设置,

返回值: TRUE 表示成功, FALSE 表示失败。

注: 要使客户端能够收到硬盘录像机主动发过来的信息, 必须将硬盘录像机的网络配置中的“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, “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。

BOOL NET_DVR_StopListen()

功能：停止监听程序

4.1.4 获取错误代码

DWORD NET_DVR_GetLastError()

功能：获取错误代码

返回值：

#define NET_DVR_NOERROR	0 //没有错误
#define NET_DVR_PASSWORD_ERROR	1 //用户名密码错误
#define NET_DVR_NOENOUGHPRI	2 //权限不足
#define NET_DVR_NOINIT	3 //没有初始化
#define NET_DVR_CHANNEL_ERROR	4 //通道号错误
#define NET_DVR_OVER_MAXLINK	5 //连接到 DVR 的客户端个数超过最大
#define NET_DVR_VERSIONNOMATCH	6 //版本不匹配
#define NET_DVR_NETWORK_FAIL_CONNECT	7 //连接服务器失败
#define NET_DVR_NETWORK_SEND_ERROR	8 //向服务器发送失败
#define NET_DVR_NETWORK_RECV_ERROR	9 //从服务器接收数据失败
#define NET_DVR_NETWORK_RECV_TIMEOUT	10 //从服务器接收数据超时
#define NET_DVR_NETWORK_ERRORDATA	11 //传送的数据有误
#define NET_DVR_ORDER_ERROR	12 //调用次序错误
#define NET_DVR_OPERNOPERMIT	13 //无此权限
#define NET_DVR_COMMANDTIMEOUT	14 //DVR 命令执行超时
#define NET_DVR_ERRORSERIALPORT	15 //串口号错误
#define NET_DVR_ERRORALARMPORT	16 //报警端口错误
#define NET_DVR_PARAMETER_ERROR	17 //参数错误
#define NET_DVR_CHAN_EXCEPTION	18 //服务器通道处于错误状态
#define NET_DVR_NODISK	19 //没有硬盘
#define NET_DVR_ERRORDISKNUM	20 //硬盘号错误
#define NET_DVR_DISK_FULL	21 //服务器硬盘满
#define NET_DVR_DISK_ERROR	22 //服务器硬盘出错
#define NET_DVR_NOSUPPORT	23 //服务器不支持
#define NET_DVR_BUSY	24 //服务器忙
#define NET_DVR_MODIFY_FAIL	25 //服务器修改不成功
#define NET_DVR_PASSWORD_FORMAT_ERROR	26 //密码输入格式不正确
#define NET_DVR_DISK_FORMATING	27 //硬盘正在格式化，不能启动操作
#define NET_DVR_DVRNORESOURCE	28 //DVR 资源不足
#define NET_DVR_DVROPRATEFAILED	29 //DVR 操作失败
#define NET_DVR_OPENHOSTSOUND_FAIL	30 //打开 PC 声音失败
#define NET_DVR_DVRVOICEOPENED	31 //服务器语音对讲被占用
#define NET_DVR_TIMEINPUTERROR	32 //时间输入不正确
#define NET_DVR_NOSPECFILE	33 //回放时服务器没有指定的文件
#define NET_DVR_CREATEFILE_ERROR	34 //创建文件出错
#define NET_DVR_FILEOPENFAIL	35 //打开文件出错
#define NET_DVR_OPERNOTFINISH	36 //上次的操作还没有完成

```

#define NET_DVR_GETPLAYTIMEFAIL      37 //获取当前播放的时间出错
#define NET_DVR_PLAYFAIL              38 //播放出错
#define NET_DVR_FILEFORMAT_ERROR      39 //文件格式不正确
#define NET_DVR_DIR_ERROR             40 //路径错误
#define NET_DVR_ALLOC_RESOUCE_ERROR   41 //资源分配错误
#define NET_DVR_AUDIO_MODE_ERROR      42 //声卡模式错误
#define NET_DVR_NOENOUGH_BUF          43 //缓冲区太小
#define NET_DVR_CREATESOCKET_ERROR    44 //创建 SOCKET 出错
#define NET_DVR_SETSOCKET_ERROR       45 //创建 SOCKET 出错
#define NET_DVR_MAX_NUM               46 //个数达到最大
#define NET_DVR_USERNOTEXIST          47 //用户不存在
#define NET_DVR_WRITEFLASHERROR       48 //写 FLASH 出错
#define NET_DVR_UPGRADEFAIL           49 //DVR 升级失败
#define NET_DVR_CARDHAVEINIT          50 //解码卡已经初始化过
#define NET_DVR_PLAYERFAILED          51 //播放器中错误
#define NET_DVR_MAX_USERNUM           52 //用户数达到最大
#define NET_DVR_GETLOCALIPANDMACFAIL  53 //获得客户端的 IP 地址或物理地
址失败
#define NET_DVR_NOENCODEING           54 //该通道没有编码
#define NET_DVR_IPMISMATCH            55 //IP 地址不匹配
#define NET_DVR_MACMISMATCH           56 //MAC 地址不匹配
#define NET_DVR_UPGRADELANGMISMATCH   57 //升级文件语言不匹配

```

4.1.5 用户注册

LONG NET_DVR_Login(char *sDVRIP,WORD wDVRPort,char *sUserName,char *sPassword ,LPNET_DVR_DEVICEINFO lpDeviceInfo)

功能：注册用户到硬盘录像机

参数说明：

sDVRIP：硬盘录像机的 IP 地址

wDVRPort：硬盘录像机的端口号

sUserName：登录的用户名

sPassword：用户密码

lpDeviceInfo：指向 NET_DVR_DEVICEINFO 结构的指针

返回值：-1 表示失败，其他值表示返回用户的 ID 值，该 ID 值是由 SDK 分配，每个用户 ID 值在客户端是唯一的。

BOOL NET_DVR_Logout(LONG lUserID)

功能：从硬盘录像机上注销某个用户

参数说明：

lUserID：用户 ID 值，由 NET_DVR_Login 返回

返回值：TRUE 表示成功，FALSE 表示失败。

注：在返回失败的情况下，SDK 仍会将该注册的用户信息注销。

4.1.6 图像预览

LONG NET_DVR_RealPlay(LONG IUserID,LPNET_DVR_CLIENTINFO lpClientInfo)

功能：启动图像实时预览

参数说明：

IUserID：用户登录 ID，NET_DVR_Login 的返回值

lpClientInfo：指向 NET_DVR_CLIENTINFO 结构的指针

返回值：-1 表示失败，其他值作为 NET_DVR_StopRealPlay 等函数的参数

BOOL NET_DVR_StopRealPlay(LONG IRealHandle)

功能：关闭图像预览功能

参数说明：

IRealHandle：NET_DVR_RealPlay 的返回值

返回值：TRUE 表示成功，FALSE 表示失败。

int NET_DVR_GetRealPlayerIndex(LONG IRealHandle)

功能：获取当前用来解码和显示的播放器句柄，可以通过该句柄来调用播放器SDK接口实现特定的功能

参数说明：

IRealHandle：NET_DVR_RealPlay 的返回值

例如使用下面两个接口：

BOOL PlayM4_GetBMP(LONG nPort,PBYTE pBitmap,DWORD nBufSize,DWORD* pBmpSize);

BOOL PlayM4_GetJPEG(LONG nPort,PBYTE pJpeg,DWORD nBufSize,DWORD* pJpegSize);

作如下调用：

PlayM4_GetBMP(NET_DVR_GetRealPlayerIndex(), pBitmap, nBufSize, pBmpSize)

PlayM4_GetJPEG(NET_DVR_GetRealPlayerIndex(), pJpeg, nBufSize, pJpegSize)

即可实现抓图 BMP 和 JPEG 到内存中。

4.1.6.1 视频参数

BOOL NET_DVR_ClientSetVideoEffect(LONG IRealHandle,DWORD dwBrightValue,DWORD dwContrastValue, DWORD dwSaturationValue,DWORD dwHueValue)

功能：调整视频参数

参数说明：

IRealHandle：NET_DVR_RealPlay()的返回值

dwBrightValue：亮度(取值为 1-10)

dwContrastValue：对比度(取值为 1-10)

dwSaturationValue：饱和度(取值为 1-10)

dwHueValue：色度(取值为 1-10)

返回值：TRUE 表示成功，FALSE 表示失败。

BOOL NET_DVR_ClientGetVideoEffect(LONG lRealHandle,DWORD *pBrightValue,DWORD *pContrastValue,DWORD *pSaturationValue,DWORD *pHueValue)

功能：获取视频参数

参数说明：

lRealHandle: NET_DVR_RealPlay()的返回值

pBrightValue: 存放亮度值的指针(取值为 1-10)

pContrastValue: 存放对比度值的指针(取值为 1-10)

pSaturationValue: 存放饱和度值的指针(取值为 1-10)

pHueValue: 存放色度值的指针(取值为 1-10)

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.6.2 叠加字符和图像

BOOL NET_DVR_RigisterDrawFun(LONG lRealHandle,void (CALLBACK* fDrawFun)(LONG lRealHandle,HDC hDc,DWORD dwUser),DWORD dwUser)

功能：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（获写字），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，而是播放器 DirectDraw 里的 Off-Screen 表面的 DC。

参数说明：

lRealHandle: NET_DVR_RealPlay()的返回值

fDrawFun: 回调函数句柄

dwUser: 用户数据

回调函数说明: void fDrawFun(long lRealHandle,HDC hDc,DWORD dwUser)

lRealHandle: NET_DVR_RealPlay()的返回值

hDc: OffScreen 表面设备上下文，你可以像操作显示窗口客户区 DC 一样操作它

dwUser: 用户数据，就是上面输入的用户数据

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.6.3 播放控制

BOOL NET_DVR_SetPlayerBufNumber(LONG lRealHandle,DWORD dwBufNum)

功能：设置播放器帧缓冲区的个数；

参数说明：

lRealHandle: NET_DVR_RealPlay()的返回值

dwBufNum: 要设置的播放器帧缓冲区最大的帧数，取值为（1—50），缓冲区越大，图象越流畅，缓冲区越小，实时性越好，若是复合流建议最小值设置成 6。

返回值: TRUE 表示成功, FALSE 表示失败。

注：这个函数要紧跟在 NET_DVR_RealPlay（）函数后调用

BOOL NET_DVR_ThrowBFrame(LONG lRealHandle,DWORD dwNum)

功能：设置解码时丢弃 B 帧的个数, dwNum :0 不丢,1-丢一个,2-丢 2 个

参数说明：

lRealHandle: NET_DVR_RealPlay()的返回值

m_ dwNum: 丢弃 B 帧的个数,取值为 0, 1, 2, 0-不丢, 1-丢 1 个 B 帧, 2-丢 2 个 B 帧；在多

路播放时，将 B 帧丢弃可以降低 CPU 的利用率，不过当一路时，最好不丢弃 B 帧

返回值：TRUE 表示成功，FALSE 表示失败。

BOOL NET_DVR_MakeKeyFrame(LONG lUserID, LONG lChannel)

