目 录

版本 1.0.0.5 发行日期：2004.11

关于此手册

以下为本手册的内容介绍：

本开发包说明书包括以下几个方面:

第一章 简 介

vc404p卡是专门针对系统开发商进行单路和多路视频开发的PCI视频卡。它具有低CPU占用率、多路实时显示、支持报警输入输出的特点。针对系统开发商，提供完整的二次开发包，通过该SDK，系统开发商可以使用VB，VC等编程软件进行系统设计，选择存储成为AVI或使用软件MPEG-4压缩引擎进行压缩，提供对图象的对比度色度亮度灰度进行调整，可以捕获图象通道中的动态图象存储成为JPG静态图象，并且可以通过卡上的I/O报警接口连接各种报警器。

他采用超强 Philips 7134 芯片。Philips 7134 芯片是一颗 9bit ADC，相对于市场上采用 8bit ADC BT878 芯片的卡来说不管是图像质量还是颜色的饱和度方面都要强很多。它独具的4线3D梳状滤波器能自动消除噪点使它的图像监视质量能比BT878提高35%左右。Philips 7134 芯片本身就集成了音视频采集的功能，但是BT878芯片需要另加一颗音频采集芯片才能采集音视频。总的来说VC404P是天敏率先推出的市场上第一款采用 Philips 7134 芯片的监控卡，它超强的功能和优质的画质，我相信它一定会成为大家DVR视频监控卡的首选产品。

1.1 硬件系统介绍

PCI 卡为单卡 4 路 SAA7134，可以同时传输 4 路实时音视频信号。同时，在机器性能允许的情况下可实现单机多卡运行，亦即可达到16路同屏预览和捕获。

1.2 软件系统介绍

本套软件实现硬件功能有：实时多路 OVERLAY 显示，实时多路音视频数据采集，并且可以进行实时多路 MPEG 带音频或不带音频数据压缩。在这版本中所涉及到的MPEG压缩都指的是MPEG4压缩。该软件包括驱动文件，SDK文件，APP文件三部分。

1.3 产品特性

◆PNP 支持，支持 Windows 2000/XP

◆支持一机多卡，一卡四路，支持 PAL/NTSC，每路视频带一路音频，各通道同时工作互不干扰。

◆支持 Overlay 多路同时预览，CPU 占用率极低。

◆Software Video codec:

支持 MPEG 4 sample profile codec

压缩位率：64K-2Mbps

帧率 1-30 帧 / 秒可选

支持 CIF Video MPEG 4 Encorder

◆Software Audio codec

◆提供MPEG4压缩引擎，可对多路视频图像进行压缩。

◆支持 A/V 复合，长时间同步。

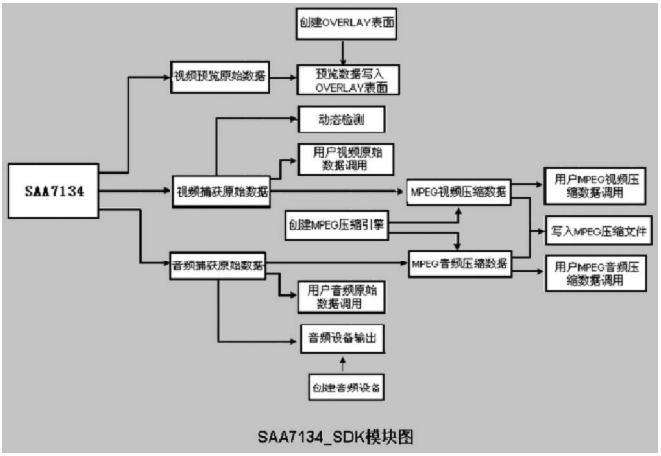
◆支持压缩流 / 预览流叠加 year/month/day/hour/min/sec,text 的功能

◆提供动态 AVI 图像捕获。

◆可将动态图像捕获为JPG静态图象存盘。

◆视频卡中内置4路报警输入4路报警输出，可接驳多种类型报警器。

◆提供功能全面的二次开发包，可应用于保安监控，医疗，交通，银行等方面的系统的开发。



SAA7134_SDK模块图

1.4 系统需求

CPU P4 1.5G 或以上、内存 256MB、硬盘 40G 以上、支持 OVERLAY 表面显卡

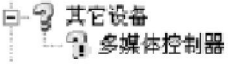
Windows 2000 、Windows XP

第二章　VC404P_401P 安装指南

3.1 驱动程序安装(以 windows xp 为例做说明)