功能：网络预览时动态产生一个关键帧

参数说明：

lUserID：用户登录 ID，NET_DVR_Login 的返回值

lChannel：通道号

返回值：TRUE 表示成功，FALSE 表示失败。

NET_DVR_API BOOL __stdcall NET_DVR_MakeKeyFrameSub(LONG lUserID, LONG lChannel);

功能：子码流生成一个 I 帧

参数说明：

lUserID：NET_DVR_Login()的返回值

lChannel：通道号

返回值：

TRUE 表示成功,FALSE 表示失败

4.1.6.4 声音控制

BOOL NET_DVR_SetAudioMode(DWORD dwMode)

功能：设置声音播放模式,是独占声卡模式还是共享声卡模式

参数说明：

dwMode：设置声卡的模式，1-独占声卡，单路音频模式；2-共享声卡，多路音频模式；

返回值：TRUE 表示成功，FALSE 表示失败。

注：该函数要在图像预览前设置。如果不设置，SDK 默认为独占声卡模式。当设置成独占模时，每一路的声音默认是关闭的；当设置成共享模式时，每一路的声音默认是打开的。

BOOL NET_DVR_OpenSound(LONG lRealHandle)

功能：独占声卡模式下打开声音；当有多路时，调用这个函数会停掉其他路的声音，一次只能预览一路声音；

参数说明：

lRealHandle：NET_DVR_RealPlay()的返回值

返回值：TRUE 表示成功，FALSE 表示失败。

BOOL NET_DVR_CloseSound()

功能：独占声卡模式下关闭声音

返回值：TRUE 表示成功，FALSE 表示失败。

BOOL NET_DVR_OpenSoundShare(LONG lRealHandle)

功能：共享声卡模式下打开一路声音预览

参数说明：

lRealHandle：NET_DVR_RealPlay()的返回值

返回值：TRUE 表示成功，FALSE 表示失败。

BOOL NET_DVR_CloseSoundShare(LONG lRealHandle)

功能：共享声卡模式下关闭一路声音预览

参数说明：

lRealHandle: NET_DVR_RealPlay()的返回值

返回值：TRUE 表示成功，FALSE 表示失败。

BOOL NET_DVR_Volume(LONG lRealHandle,WORD wVolume)

功能：调整音量

参数说明：

lRealHandle: NET_DVR_RealPlay()的返回值

wVolume:设置后的音量，取值范围：0-0xffff

返回值：TRUE 表示成功，FALSE 表示失败。

4.1.6.5 捕获

BOOL NET_DVR_SetRealDataCallBack(LONG lRealHandle,void(CALLBACK *fRealDataCallBack) (LONG lRealHandle, DWORD dwDataType, BYTE *pBuffer,DWORD dwBufSize,DWORD dwUser),DWORD dwUser)

功能：设置回调函数，用户自己处理客户端收到的数据

参数说明：

lRealHandle: NET_DVR_RealPlay()的返回值

fRealDataCallBack: 回调函数

dwUser: 用户数据

回调函数说明：

lRealHandle: NET_DVR_RealPlay()的返回值

dwDataType: 数据类型

#define NET_DVR_SYSHEAD 1 //系统头数据

#define NET_DVR_STREAMDATA 2 //流数据

pBuffer: 存放数据的缓冲区指针

dwBufSize: 缓冲区的大小

dwUser: 用户数据，就是上面输入的用户数据

返回值：TRUE 表示成功，FALSE 表示失败。

注：此函数包括开始和停止用户处理客户端收到的数据，当 fRealDataCallBack 不为 NULL 时，开始用户处理客户端收到的数据，当设置为 NULL 表示停止用户处理客户端收到的数据。当用户开始接收数据时，第一个包是 40 个字节的文件头，用户可以用这个头来打开播放器，以后回调的就是压缩的码流。IP Camera 设备下调用此接口返回的是经过转化的 HIKVISION 格式的 MPEG4 流数据。

NET_DVR_API BOOL __stdcall NET_DVR_SetStandardDataCallBack (LONG lRealHandle, void(CALLBACK *fStdDataCallBack) (LONG lRealHandle, DWORD dwDataType, BYTE *pBuffer, DWORD dwBufSize, DWORD dwUser), DWORD dwUser)

功能：设置回调函数，用户自己处理客户端收到的数据，该接口目前仅供 IP Camera 设备的 RTSP 方式下使用。

参数说明：

lRealHandle: NET_DVR_RealPlay()的返回值

fStdDataCallBack: 回调函数

dwUser: 用户数据

回调函数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

dwDataType: 数据类型

```
#define NET_DVR_SYSHEAD      1    //系统头数据
#define NET_DVR_STD_VIDEODATA 4    //标准视频流数据
#define NET_DVR_STD_AUDIODATA 5    //标准音频流数据
```

pBuffer: 存放数据的缓冲区指针

dwBufSize: 缓冲区的大小

dwUser: 用户数据, 就是上面输入的用户数据

返回值: TRUE 表示成功, FALSE 表示失败。

注: 此函数包括开始和停止用户处理客户端收到的数据, 当 fStdDataCallBack 不为 NULL 时, 开始用户处理客户端收到的数据, 当设置为 NULL 表示停止用户处理客户端收到的数据. 当用户开始接收数据时, 第一个包是 40 个字节的文件头, 用户可以用这个头来打开播放器, 以后回调的就是压缩的标准码流。

BOOL NET_DVR_SaveRealData(LONG lRealHandle, char *sFileName)

功能: 保存捕获到的数据到指定的文件(*.mp4)中

参数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

sFileName: 文件名, 后缀为 .mp4

返回值: TRUE 表示成功, FALSE 表示失败。

注: IP Camera 设备下调用此接口保存的是经过转化的 HIKVISION 格式的 MPEG4 流数据。

BOOL NET_DVR_StopSaveRealData(LONG lRealHandle)

功能: 停止捕获

参数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.6.6 抓图

BOOL NET_DVR_CapturePicture(LONG lRealHandle, char *sPicFileName)

功能: 抓图并转换成 32 位真彩色 BMP 位图

参数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

sPicFileName: 保存 BMP 图象的文件名, 后缀为 .bmp, 文件的长度小于等于 100 字节。

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.7 云台控制

BOOL NET_DVR_GetPTZCtrl(LONG lRealHandle)

功能: 查看云台控制权, 必须在启动预览之后调用 (此接口保留)

参数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

返回值: TRUE 表示成功获得控制权, 可以控制云台, FALSE 表示失败, 不能控制云台, 调用 NET_DVR_GetLastError() 查看错误类型。

BOOL NET_DVR_GetPTZCtrl_Other(LONG lUserID, LONG lChannel)

功能: 查看云台控制权, 不需要预览图像 (此接口保留)

参数说明:

lUserID: NET_DVR_Login ()的返回值

lChannel: 硬盘录像机的通道号

返回值: TRUE 表示成功获得控制权, 可以控制云台, FALSE 表示失败, 不能控制云台, 调用 NET_DVR_GetLastError() 查看错误类型。

BOOL NET_DVR_PTZControl(LONG lRealHandle, DWORD dwPTZCommand, DWORD dwStop)

功能: 云台控制, 必须在启动预览之后调用

参数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

dwPTZCommand: 云台控制命令

```
#define LIGHT_PWRON 2      /* 接通灯光电源 */
#define WIPER_PWRON 3      /* 接通雨刷开关 */
#define FAN_PWRON 4        /* 接通风扇开关 */
#define HEATER_PWRON 5     /* 接通加热器开关 */
#define AUX_PWRON 6        /* 接通辅助设备开关 */
#define ZOOM_IN 11         /* 焦距以变大(倍率变大) */
#define ZOOM_OUT 12        /* 焦距变小(倍率变小) */
#define FOCUS_IN 13        /* 焦点前调 */
#define FOCUS_OUT 14       /* 焦点后调 */
#define IRIS_ENLARGE 15    /* 光圈扩大 */
#define IRIS_SHRINK 16     /* 光圈缩小 */
#define TILT_UP 21         /* 云台向上 */
#define TILT_DOWN 22       /* 云台向下 */
#define PAN_LEFT 23        /* 云台左转 */
#define PAN_RIGHT 24       /* 云台右转 */
#define PAN_AUTO 29        /* 云台以 SS 的速度左右自动扫描 */
```

dwStop: 让云台停止动作还是开始动作, 0-开始, 1-停止, 云台的每一个动作都要调用该接口两次, 前面两个参数一样, 只是 dwStop 一次为 0, 一次为 1

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_PTZControl_Other(LONG lUserID, LONG lChannel, DWORD dwPTZCommand, DWORD dwStop)

功能: 云台控制, 不需要预览图像

参数说明:

lUserID: NET_DVR_Login ()的返回值

lChannel: 硬盘录像机的通道号

dwPTZCommand: 云台控制命令

dwStop 是让云台停止动作还是开始动作, 0-开始, 1-停止, 云台的每一个动作都要调用该接口两次, 前面两个参数一样, 只是 dwStop 一次为 0, 一次为 1

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_TransPTZ(LONG lRealHandle,char *pPTZCodeBuf,DWORD dwBufSize)

功能: 透明云台控制, 必须在启动预览之后调用

参数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

pPTZCodeBuf: 存放云台控制码缓冲区的指针

dwBufSize: 云台控制码的长度

返回值: TRUE表示成功, FALSE表示失败。

BOOL NET_DVR_TransPTZ_Other(LONG lUserID,LONG lChannel,char *pPTZCodeBuf, DWORD dwBufSize)

功能: 透明云台控制, 不需要预览图像

参数说明:

lUserID: NET_DVR_Login ()的返回值

lChannel: 硬盘录像机的通道号, 指明是往哪个解码器发送数据

pPTZCodeBuf: 存放云台控制码缓冲区的指针

dwBufSize: 云台控制码的长度

返回值: TRUE表示成功, FALSE表示失败。

BOOL NET_DVR_PTZPreset(LONG lRealHandle,DWORD dwPTZPresetCmd,DWORD dwPresetIndex)

功能: 云台预制位操作, 必须在启动预览之后调用

参数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

dwPTZPresetCmd: 云台预制位命令:

define SET_PRESET 8 /* 设置预置点 */

define CLE_PRESET 9 /* 清除预置点 */

define GOTO_PRESET 39 /* 转到预置点 */

dwPresetIndex: 预制点的序号, 最多支持 128 个预制点

返回值: TRUE表示成功, FALSE表示失败。

BOOL NET_DVR_PTZPreset_Other(LONG lUserID,LONG lChannel,DWORD dwPTZPresetCmd,DWORD dwPresetIndex)

功能: 云台预制位操作, 不需要预览图像

参数说明:

lUserID: NET_DVR_Login ()的返回值

lChannel: 硬盘录像机的通道号

dwPTZPresetCmd: 云台预制位命令:

define SET_PRESET 8 /* 设置预置点 */

define CLE_PRESET 9 /* 清除预置点 */

define GOTO_PRESET 39 /* 转到预置点 */

dwPresetIndex: 预制点的序号,最多支持 128 个预制点

返回值: TRUE 表示成功, FALSE 表示失败。

注:

NET_DVR_PTZControl()

NET_DVR_PTZControl_Other()

NET_DVR_PTZPreset()

NET_DVR_PTZPreset_Other()

这四个接口都是由服务器来确定给云台发送什么云台控制码, 而客户端只需要调用相应的宏定义即可, 服务器会根据目前设置的解码器种类和解码器地址向云台发送控制码, 如果目前设置的解码器不是需要的, 则需要在服务器端修改, 如果用的解码器服务器端不支持, 则无法用这几个接口来控制。

NET_DVR_TransPTZ()

NET_DVR_TransPTZ_Other()

这两个接口则是由客户端向云台发送控制码, 不需要设置服务器端的解码器, 服务器只是将收到的云台控制码传送给解码器, 不做任何处理。

BOOL NET_DVR_PTZControl_EX(LONG lRealHandle,DWORD dwPTZCommand,DWORD dwStop)

功能: 云台控制, 必须在启动预览之后调用, 性能比NET_DVR_PTZControl好, 只能控制V1.4以及以上版本的设备。

参数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

dwPTZCommand: 云台控制命令

dwStop: 让云台停止动作还是开始动作, 0-开始, 1-停止, 云台的每一个动作都要调用该接口两次, 前面两个参数一样, 只是 dwStop 一次为 0, 一次为 1

返回值: TRUE表示成功, FALSE表示失败。

BOOL NET_DVR_TransPTZ_EX(LONG lRealHandle,char *pPTZCodeBuf,DWORD dwBufSize)

功能: 透明云台控制, 必须在启动预览之后调用, 性能比NET_DVR_TransPTZ好, 只能控制V1.4以及以上版本的设备。

参数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

pPTZCodeBuf: 存放云台控制码缓冲区的指针

dwBufSize: 云台控制码的长度

返回值: TRUE表示成功, FALSE表示失败。

BOOL NET_DVR_PTZPreset_EX(LONG lRealHandle,DWORD dwPTZPresetCmd,DWORD dwPresetIndex)

功能: 云台预制位操作, 必须在启动预览之后调用, 性能比 NET_DVR_PTZPreset 好, 只能控制 V1.4 以及以上版本的设备。

参数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

dwPTZPresetCmd: 云台预制位命令:

dwPresetIndex: 预制点的序号, 最多支持 128 个预制点

返回值: TRUE表示成功, FALSE表示失败。

BOOL NET_DVR_PTZControlWithSpeed(LONG lRealHandle, DWORD dwPTZCommand, DWORD dwStop, DWORD dwSpeed)

功能: 带速度云台控制, 必须在启动预览之后调用

参数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

dwPTZCommand: 云台控制命令

dwStop: 让云台停止动作还是开始动作, 0-开始, 1-停止, 云台的每一个动作都要调用该接口两次, 前面两个参数一样, 只是 dwStop 一次为 0, 一次为 1

dwSpeed: 云台控制的速度, 用户按不同解码器的速度控制值设置。取值为 1-7
返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_PTZControlWithSpeed_Other(LONG lUserID, LONG lChannel, DWORD dwPTZCommand, DWORD dwStop, DWORD dwSpeed)

功能: 云台控制, 不需要预览图像

参数说明:

lUserID: NET_DVR_Login ()的返回值

lChannel: 硬盘录像机的通道号

dwPTZCommand: 云台控制命令

dwStop 是让云台停止动作还是开始动作,0-开始, 1-停止, 云台的每一个动作都要调用该接口两次, 前面两个参数一样, 只是 dwStop 一次为 0, 一次为 1

dwSpeed: 云台控制的速度, 用户按不同解码器的速度控制值设置。取值为 1-7

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_PTZCruise(LONG lRealHandle, DWORD dwPTZCruiseCmd, BYTE byCruiseRoute, BYTE byCruisePoint, WORD wInput)

功能: 控制云台巡航, 必须在启动预览之后调用

参数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

dwPTZCruiseCmd: 云台控制命令

```
#define FILL_PRE_SEQ    30 // 将预置点加入巡航序列
#define SET_SEQ_DWELL   31 // 设置巡航点停顿时间
#define SET_SEQ_SPEED   32 // 设置巡航速度
#define CLE_PRE_SEQ     33 // 将预置点从巡航序列中删除
#define RUN_SEQ         37 // 开始巡航
#define STOP_SEQ        38 // 停止巡航
```

byCruiseRoute: 巡航路径, 最多支持 32 条路径

byCruisePoint: 巡航点, 最多支持 32 个点

wInput: 不同巡航命令时的值不同, 预置点(最大 128)、时间(最大 255)、速度(最大 15)

返回值:

TRUE 表示成功, FALSE 表示失败

BOOL NET_DVR_PTZCruise_Other(LONG lUserID, LONG lChannel, DWORD dwPTZCruiseCmd, BYTE byCruiseRoute, BYTE byCruisePoint, WORD wInput)

功能: 云台控制, 不需要预览图像

参数说明:

lUserID: NET_DVR_Login ()的返回值

lChannel: 硬盘录像机的通道号

dwPTZCruiseCmd: 云台控制命令

```
#define FILL_PRE_SEQ    30 // 将预置点加入巡航序列
#define SET_SEQ_DWELL   31 // 设置巡航点停顿时间
#define SET_SEQ_SPEED   32 // 设置巡航速度
#define CLE_PRE_SEQ     33 // 将预置点从巡航序列中删除
#define RUN_SEQ         37 // 开始巡航
#define STOP_SEQ        38 // 停止巡航
```

byCruiseRoute: 巡航路径, 最多支持 32 条路径

byCruisePoint: 巡航点, 最多支持 32 个点

wInput: 不同巡航命令时的值不同, 预置点(最大 128)、时间(最大 255)、速度(最大 15)

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_PTZCruise_EX(LONG lRealHandle,DWORD dwPTZCruiseCmd,BYTE byCruiseRoute, BYTE byCruisePoint, WORD wInput)

功能: 云台预置位操作, 必须在启动预览之后调用, 性能比 NET_DVR_PTZCruise 好, 只能控制 V1.4 及以上版本的设备。

参数说明:

lRealHandle: NET_DVR_RealPlay()的返回值

dwPTZCruiseCmd: 云台控制命令

```
#define FILL_PRE_SEQ    30 // 将预置点加入巡航序列
#define SET_SEQ_DWELL   31 // 设置巡航点停顿时间
#define SET_SEQ_SPEED   32 // 设置巡航速度
#define CLE_PRE_SEQ     33 // 将预置点从巡航序列中删除
#define RUN_SEQ         37 // 开始巡航
#define STOP_SEQ        38 // 停止巡航
```

byCruiseRoute: 巡航路径,最多支持 32 条路径

byCruisePoint: 巡航点, 最多支持 32 个点

wInput: 不同巡航命令时的值不同, 预置点(最大 128)、时间(最大 255)、速度(最大 15)

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_PTZTrack(LONG lRealHandle,DWORD dwPTZTrackCmd)

功能:云台轨迹操作(启动图像预览)

参数:

lRealHandle: NET_DVR_RealPlay()的返回值

dwPTZTrackCmd: 云台轨迹命令:

```
#define STA_MEM_CRUISE  34 // 开始记录轨迹
#define STO_MEM_CRUISE  35 // 停止记录轨迹
#define RUN_CRUISE      36 // 开始轨迹
```

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_PTZTrack_Other(LONG lUserID,LONG lChannel,DWORD dwPTZTrackCmd)

功能:云台轨迹操作, 不需要预览图像

参数:

lUserID: NET_DVR_Login ()的返回值

lChannel: 硬盘录像机的通道号

dwPTZTrackCmd: 云台轨迹命令:

```
#define STA_MEM_CRUISE  34 // 开始记录轨迹
#define STO_MEM_CRUISE  35 // 停止记录轨迹
#define RUN_CRUISE      36 // 开始轨迹
```

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_PTZTrack_EX(LONG lRealHandle,DWORD dwPTZTrackCmd)

功能:云台轨迹操作(启动图象预览), 必须在启动预览之后调用, 性能比 NET_DVR_PTZTrack 好, 只能控制 V1.4 及以上版本的设备。

参数:

lRealHandle: NET_DVR_RealPlay()的返回值

dwPTZTrackCmd: 云台轨迹命令:

```
#define STA_MEM_CRUISE    34 // 开始记录轨迹
#define STO_MEM_CRUISE    35 // 停止记录轨迹
#define RUN_CRUISE        36 // 开始轨迹
```

返回值:

TRUE 表示成功,FALSE 表示失败

4.1.8 回放与下载

4.1.8.1 查找文件

LONG NET_DVR_FindFile(LONG lUserID,LONG lChannel,DWORD dwFileType, LPNET_DVR_TIME lpStartTime, LPNET_DVR_TIME lpStopTime)

功能: 查找服务器文件

参数说明:

lUserID: NET_DVR_Login ()的返回值

lChannel: 通道号

dwFileType: 要查找的文件类型, 0xff — 全部, 0 — 定时录像, 1 — 移动侦测, 2 — 报警触发, 3 — 报警|动测, 4 — 报警&动测, 5 — 命令触发, 6 — 手动录像

lpStartTime: 文件的开始时间

lpStopTime: 文件的结束时间

返回值: -1 表示失败, 其他值作为 NET_DVR_FindClose 等函数的参数

注:当该接口返回成功后, 就可以调用 NET_DVR_FindNextFile 接口来获取文件信息

LONG NET_DVR_FindFileByCard(LONG lUserID,LONG lChannel,DWORD dwFileType, BOOL bNeedCardNum, BYTE *sCardNumber, LPNET_DVR_TIME lpStartTime, LPNET_DVR_TIME lpStopTime);

功能: 根据卡号、时间查找文件。

参数说明:

lUserID: NET_DVR_Login ()的返回值

lChannel: 通道号

dwFileType: 要查找的文件类型, 0xFF — 全部, 0 — 定时录像, 1 — 移动侦测, 2 — 接近报警, 3 — 出钞报警, 4 — 进钞报警, 5 — 命令触发, 6 — 手动录像, 7 — 震动报警

bNeedCardNum: 是否需要卡号信息 TRUE: 需要; FALSE: 不需要

sCardNumber: 卡号信息

lpStartTime: 文件的开始时间

lpStopTime: 文件的结束时间

返回值: -1 表示失败, 其他值作为 NET_DVR_FindClose 等函数的参数

注: 当该接口返回成功后, 就可以调用 NET_DVR_FindNextFile 接口来获取文件信息

LONG NET_DVR_FindNextFile(LONG IFindHandle, LPNET_DVR_FIND_DATA lpFindData)

功能: 获取文件信息

参数说明:

IFindHandle : 文件查找句柄, 由 NET_DVR_FindFile 返回

lpFindData: 保存文件信息的指针

返回值: -1 表示失败, NET_DVR_SUCCESS 表示获取文件信息成功; NET_DVR_ISFINDING 表示正在查找请耐心等待; NET_DVR_NOMOREFILE 表示没有更多的文件, 查找结束; NET_DVR_FILE_NOFIND 表示没有找到文件; 其他值为异常。

//函数返回值

#define NET_DVR_FILE_SUCCESS	1000	//获得文件信息
#define NET_DVR_FILE_NOFIND	1001	//没有文件
#define NET_DVR_ISFINDING	1002	//正在查找文件
#define NET_DVR_NOMOREFILE	1003	//查找文件时没有更多的文件
#define NET_DVR_FILE_EXCEPTION	1004	//查找文件时异常

BOOL NET_DVR_FindClose(LONG IFindHandle)

功能: 关闭 NET_DVR_FindFile 创建的句柄, 释放资源

参数说明:

IFindHandle : 文件查找句柄, 由 NET_DVR_FindFile 返回

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.8.2 回放

LONG NET_DVR_PlayBackByName(LONG IUserID, char *sPlayBackFileName, HWND hWnd)

功能: 按文件名回放

参数说明:

IUserID: NET_DVR_Login ()的返回值

sPlayBackFileName: 要回放的文件名

hWnd: 回放文件的窗口句柄

返回值: -1 表示失败, 其他值作为 NET_DVR_StopPlayBack 等函数的参数

LONG NET_DVR_PlayBackByTime(LONG IUserID, LONG IChannel, LPNET_DVR_TIME lpStartTime, LPNET_DVR_TIME lpStopTime, HWND hWnd)

功能: 按时间回放

参数说明:

IUserID: NET_DVR_Login ()的返回值

IChannel: 通道号

lpStartTime: 文件的开始时间

lpStopTime: 文件的结束时间

hWnd: 回放的窗口句柄

返回值: -1 表示失败, 其他值作为 NET_DVR_StopPlayBack 等函数的参数

BOOL NET_DVR_StopPlayBack(LONG IPlayHandle)

功能: 停止回放

参数说明:

IPlayHandle: 播放句柄, NET_DVR_PlayBackByName()或者 NET_DVR_PlayBackByTime()的返回值

返回值: TRUE 表示成功, FALSE 表示失败。

int NET_DVR_GetPlayBackPlayerIndex(LONG IPlayHandle)

功能: 获取当前用来解码和显示的播放器句柄, 可以通过该句柄来调用播放器SDK接口实现特定的功能

参数说明:

IPlayHandle: NET_DVR_PlayBackByName 或者 NET_DVR_PlayBackByTime 的返回值

BOOL PlayM4_GetBMP(LONG nPort,PBYTE pBitmap,DWORD nBufSize,DWORD* pBmpSize);
BOOL PlayM4_GetJPEG(LONG nPort,PBYTE pJpeg,DWORD nBufSize,DWORD* pJpegSize);

作如下调用:

PlayM4_GetBMP(NET_DVR_GetPlayBackPlayerIndex (), pBitmap, nBufSize, pBmpSize)

PlayM4_GetJPEG(NET_DVR_GetPlayBackPlayerIndex (), pJpeg, nBufSize, pJpegSize)

即可实现抓图 BMP 和 JPEG 到内存中。

4.1.8.3 播放控制

BOOL NET_DVR_PlayBackControl(LONG IPlayHandle,DWORD dwControlCode,DWORD dwInValue,DWORD *lpOutValue)

功能: 控制回放时的状态

参数说明:

IPlayHandle: 播放句柄, NET_DVR_PlayBackByName()或者 NET_DVR_PlayBackByTime()的返回值

dwControlCode: 控制命令

#define NET_DVR_PLAYSTART	1//开始播放
#define NET_DVR_PLAYSTOP	2//停止播放
#define NET_DVR_PLAYPAUSE	3//暂停播放
#define NET_DVR_PLAYRESTART	4//恢复播放
#define NET_DVR_PLAYFAST	5//快放
#define NET_DVR_PLAYSLOW	6//慢放
#define NET_DVR_PLAYNORMAL	7//正常速度
#define NET_DVR_PLAYFRAME	8//单帧放
#define NET_DVR_PLAYSTARTAUDIO	9//打开声音
#define NET_DVR_PLAYSTOPAUDIO	10//关闭声音
#define NET_DVR_PLAYAUDIOVOLUME	11//调节音量

```

#define NET_DVR_PLAYSETPOS      12//改变文件回放的进度
#define NET_DVR_PLAYGETPOS      13//获取文件回放的进度
#define NET_DVR_PLAYGETTIME     14//获取当前已经播放的时间
#define NET_DVR_PLAYGETFRAME    15//获取当前已经播放的帧数
#define NET_DVR_GETTOTALFRAMES  16//获取当前播放文件总的帧数
#define NET_DVR_GETTOTALTIME    17//获取当前播放文件总的时间
#define NET_DVR_THROWBFRAME     20//丢 B 帧
dwInValue: 输入参数, 比如设置文件回放的进度时, 此参数表示进度值
lpOutValue: 输出参数, 比如获取当前播放文件总的时间, 此参数就是得到的总时间
返回值: TRUE 表示成功, FALSE 表示失败。

```

BOOL NET_DVR_RefreshPlay(LONG lPlayHandle)

功能: 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, 其它情况无效。

参数说明:

lPlayHandle: 播放句柄, NET_DVR_PlayBackByName()或者 NET_DVR_PlayBackByTime()的返回值

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.8.4 捕获

BOOL NET_DVR_PlayBackSaveData(LONG lPlayHandle, char *sFileName)

功能: 保存回放的数据

参数说明:

lPlayHandle: 播放句柄, NET_DVR_PlayBackByName()或者 NET_DVR_PlayBackByTime()的返回值

sFileName: 保存的文件名

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_StopPlayBackSave(LONG lPlayHandle)

功能: 停止保存

参数说明:

lPlayHandle: 播放句柄, NET_DVR_PlayBackByName()或者 NET_DVR_PlayBackByTime()的返回值

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_SetPlayDataCallBack(LONG lPlayHandle, void(CALLBACK *fPlayDataCallBack) (LONG lPlayHandle, DWORD dwDataType, BYTE *pBuffer, DWORD dwBufSize, DWORD dwUser), DWORD dwUser)

功能: 设置回调函数, 用户自己处理码流数据

参数说明:

lPlayHandle: 播放句柄, NET_DVR_PlayBackByName()或者 NET_DVR_PlayBackByTime()的返回值

fPlayDataCallBack: 回调函数

dwUser: 用户数据

回调函数说明:

lPlayHandle: 播放句柄, NET_DVR_PlayBackByName()或者 NET_DVR_PlayBackByTime()的返回值

dwDataType: 数据类型

#define NET_DVR_SYSHEAD 1 //系统头数据

#define NET_DVR_STREAMDATA 2 //流数据

pBuffer: 存放数据的缓冲区指针

dwBufSize: 缓冲区的大小

dwUser: 用户数据, 就是上面输入的用户数据

返回值: TRUE 表示成功, FALSE 表示失败。

注: 此函数包括开始和停止用户处理码流数据, 当 fPlayDataCallBack 不为 NULL 时, 开始用户处理收到的数据, 当设置为 NULL 表示停止用户处理收到的数据. 当用户开始接收数据时, 第一个包是 40 个字节的文件头, 用户可以用这个头来打开播放器, 以后回调的就是音视频数据。

4.1.8.5 获取 OSD 时间

BOOL NET_DVR_GetPlayBackOsdTime(LONG lPlayHandle, LPNET_DVR_TIME lpOsdTime)

功能: 获取回放显示的 OSD 时间

参数说明:

lPlayHandle: 播放句柄, NET_DVR_PlayBackByName()或者 NET_DVR_PlayBackByTime()的返回值

lpOsdTime: 获得的时间, 一个指向 NET_DVR_TIME 结构的指针

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.8.6 抓图

BOOL NET_DVR_PlayBackCaptureFile(LONG lPlayHandle, char *sFileName)

功能: 回放时抓图

参数说明:

lPlayHandle: 播放句柄, NET_DVR_PlayBackByName()或者 NET_DVR_PlayBackByTime()的返回值

sFileName: 保存图片的文件名, 文件名的长度小于等于 100 字节

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.8.7 下载

LONG NET_DVR_GetFileByName(LONG lUserID, char *sDVRFileName, char *sSavedFileName)

功能: 按文件名下载

参数说明:

lUserID: NET_DVR_Login ()的返回值

sDVRFileName: 要下载的文件名

sSavedFileName: 下载后保存到 PC 机的文件名

返回值: -1 表示失败, 其他值表示成功, 作为 NET_DVR_StopGetFile 等函数的参数

LONG NET_DVR_GetFileByTime(LONG lUserID, LONG lChannel, LPNET_DVR_TIME lpStartTime, LPNET_DVR_TIME lpStopTime, char *sSavedFileName)

功能: 按时间下载

参数说明:

lUserID: NET_DVR_Login () 的返回值

lChannel: 通道号

lpStartTime: 开始时间

lpStopTime: 结束时间

sSavedFileName: 下载后保存到PC机的文件名

返回值: -1 表示失败, 其他值表示成功, 作为 NET_DVR_StopGetFile 等函数的参数

BOOL NET_DVR_StopGetFile(LONG lFileHandle)

功能: 停止下载

参数说明:

lFileHandle: 下载句柄, NET_DVR_GetFileByName () 或者 NET_DVR_GetFileByTime () 的返回值

返回值: TRUE 表示成功, FALSE 表示失败。

int NET_DVR_GetDownloadPos(LONG lFileHandle)

功能: 获取下载的进度

参数说明:

lFileHandle: 下载句柄, NET_DVR_GetFileByName () 或者 NET_DVR_GetFileByTime () 的返回值

返回值: -1 表示失败, 0-100: 下载的进度, 100 表示下载结束, >100: 由于网络原因或 DVR 忙, 下载异常终止

4.1.9 报警

LONG NET_DVR_SetupAlarmChan(LONG lUserID)

功能: 建立报警上传通道

参数说明:

lUserID: NET_DVR_Login () 的返回值

返回值: -1 表示失败, 其他值作为 NET_DVR_CloseAlarmChan () 等函数的参数

BOOL NET_DVR_CloseAlarmChan(LONG lAlarmHandle)

功能: 断开报警上传通道

参数说明:

lAlarmHandle: NET_DVR_SetupAlarmChan 的返回值

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_SetAlarmOut(LONG lUserID, LONG lAlarmOutPort, LONG lAlarmOutStatic)

功能: 设置报警输出

参数说明:

lUserID: NET_DVR_Login () 的返回值

lAlarmOutPort: 报警输出端口, 从 0 开始, 0xff 表示全部
 lAlarmOutStatic: 报警输出状态, 0—停止输出, 1—输出
 返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_GetAlarmOut(LONG lUserID, LPNET_DVR_ALARMOUTSTATUS lpAlarmOutState)

功能: 获取报警输出

参数说明:

lUserID: NET_DVR_Login () 的返回值

lpAlarmOutState: 指向 NET_DVR_ALARMOUTSTATUS 结构的指针

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.10 语音对讲和转发

4.1.10.1 语音对讲

LONG NET_DVR_StartVoiceCom(LONG lUserID, void(CALLBACK *fVoiceDataCallBack)(LONG lVoiceComHandle, char *pRecvDataBuffer, DWORD dwBufSize, BYTE byAudioFlag, DWORD dwUser), DWORD dwUser)

功能: 开始语音对讲

参数说明:

lUserID: NET_DVR_Login () 的返回值

fVoiceDataCallBack: 回调函数, 回调音频数据

dwUser: 用户数据

回调函数说明:

lVoiceComHandle: NET_DVR_StartVoiceCom () 的返回值

pRecvDataBuffer: 存放数据的缓冲区指针

dwBufSize: 缓冲区的大小

byAudioFlag: 数据类型

0- 客户端采集的音频数据

1- 客户端收到设备端的音频数据

dwUser: 用户数据, 就是上面输入的用户数据

返回值: -1 表示失败, 其他值作为 NET_DVR_SetVoiceComClientVolume () 等函数的参数

BOOL NET_DVR_SetVoiceComClientVolume(LONG lVoiceComHandle, WORD wVolume)

功能: 设置语音对讲 PC 端的音量

参数说明:

lVoiceComHandle: NET_DVR_StartVoiceCom 的返回值

wVolume: 设置后的音量, 从 0-0xffff

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.10.2 语音转发

LONG NET_DVR_StartVoiceCom_MR(LONG IUserID, void(CALLBACK *fVoiceDataCallBack)(LONG IVoiceComHandle,char *pRecvDataBuffer,DWORD dwBufSize,BYTE byAudioFlag,DWORD dwUser), DWORD dwUser);

功能:开始语音转发

参数:

IUserID: NET_DVR_Login()的返回值

fVoiceDataCallBack: 回调函数, 将 DVR 语音数据回调, 得到的数据是编码以后的音频数据, 需调用我们提供的音频解码函数 (详见音频编解码) 后方可得到 PCM 数据。

dwUser: 用户数据

回调函数说明:

IVoiceComHandle: NET_DVR_StartVoiceCom_MR ()的返回值

pRecvDataBuffer: 存放数据的缓冲区指针

dwBufSize: 缓冲区的大小,为 80 的整数倍

byAudioFlag: 数据类型

1- 客户端收到设备端的音频数据

dwUser: 用户数据, 就是上面输入的用户数据

返回值:

-1 表示失败,其他值作为 NET_DVR_VoiceComSendData NET_DVR_StopVoiceCom 等函数的参数

BOOL NET_DVR_VoiceComSendData(LONG IVoiceComHandle,char *pSendBuf,DWORD dwBufSize);

功能:发送语音数据, 发送编码以后的音频数据 (详见音频编解码)

参数:

IVoiceComHandle: NET_DVR_StartVoiceCom_MR()的返回值

pSendBuf:存放语音数据的缓冲区

dwBufSize: 语音数据大小, 大小是固定 80 字节。

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_StopVoiceCom(LONG IVoiceComHandle)

功能: 停止语音对讲或者语音转发

参数说明:

IVoiceComHandle: NET_DVR_StartVoiceCom 的返回值

返回值: TRUE 表示成功, FALSE 表示失败。

注: 语音转发的音频数据采样格式和音频播放格式必须符合以下标准: 采样频率为 16000, 16 位采样, 单通道。

采样标准:

const int SAMPLES_PER_SECOND = 16000;

const int CHANNEL = 1;

const int BITS_PER_SAMPLE = 16;

WAVEFORMATEX m_wavFormatEx;

m_wavFormatEx.cbSize = sizeof(m_wavFormatEx);

m_wavFormatEx.nBlockAlign = CHANNEL * BITS_PER_SAMPLE / 8;

```
m_wavFormatEx.nChannels = CHANNEL;
m_wavFormatEx.nSamplesPerSec = SAMPLES_PER_SECOND;
m_wavFormatEx.wBitsPerSample = BITS_PER_SAMPLE;
m_wavFormatEx.nAvgBytesPerSec=SAMPLES_PER_SECOND*m_wavFormatEx.nBlockAlign
```

4.1.11 语音广播

BOOL NET_DVR_ClientAudioStart()

功能：启动 PC 端声音捕获

返回值：TRUE 表示成功，FALSE 表示失败。

BOOL NET_DVR_ClientAudioStop()

功能：停止 PC 端声音捕获

返回值：TRUE 表示成功，FALSE 表示失败。

BOOL NET_DVR_AddDVR(LONG IUserID)

功能：添加一台 DVR 到可以接收 PC 机声音的组里

参数说明：

IUserID: NET_DVR_Login () 的返回值

返回值：TRUE 表示成功，FALSE 表示失败。

BOOL NET_DVR_DelDVR(LONG IUserID)

功能：删除已经加到组的 DVR

参数说明：

IUserID: NET_DVR_Login () 的返回值

返回值：TRUE 表示成功，FALSE 表示失败。

4.1.12 音频编解码

4.1.12.1 音频解码

void* NET_DVR_InitG722Decoder(int nBitrate);

功能:初始化音频解码

参数说明:

nBitrate: 码率 nBitrate = 16000;

返回值:

音频解码句柄

BOOL NET_DVR_DecodeG722Frame(void *pDecHandle, unsigned char* pInBuffer, unsigned char* pOutBuffer);

功能:音频解码,

参数说明:

pDecHandle: 音频解码句柄

pInBuffer: 输入缓冲区, 编码后数据 size = 80
 pOutBuffer: 输出缓冲区, 解码后数据 size = 1280
 返回值:
 TRUE 表示成功,FALSE 表示失败

void NET_DVR_ReleaseG722Decoder(void *pDecHandle);

功能:释放音频解码资源

参数说明:

pDecHandle: 音频解码句柄

4.1.12.2 音频编码

void* NET_DVR_InitG722Encoder();

功能:初始化音频编码

参数说明:

返回值:

音频编码句柄

BOOL NET_DVR_EncodeG722Frame(void *pEncodeHandle,unsigned char* pInBuffer, unsigned char* pOutBuffer);

功能:音频编码

参数说明:

pEncodeHandle: 音频编码句柄

pInBuffer: 输入缓冲区, 按**采样标准**(采样频率为 16000, 16 位采样, 单通道)得到的原始音频数据 size = 1280

pOutBuffer: 输出缓冲区, 编码后的数据 size = 80

返回值:

TRUE 表示成功,FALSE 表示失败

void NET_DVR_ReleaseG722Encoder(void *pEncodeHandle);

功能:释放音频编码资源

参数说明:

pDecHandle: 音频编码句柄

4.1.13 透明通道

LONG NET_DVR_SerialStart(LONG IUserID, LONG ISerialPort, void(CALLBACK *fSerialDataCallBack)(LONG ISerialHandle, char *pRecvDataBuffer, DWORD dwBufSize, DWORD dwUser), DWORD dwUser)

功能: 建立透明通道

参数说明:

IUserID: NET_DVR_Login () 的返回值

ISerialPort: 串口号, 1-232 串口, 2-485 串口

dwUser: 用户数据

fSerialDataCallBack: 回调函数

回调函数说明:

lSerialHandle: NET_DVR_SerialStart()的返回值

pRecvDataBuffer: 存放接收到数据的缓冲区指针

dwBufSize: 缓冲区的大小

dwUser: 上面的用户数据

返回值: -1表示失败, 其他值作为NET_DVR_SerialSend()等函数的参数

BOOL NET_DVR_SerialSend(LONG lSerialHandle,LONG lChannel,char *pSendBuf,DWORD dwBufSize)

功能: 通过透明通道向 DVR 串口发送数据

参数说明:

lSerialHandle: NET_DVR_SerialStart 的返回值

lChannel: 硬盘录像机的通道号, 以 485 建立透明通道时有效,指明往哪个通道送数据,以 232 建立透明通道时设置成 0;

pSendBuf: 要发送的缓冲区的指针

dwBufSize: 缓冲区的大小

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_SerialStop(LONG lSerialHandle)

功能: 断开透明通道

参数说明:

lSerialHandle: NET_DVR_SerialStart 的返回值

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.14 远程控制本地显示

BOOL NET_DVR_ClickKey(LONG lUserID,LONG lKeyIndex)

功能: 远程控制面板上的键

参数说明:

lUserID: NET_DVR_Login () 的返回值

lKeyIndex:

1—按钮 1,2-按钮 2.....9-按钮 9, 10—按钮 0, 11-POWER,12—MENU,13—ENTER,14—"ESC",15—"上"或者"云台上开始",16-"下"或者"云台下开始",17—"左"或者"云台左开始",18—"右"或者"云台右开始",19-"EDIT"或者"光圈+开始",22-"PLAY",23-"REC",24-"PAN"或者"光圈-开始",25-"多画面"或者"聚焦-开始",26—"输入法"或者"聚焦+开始",27-"对讲",28-"系统信息",29-"快进",30-"快退",32-"云台上结束",33-"云台下结束",34-"云台左结束",35-"云台右结束",36-"光圈+结束",37-"光圈-结束",38-"聚焦+结束",39-"聚焦-结束",40-"变倍+开始",41-"变倍+结束",42-"变倍-开始",43-"变倍-结束"

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.15 远程手动录像

BOOL NET_DVR_StartDVRRecord(LONG lUserID,LONG lChannel ,LONG lRecordType)

功能: 客户端启动硬盘录像机录像

参数说明:

IUserID: NET_DVR_Login ()的返回值

IChannel: 通道号

IRecordType: 录像类型(此参数暂时无效)

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_StopDVRRecord(LONG IUserID,LONG IChannel)

功能: 客户端停止硬盘录像机录像

参数说明:

IUserID: NET_DVR_Login ()的返回值

IChannel: 通道号

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.16 解码卡

BOOL NET_DVR_InitDevice_Card(long *pDeviceTotalChan)

功能: 初始化解码卡

参数说明:

pDeviceTotalChan: 保存解码卡通道个数的指针

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_ReleaseDevice_Card()

功能: 释放解码卡资源

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_InitDDraw_Card(HWND hParent,COLORREF colorKey)

功能: 初始化解码卡 OVERLAY 表面

参数说明:

hParent: 父窗口句柄

colorKey: 用户设置的透明色

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_ReleaseDDraw_Card()

功能: 释放解码卡使用的 OVERLAY 表面

返回值: TRUE 表示成功, FALSE 表示失败。

LONG NET_DVR_RealPlay_Card(LONG IUserID,LPNET_DVR_CARDINFO lpCardInfo,long IChannelNum)

功能: 用解码卡预览图象

参数说明:

IUserID: NET_DVR_Login ()的返回值

lpCardInfo: 指向 NET_DVR_CARDINFO 结构的指针

IChannelNum: 使用解码卡的哪个通道来解码,从 0 开始

返回值: -1 表示失败, 其他值作为 NET_DVR_ResetPara_Card ()等函数的参数

BOOL NET_DVR_ResetPara_Card(LONG IRealHandle,LPDISPLAY_PARA lpDisplayPara)

功能: 重置解码卡显示参数

参数说明:

lRealHandle: 播放句柄, NET_DVR_RealPlay_Card()的返回值

lpDisplayPara: 指向 DISPLAY_PARA 结构的指针

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_RefreshSurface_Card()

功能: 刷新解码卡使用的 OVERLAY 表面

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_ClearSurface_Card()

功能: 清除解码卡使用的 OVERLAY 表面

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_RestoreSurface_Card()

功能: 恢复解码卡使用的 OVERLAY 表面

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_OpenSound_Card(LONG lRealHandle)

功能: 输出某一路的声音

参数说明:

lRealHandle: 播放句柄, NET_DVR_RealPlay_Card()的返回值

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_CloseSound_Card(LONG lRealHandle)

功能: 关闭某一路的声音

参数说明:

lRealHandle: 播放句柄, NET_DVR_RealPlay_Card()的返回值

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_SetVolume_Card(LONG lRealHandle,WORD wVolume)

功能: 调节某一路的音量

参数说明:

lRealHandle: 播放句柄, NET_DVR_RealPlay_Card()的返回值

wVolume: 设置的音量

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_AudioPreview_Card(LONG lRealHandle,BOOL bEnable)

功能: 在 PC 机上预览声音

参数说明:

lRealHandle: 播放句柄, NET_DVR_RealPlay_Card()的返回值

bEnable: TRUE 表示打开声音预览, FALSE 表示关闭声音

返回值: TRUE 表示成功, FALSE 表示失败。

HANDLE NET_DVR_GetChanHandle_Card(LONG lRealHandle);

功能: 获取解码卡的通道句柄

参数说明:

lRealHandle: 播放句柄, NET_DVR_RealPlay_Card()的返回值

返回值: 0 失败, 其他为解码卡的句柄。

BOOL NET_DVR_GetSerialNum_Card(long lChannelNum,DWORD *pDeviceSerialNo)

功能：获取解码卡序列号

参数说明：

lChannelNum：解码卡通道号

pDeviceSerialNo：保存解码卡序列号的指针

返回值：TRUE表示成功，FALSE表示失败。

BOOL NET_DVR_SetDspErrMsg_Card(UINT nMessage,HWND hWnd)

功能：设置解码卡DSP死掉后向主机发送的消息

参数说明：

nMessage：DSP死掉向主机发送的消息，wParam返回解码卡的通道号

hWnd：发送消息的窗口

版权所有杭州海康威视数字技术有限公司 第46 页 杭州海康威视数字技术有限公司技术文档

返回值：TRUE表示成功，FALSE表示失败。

注：我们可以通过 **#define WM_MYCOMMAND WM_USER+1** 定义一个用户自定义的消息，这个消息对应的消息处理函数为 **void OnMyCommand(WPARAM wParam, LPARAM lParam)**，wParam 返回解码卡的通道号，它们之间的映射关系由 **ON_MESSAGE(WM_MYCOMMAND,OnMyCommand)**来实现，此时，我们可以这样调用该函数：**NET_DVR_SetDspErrMsg_Card (WM_MYCOMMAND,hWnd)**。

BOOL NET_DVR_ResetDSP_Card(long lChannelNum)

功能：重置DSP,当用户收到DSP死掉的消息后,可以调用这个接口来重置DSP

参数说明：

lChannelNum：解码卡通道号

返回值：TRUE表示成功，FALSE表示失败。

BOOL NET_DVR_CapturePicture_Card(LONG lRealHandle, char *sPicFileName)

功能:硬解码抓图

参数说明:

lRealHandle: NET_DVR_RealPlay_Card()的返回值

sPicFileName: 文件名

返回值：TRUE 表示成功,FALSE 表示失败

4.1.17 日志

LONG NET_DVR_FindDVRLog(LONG lUserID,LONG lSelectMode, DWORD dwMajorType,DWORD dwMinorType, LPNET_DVR_TIME lpStartTime, LPNET_DVR_TIME lpStopTime)

功能：获取硬盘录像机日志

参数说明：

lUserID：NET_DVR_Login ()的返回值

lSelectMode：查询方式:0—全部，1—按类型，2—按时间，3—按时间和类型

dwMajorType：主类型

dwMinorType：次类型

lpStartTime：开始的时间

lpStopTime：结束的时间

返回值：-1 表示失败，其他值作为 NET_DVR_FindNextLog ()等函数的参数

LONG NET_DVR_FindNextLog(LONG lLogHandle,LPNET_DVR_LOG lpLogData)

功能：获取日志信息

参数说明：

lLogHandle: NET_DVR_FindDVRLog的返回值

lpLogData: 返回的日志信息

返回值：如下定义：

#define NET_DVR_FILE_SUCCESS	1000	//获得文件信息
#define NET_DVR_FILE_NOFIND	1001	//没有文件
#define NET_DVR_ISFINDING	1002	//正在查找文件
#define NET_DVR_NOMOREFILE	1003	//查找文件时没有更多的文件
#define NET_DVR_FILE_EXCEPTION	1004	//查找文件时异常

BOOL NET_DVR_FindLogClose(LONG lLogHandle)

功能：释放查找日志的资源

参数说明：

lLogHandle: NET_DVR_FindDVRLog的返回值

返回值：TRUE 表示成功，FALSE 表示失败。

4.1.18 获取设备状态

BOOL NET_DVR_GetDVRWorkState(LONG lUserID,LPNET_DVR_WORKSTATE lpWorkState)

功能：获取硬盘录像机工作状态

参数说明：

lUserID: NET_DVR_Login ()的返回值

lpWorkState: 存放获得工作状态信息

返回值：TRUE 表示成功，FALSE 表示失败。

4.1.19 JPEG 抓图

typedef struct

```
{
    WORD  wPicSize;          /* 0=CIF, 1=QCIF, 2=D1, 3=UXGA */
    WORD  wPicQuality;       /* 图片质量系数 0-最好 1-较好 2-一般 */
}NET_DVR_JPEGPARA, *LPNET_DVR_JPEGPARA;
```

BOOL NET_DVR_CaptureJPEGPicture(LONG lUserID, LONG lChannel, LPNET_DVR_JPEGPARA lpJpegPara, char *sPicFileName);

功能:抓 JPEG 图,保存成文件;

参数:

lUserID: NET_DVR_Login()的返回值

lChannel: 通道号

lpJpegPara:指向 NET_DVR_JPEGPARA 的指针

sPicFileName: 保存 JPEG 数据的文件

返回值:

TRUE 表示成功,FALSE 表示失败

NET_DVR_API BOOL __stdcall NET_DVR_CaptureJPEGPicture_NEW(LONG IUserID, LONG IChannel, LPNET_DVR_JPEGPARA lpJpegPara, char *sJpegPicBuffer, DWORD dwPicSize, LPDWORD lpSizeReturned)

功能:抓 JPEG 图, 保存在内存中;

参数:

IUserID: NET_DVR_Login()的返回值

IChannel: 通道号

lpJpegPara:指向NET_DVR_JPEGPARA的指针

sJpegPicBuffer: 保存JPEG数据的缓冲区

dwPicSize: 输入缓冲区大小

lpSizeReturned:返回图片的大小

返回值:

TRUE表示成功,FALSE表示失败

4.1.20 多显卡支持

BOOL NET_DVR_InitDDrawDevice();

功能: 枚举系统中的显示设备

参数说明:

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_ReleaseDDrawDevice();

功能: 释放枚举显示设备的过程中分配的资源

参数说明:

返回值: TRUE 表示成功, FALSE 表示失败。

LONG NET_DVR_GetDDrawDeviceTotalNums();

功能: 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);

参数说明:

返回值: 如果返回 0 表示失败; 如果返回 0, 则表示系统中只有主显示设备。如果返回 1, 则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。返回其他值, 则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。

BOOL NET_DVR_SetDDrawDevice(LONG IPlayPort, DWORD nDeviceNum);

功能: 设置播放窗口使用的显卡。**注意: 该窗口必须在该显卡所对应的监视器上才能显示播放画面。**

参数说明:

nDeviceNum 显示设备的设备号, 如果是 0, 则表示使用主显示设备。

返回值: TRUE 表示成功, FALSE 表示失败。

4.1.21 多路解码器

◆ 启动/停止动态解码

NET_DVR_API BOOL __stdcall NET_DVR_MatrixStartDynamic(LONG IUserID, DWORD dwDecChanNum, LPNET_MATRIX_DYNAMIC_DEC lpDynamicInfo);

功能:启动动态连接测试

参数说明:

IUserID: NET_DVR_Login()的返回值

dwDecChanNum:解码通道

lpDynamicInfo:指向 NET_MATRIX_DYNAMIC_DEC 结构的指针

返回值:

TRUE 表示成功,FALSE 表示失败

NET_DVR_API BOOL __stdcall NET_DVR_MatrixStopDynamic(LONG IUserID, DWORD dwDecChanNum);

功能:停止动态连接测试

参数说明:

IUserID: NET_DVR_Login()的返回值

dwDecChanNum:解码通道

返回值:

TRUE 表示成功,FALSE 表示失败

◆ 设置获取轮循解码通道参数

NET_DVR_API BOOL __stdcall NET_DVR_MatrixSetLoopDecChanInfo(LONG IUserID, DWORD dwDecChanNum, LPNET_MATRIX_LOOP_DECINFO lpInter);

功能:设置轮循解码通道

参数说明:

IUserID: NET_DVR_Login()的返回值

dwDecChanNum: 解码通道

lpInter:指向 NET_MATRIX_LOOP_DECINFO 结构的指针

返回值:

TRUE 表示成功,FALSE 表示失败

NET_DVR_API BOOL __stdcall NET_DVR_MatrixGetLoopDecChanInfo(LONG IUserID, DWORD dwDecChanNum, LPNET_MATRIX_LOOP_DECINFO lpInter);

功能:获取轮循解码通道

参数说明:

IUserID: NET_DVR_Login()的返回值

dwDecChanNum: 解码通道

lpInter:指向 NET_MATRIX_LOOP_DECINFO 结构的指针

返回值:

TRUE 表示成功,FALSE 表示失败

◆ 解码轮循开关

NET_DVR_API BOOL __stdcall NET_DVR_MatrixSetLoopDecChanEnable(LONG IUserID,

DWORD dwDecChanNum, DWORD dwEnable);

功能:设置解码通道轮循开关

参数说明:

lUserID: NET_DVR_Login()的返回值

dwDecChanNum: 解码通道

dwEnable:0 表示关闭 1 表示打开

返回值:

TRUE 表示成功,FALSE 表示失败

NET_DVR_API BOOL __stdcall NET_DVR_MatrixGetLoopDecChanEnable(LONG lUserID, DWORD dwDecChanNum, DWORD dwEnable);

功能:获取解码通道轮循开关

参数说明:

lUserID: NET_DVR_Login()的返回值

dwDecChanNum: 解码通道

dwEnable:0 表示关闭 1 表示打开

返回值:

TRUE 表示成功,FALSE 表示失败

NET_DVR_API BOOL __stdcall NET_DVR_MatrixGetLoopDecEnable(LONG lUserID, DWORD wEnable);

功能:获取所有解码通道轮循开关

参数说明:

lUserID: NET_DVR_Login()的返回值

dwEnable: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

◆ 获取解码通道状态

NET_DVR_API BOOL __stdcall NET_DVR_MatrixGetDecChanInfo(LONG lUserID, DWORD dwDecChanNum, LPNET_DVR_MATRIX_DEC_CHAN_INFO lpInter);

功能:获取当前解码通道信息

参数说明:

lUserID: NET_DVR_Login()的返回值

dwDecChanNum: 解码通道

lpInter:指向 NET_DVR_MATRIX_DEC_CHAN_INFO 结构的指针

返回值:

TRUE 表示成功,FALSE 表示失败

◆ 获取/设置解码输出通道的视频制式

NET_DVR_API BOOL __stdcall NET_DVR_MatrixGetVideoStandard(LONG lUserID, DWORD dwDecChanNum, LPDWORD lpdwVideoStandard);

功能:获取解码输出通道的视频制式

参数说明:

lUserID: NET_DVR_Login()的返回值

dwDecChanNum: 解码通道

dwVideoStandard:视频制式 0:PAL 1:NTSC

返回值:

TRUE 表示成功,FALSE 表示失败

NET_DVR_API BOOL __stdcall NET_DVR_MatrixSetVideoStandard(LONG IUserID, DWORD dwDecChanNum, DWORD dwVideoStandard);

功能:设置解码输出通道的视频制式

参数说明:

IUserID: NET_DVR_Login()的返回值

dwDecChanNum: 解码通道

*dwVideoStandard:视频制式 0:PAL 1:NTSC

返回值:

TRUE 表示成功,FALSE 表示失败

◆ 设置获取解码通道开关

NET_DVR_API BOOL __stdcall NET_DVR_MatrixSetDecChanEnable(LONG IUserID, DWORD dwDecChanNum, DWORD dwEnable);

功能:设置解码通道开关

参数说明:

IUserID: NET_DVR_Login()的返回值

dwDecChanNum: 解码通道

dwEnable:0 表示关闭 1 表示打开

返回值:

TRUE 表示成功,FALSE 表示失败

NET_DVR_API BOOL __stdcall NET_DVR_MatrixGetDecChanEnable(LONG IUserID, DWORD dwDecChanNum, DWORD dwEnable);

功能:获取解码通道开关

参数说明:

IUserID: NET_DVR_Login()的返回值

dwDecChanNum: 解码通道

dwEnable:0 表示关闭 1 表示打开

返回值:

TRUE 表示成功,FALSE 表示失败

4.1.22 获取 SDK 的 Build 号

NET_DVR_API DWORD __stdcall NET_DVR_GetSDKBuildVersion();

功能: 获取 SDK 的 build 号

返回值: SDK 的 build 号, 2 个高字节表示版本, 2 个低字节表示版本修改增值, 返回形如 0x0202****

4.2 设备端配置函数

4.2.1 参数配置

BOOL NET_DVR_GetDVRConfig(LONG IUserID, DWORD dwCommand,LONG IChannel,

LPVOID lpOutBuffer,DWORD dwOutBufferSize,LPDWORD lpBytesReturned)

功能：获取硬盘录像机的参数

参数说明：

lUserID: NET_DVR_Login ()的返回值

dwCommand: 参数类型

lChannel: 通道号，如果不是通道参数，lChannel 不用,置为-1 即可

lpOutBuffer: 存放输出参数的缓冲区

dwOutBufferSize: 缓冲区的大小

lpBytesReturned: 实际返回的缓冲区大小

返回值: TRUE 表示成功，FALSE 表示失败。

BOOL NET_DVR_SetDVRConfig(LONG lUserID, DWORD dwCommand,LONG lChannel, LPVOID lpInBuffer,DWORD dwInBufferSize)

设置硬盘录像机的参数

参数说明：

lUserID: NET_DVR_Login ()的返回值

dwCommand: 参数类型

lChannel: 通道号，如果不是通道参数，lChannel 不用,置为 0 即可

lpInBuffer: 存放输入参数的缓冲区

dwInBufferSize: 缓冲区的大小

返回值: TRUE 表示成功，FALSE 表示失败。

dwCommand 的类型定义如下：

//NET_DVR_DEVICECFG 结构

#define NET_DVR_GET_DEVICECFG 100 //获取设备参数

#define NET_DVR_SET_DEVICECFG 101 //设置设备参数

// NET_DVR_NETCFG 结构

#define NET_DVR_GET_NETCFG 102 //获取网络参数

#define NET_DVR_SET_NETCFG 103 //设置网络参数

// NET_DVR_NETCFG_OTHER 结构

#define NET_DVR_GET_NETCFG_OTHER 244 //获取网络参数(多路解码器)

#define NET_DVR_SET_NETCFG_OTHER 245 //设置网络参数(多路解码器)

//NET_DVR_PICCFG 结构

#define NET_DVR_GET_PICCFG 104 //获取图象参数

#define NET_DVR_SET_PICCFG 105 //设置图象参数

//NET_DVR_PICCFG_EX结构

#define NET_DVR_GET_PICCFG_EX 200 //获取图象参数（扩展）

#define NET_DVR_SET_PICCFG_EX 201 //设置图象参数（扩展）

// NET_DVR_COMPRESSIONCFG 结构

#define NET_DVR_GET_COMPRESSCFG 106 //获取压缩参数

#define NET_DVR_SET_COMPRESSCFG 107 //设置压缩参数

// NET_DVR_COMPRESSIONCFG_EX 结构

#define NET_DVR_GET_COMPRESSCFG_EX 204 //获取压缩参数(扩展)

#define NET_DVR_SET_COMPRESSCFG_EX 205 //设置压缩参数(扩展)

//

#define NET_DVR_GET_RECORDCFG 108 //获取录像时间参数

#define NET_DVR_SET_RECORDCFG 109 //设置录像时间参数

```

#define NET_DVR_GET_DECODERCFG 110 //获取解码器参数
#define NET_DVR_SET_DECODERCFG 111 //设置解码器参数
#define NET_DVR_GET_RS232CFG 112 //获取 232 串口参数
#define NET_DVR_SET_RS232CFG 113 //设置 232 串口参数
#define NET_DVR_GET_ALARMCFG 114 //获取报警输入参数
#define NET_DVR_SET_ALARMCFG 115 //设置报警输入参数
#define NET_DVR_GET_ALARMOUTCFG 116 //获取报警输出参数
#define NET_DVR_SET_ALARMOUTCFG 117 //设置报警输出参数
#define NET_DVR_GET_TIMECFG 118 //获取 DVR 时间
#define NET_DVR_SET_TIMECFG 119 //设置 DVR 时间
#define NET_DVR_GET_PREVIEWCFG 120 //获取预览参数
#define NET_DVR_SET_PREVIEWCFG 121 //设置预览参数
#define NET_DVR_GET_VIDEOOUTCFG 122 //获取视频输出参数
#define NET_DVR_SET_VIDEOOUTCFG 123 //设置视频输出参数
#define NET_DVR_GET_USERCFG 124 //获取用户参数
#define NET_DVR_SET_USERCFG 125 //设置用户参数
#define NET_DVR_GET_EXCEPTIONCFG 126 //获取异常参数
#define NET_DVR_SET_EXCEPTIONCFG 127 //设置异常参数
#define NET_DVR_GET_SHOWSTRING 130 //获取叠加字符参数
#define NET_DVR_SET_SHOWSTRING 131 //设置叠加字符参数
// NET_DVR_COMPRESSIONCFG 结构
#define NET_DVR_GET_EVENTCOMPCFG 132 //获取事件触发录像参数
#define NET_DVR_SET_EVENTCOMPCFG 133 //设置事件触发录像参数
#define NET_DVR_GET_AUXOUTCFG 140 //获取报警触发辅助输出设置
#define NET_DVR_SET_AUXOUTCFG 141 //设置报警触发辅助输出设置
#define NET_DVR_GET_PREVIEWCFG_AUX 142 //获取-s系列双输出预览参数
#define NET_DVR_SET_PREVIEWCFG_AUX 143 //设置-s系列双输出预览参数
#define NET_DVR_GET_USERCFG_EX 202 //获取用户参数
#define NET_DVR_SET_USERCFG_EX 203 //设置用户参数
#define NET_DVR_GET_NETAPPCFG 222 //获取网络应用参数
NTP/DDNS/EMAIL
#define NET_DVR_SET_NETAPPCFG 223 //设置网络应用参数
NTP/DDNS/EMAIL
#define NET_DVR_GET_NTPCFG 224 //获取网络应用参数 NTP
#define NET_DVR_SET_NTPCFG 225 //设置网络应用参数 NTP
#define NET_DVR_GET_DDNSCFG 226 //获取网络应用参数 DDNS
#define NET_DVR_SET_DDNSCFG 227 //设置网络应用参数 DDNS
#define NET_DVR_GET_EMAILCFG 228 //获取网络应用参数 EMAIL
#define NET_DVR_SET_EMAILCFG 229 //设置网络应用参数 EMAIL
#define NET_DVR_GET_NFSCFG 230 //获取NFS 配置
#define NET_DVR_SET_NFSCFG 231 //设置NFS 配置
//叠加字符
#define NET_DVR_GET_SHOWSTRING_EX 238 //获取叠加字符参数扩展
#define NET_DVR_SET_SHOWSTRING_EX 239 //设置叠加字符参数扩展

//NET_DVR_EMAILCFG结构
#define NET_DVR_GET_EMAILPARACFG 250 //获取EMAIL配置

```

```
#define NET_DVR_SET_EMAILPARACFG    251    //设置EMAIL配置

//扩展DDNS
#define NET_DVR_GET_DDNSCFG_EX      274    //获取扩展DDNS参数
#define NET_DVR_SET_DDNSCFG_EX      275    //设置扩展DDNS参数
```

注：以获取设备的参数和设置设备的参数为例，代码如下：

//获取设备的参数

```
NET_DVR_DEVICECFG devicecfg;
```

```
DWORD BytesReturned;
```

```
NET_DVR_GetDVRConfig(IUserID,    NET_DVR_GET_DEVICECFG,    0,    &devicecfg,
sizeof(NET_DVR_DEVICECFG), &BytesReturned);
```

//设置设备的参数

```
NET_DVR_DEVICECFG devicecfg;
```

```
NET_DVR_SetDVRConfig(IUserID,    NET_DVR_SET_DEVICECFG,    0,    &devicecfg,
sizeof(NET_DVR_DEVICECFG));
```

```
BOOL NET_DVR_SetScaleCFG(LONG IUserID, DWORD dwScale);
```

功能：设置缩放

参数说明：

IUserID: NET_DVR_Login ()的返回值

dwScale: 是否缩放 0—否,1—是;

返回值: TRUE 表示成功, FALSE 表示失败。

```
BOOL NET_DVR_GetScaleCFG(LONG IUserID, DWORD *lpOutScale);
```

功能：获取是否设置缩放

参数说明：

IUserID: NET_DVR_Login ()的返回值

dwScale: 是否缩放 0—否,1—是;

返回值: TRUE 表示成功, FALSE 表示失败。

```
BOOL NET_DVR_SetATMPortCFG(LONG IUserID, WORD wATMPort);
```

功能：设置 ATM 端口

参数说明：

IUserID: NET_DVR_Login ()的返回值

wATMPort: ATM 端口值

返回值: TRUE 表示成功, FALSE 表示失败。

```
BOOL NET_DVR_GetATMPortCFG(LONG IUserID, WORD *lpOutATMPort);
```

功能：获取 ATM 端口

参数说明：

IUserID: NET_DVR_Login ()的返回值

wATMPort: ATM 端口值

返回值: TRUE 表示成功, FALSE 表示失败。

```
BOOL NET_DVR_GetConfigFile(LONG IUserID, char *sFileName);
```

功能：获取所有的配置文件

参数说明:

lUserID: NET_DVR_Login ()的返回值

sFileName: 存放保存配置文件的文件名 (二进制文件)

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_SetConfigFile(LONG lUserID, char *sFileName);

功能: 设置所有的配置文件

参数说明:

lUserID: NET_DVR_Login ()的返回值

sFileName: 存放保存配置文件的文件名 (二进制文件)

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_GetConfigFile_EX(LONG lUserID, char *sOutBuffer, DWORD dwOutSize);

功能: 获取所有的配置文件

参数说明:

lUserID: NET_DVR_Login ()的返回值

sOutBuffer: 存放配置参数的缓冲区

dwOutSize: 缓冲区大小

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_SetConfigFile_EX(LONG lUserID, char *sInBuffer, DWORD dwInSize);

功能: 设置所有的配置文件

参数说明:

lUserID: NET_DVR_Login ()的返回值

sInBuffer: 存放配置参数的缓冲区

dwInSize: 缓冲区大小

返回值: TRUE 表示成功, FALSE 表示失败。

4.2.2 恢复默认值

BOOL NET_DVR_RestoreConfig(LONG lUserID)

功能: 恢复 DVR 默认参数

参数说明:

lUserID: NET_DVR_Login ()的返回值

返回值: TRUE 表示成功, FALSE 表示失败。

4.2.3 保存参数

BOOL NET_DVR_SaveConfig(LONG lUserID)

功能: 保存参数到 FLASH 中

参数说明:

lUserID: NET_DVR_Login ()的返回值

返回值: TRUE 表示成功, FALSE 表示失败。

4.2.4 重启/关闭设备

BOOL NET_DVR_RebootDVR(LONG lUserID)

功能：重启硬盘录像机

参数说明：

lUserID: NET_DVR_Login () 的返回值

返回值：TRUE 表示成功，FALSE 表示失败。

BOOL NET_DVR_ShutDownDVR(LONG lUserID)

功能：关闭硬盘录像机

参数说明：

lUserID: NET_DVR_Login () 的返回值

返回值：TRUE 表示成功，FALSE 表示失败。

4.2.5 远程升级

LONG NET_DVR_Upgrade(LONG lUserID,char *sFileName)

功能：远程升级

参数说明：

lUserID: NET_DVR_Login () 的返回值

sFileName: 升级文件的文件名，文件名长度小于等于 100 个字节

返回值：-1 表示失败，其他值作为 NET_DVR_GetUpgradeState 等函数的参数

BOOL NET_DVR_CloseUpgradeHandle(LONG lUpgradeHandle)

功能：关闭 NET_DVR_Upgrade 接口所创建的句柄，释放资源

参数说明：

lUpgradeState: NET_DVR_Upgrade() 的返回值

返回值：TRUE 表示成功，FALSE 表示失败。

int NET_DVR_GetUpgradeState(LONG lUpgradeHandle)

功能：获取升级的状态

参数说明：

lUpgradeState: NET_DVR_Upgrade() 的返回值

返回值：-1 表示失败，其他值如下定义：

- 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

4.2.6 远程格式化硬盘

LONG NET_DVR_FormatDisk(LONG lUserID,LONG lDiskNumber)

功能：远程格式化硬盘

参数说明:

lUserID: NET_DVR_Login ()的返回值

lDiskNumber: 硬盘号, 从 0 开始, 0xff 表示所有硬盘

返回值: -1 表示失败, 其他值作为 NET_DVR_CloseFormatHandle 等函数的参数

BOOL NET_DVR_CloseFormatHandle(LONG lFormatHandle)

功能: 关闭 NET_DVR_FormatDisk 接口所创建的句柄, 释放资源

参数说明:

lFormatHandle: NET_DVR_FormatDisk ()的返回值

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_GetFormatProgress(LONG lFormatHandle, LONG *pCurrentFormatDisk, LONG *pCurrentDiskPos, LONG *pFormatFinish)

功能: 获取格式化的进度

参数说明:

lFormatHandle: 调用 NET_DVR_FormatDisk 的返回值

pCurrentFormatDisk: 指向保存当前正在格式化的硬盘号的指针, 硬盘号从 0 开始, -1 为初始状态

pCurrentDiskPos: 指向保存当前正在格式化的硬盘的进度的指针, 进度是 0 到 100

pFormatStatic: 指向保存硬盘格式化状态的指针, 1 表示硬盘全部格式化完成, 0 表示正在格式化

2 表示格式化当前硬盘出错, 不能继续格式化此硬盘, 本地和网络硬盘都会出现此错误

3 表示由于网络异常造成网络硬盘丢失而不能开始格式化当前硬盘

返回值: TRUE 表示成功, FALSE 表示失败。

4.2.7 配置交易信息

BOOL NET_DVR_ClientGetframeformat(LONG lUserID, LPNET_DVR_FRAMEFORMAT lpFrameFormat);

功能: 获取 ATM 硬盘录像机的帧格式

参数说明:

lUserID: NET_DVR_Login ()的返回值

lpFrameFormat: 指向结构 NET_DVR_FRAMEFORMAT 的指针

返回值: TRUE 表示成功, FALSE 表示失败。

BOOL NET_DVR_ClientSetframeformat(LONG lUserID, LPNET_DVR_FRAMEFORMAT lpFrameFormat);

功能: 设置 ATM 硬盘录像机的帧格式

参数说明:

lUserID: NET_DVR_Login ()的返回值

lpFrameFormat: 指向结构 NET_DVR_FRAMEFORMAT 的指针

返回值: TRUE 表示成功, FALSE 表示失败。

4.2.8 6001D/F 配置函数

BOOL NET_DVR_SetDecInfo(LONG lUserID, LONG lChannel, LPNET_DVR_DECCEFG lpDecoderinfo);

功能:配置解码设备参数

参数说明:

lUserID: NET_DVR_Login()的返回值

lChannel: 通道号

lpDecoderinfo:指向 NET_DVR_DECCFG 结构的指针

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_GetDecInfo(LONG lUserID, LONG lChannel, LPNET_DVR_DECCFG lpDecoderinfo);

功能:获取解码设备参数

参数说明:

lUserID: NET_DVR_Login()的返回值

lChannel: 通道号

lpDecoderinfo:指向 NET_DVR_DECCFG 结构的指针

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_SetDecTransPort(LONG lUserID, LPNET_DVR_PORTCFG lpTransPort);

功能:设置解码设备透明通道参数

参数说明:

lUserID: NET_DVR_Login()的返回值

lpTransPort:指向 NET_DVR_PORTCFG 结构的指针

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_GetDecTransPort(LONG lUserID, LPNET_DVR_PORTCFG lpTransPort);

功能:获取解码设备透明通道参数

参数说明:

lUserID: NET_DVR_Login()的返回值

lpTransPort:指向 NET_DVR_PORTCFG 结构的指针

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_DecPlayBackCtrl(LONG lUserID, LONG lChannel, DWORD dwControlCode, DWORD dwInValue,DWORD *lpOutValue, LPNET_DVR_PLAYREMOTEFILE lpRemoteFileInfo);

功能:解码设备远程回放下载文件控制

参数说明:

lUserID: NET_DVR_Login()的返回值

lChannel: 通道号

dwControlCode: 控制码

#define NET_DVR_PLAYSTART	1//开始播放
#define NET_DVR_PLAYSTOP	2//停止播放
#define NET_DVR_PLAYPAUSE	3//暂停播放
#define NET_DVR_PLAYRESTART	4//恢复播放
#define NET_DVR_PLAYFAST	5//快放
#define NET_DVR_PLAYSLOW	6//慢放

dwInValue: 设置的参数

lpOutValue: 返回的值

lpRemoteFileInfo:指向 NET_DVR_PLAYREMOTEFILE 结构的指针

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_StartDecSpecialCon(LONG IUserID, LONG IChannel, LPNET_DVR_DECCHANINFO lpDecChanInfo);

功能:启动解码设备动态控制解码连接

参数说明:

IUserID: NET_DVR_Login()的返回值

IChannel: 通道号

lpDecChanInfo:指向 NET_DVR_DECCHANINFO 结构的指针

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_StopDecSpecialCon(LONG IUserID, LONG IChannel, LPNET_DVR_DECCHANINFO lpDecChanInfo);

功能:停止解码设备动态控制解码连接

参数说明:

IUserID: NET_DVR_Login()的返回值

IChannel: 通道号 0xffffffff 表示所有的解码通道

lpDecChanInfo:指向 NET_DVR_DECCHANINFO 结构的指针

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_DecCtrlDec(LONG IUserID, LONG IChannel, DWORD dwControlCode);

功能:解码设备运行状态控制

参数说明:

IUserID: NET_DVR_Login()的返回值

IChannel: 通道号 0xffffffff 表示所有的通道

dwControlCode:控制状态 1—启动解码 2—停止解码 3—停止当前轮循 4—继续轮循

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_DecCtrlScreen(LONG IUserID, LONG IChannel, DWORD dwControl);

功能:解码设备解码窗口放大还原

参数说明:

IUserID: NET_DVR_Login()的返回值

IChannel: 通道号

dwControl:全屏还是还原 1 表示全屏 2 表示还原

返回值:

TRUE 表示成功,FALSE 表示失败

BOOL NET_DVR_GetDecCurLinkStatus(LONG IUserID, LONG IChannel, LPNET_DVR_DECSTATUS lpDecStatus);

功能:获取解码设备当前连接状态

参数说明:

IUserID: NET_DVR_Login()的返回值

lChannel: 通道号

lpDecStatus:指向 NET_DVR_DECSTATUS 结构的指针

返回值:

TRUE 表示成功,FALSE 表示失败

4.2.9 IP 快球配置函数

NET_DVR_API BOOL __stdcall NET_DVR_PTZSelZoomIn(LONG lRealHandle, LPNET_DVR_POINT_FRAME pStruPointFrame);

功能:云台图象区域选择放大或缩小

参数:

lRealHandle: NET_DVR_RealPlay()的返回值

pStruPointFrame: 指向 NET_DVR_POINT_FRAME 的指针

返回值:

TRUE 表示成功,FALSE 表示失败

NET_DVR_API BOOL __stdcall NET_DVR_GetPTZCruise(LONG lUserID,LONG lChannel,LONG lCruiseRoute, LPNET_DVR_CRUISE_RET lpCruiseRet)

功能:获取云台巡航路径 (IP快球)

参数:

lUserID: NET_DVR_Login()的返回值

lChannel: DVR的通道号

lCruiseRoute: 巡航路径,最多支持32条路径

lpCruiseRet: 返回的NET_DVR_CRUISE_RET结构

返回值:

TRUE 表示成功,FALSE 表示失败