驱动程序的安装以 windows xp 为例做说明,并不是表明该驱动程序不可以在别的操作系统中使用，VC404P 驱动程序支持 windows 2000。

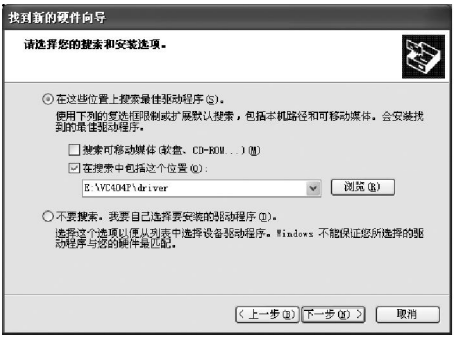
安装前注意：可能在某些操作系统中安装时没有提示找到新硬件。



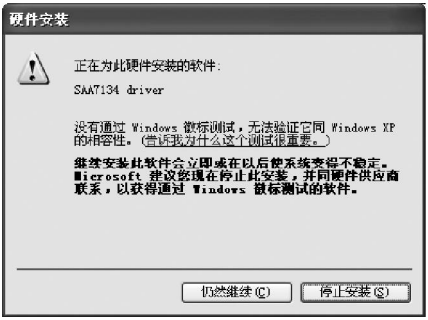
你可以在设备管理器中找到如图中的设备，请您删除它们，再重新启动系统。

由于 VC404P 卡具有即插即用的功能，所以在硬件安装完毕后，启动 wiodows 时会自动检测到 VC404P 卡，此时系统将提示“ 发现新硬件 ”并弹出：“ 找到新硬件向导 ”。

1)　请选择“ 从列表或指定位置安装（高级）”，然后单击“ 下一步 ”。



2)　插入随卡附送的天敏 VC404P 驱动光碟，选择“ 在这些位置上搜索最佳驱动程序 ”，单击“ 浏览 ”，选择 SDK300 驱动所在目录，单击“ 下一步 ”。



3） windows xp 提示您，它无法了解此驱动程序没有通过Microsoft 测试，请单击“仍然继续”。

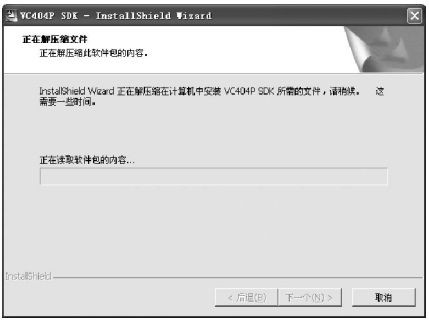
4）找到新硬件向导正在拷贝文件。

6）找到新硬件向导正在拷贝文件。

7） 请确认您的系统设备管理器是否包含SAA7134 设备，并且没有任何感叹号（！）或是“X”号。

3.2 应用软件安装(以windows xp为例做说明)

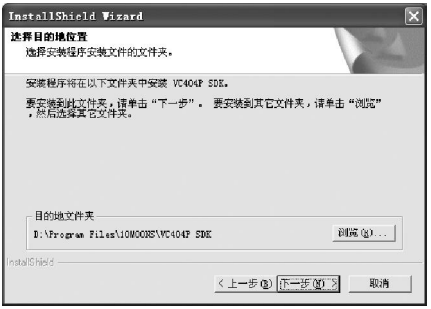
应用软件的安装以windows xp为例做说明,并不是表明该驱动程序不可以在别的操作系统中使用,VC404P应用软件支持windows 2000。



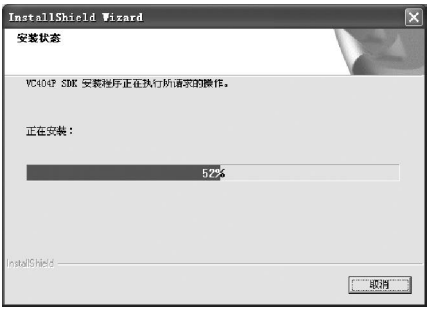
1） 将天敏产品附送的光碟放入CDO-ROM驱动器中并运行：x:\VC404P\app\VC404P.exe进入VC404P的安装界面。

2） 选择安装语言，选好后单击“确定”。

3） 单击“下一步”。



4）选择目标文件夹，可以使用默认的目标文件夹：C:\Program Files\10moons\VC404P 也可以点击旁边的“浏览”按钮选择目标文件夹，选好后点击“下一步”。



6）Windows 开始安装 VC404P 的应用软件。

6）单击“完成”按钮，Windows 已经完成了 VC404P 的应用软件安装。

第三章 10MOONS VC404P_401P SDK 函数说明

Sa7134Capture.DLL 使用说明 (for VC++ 6.0)

操作系统：Windows 2000/winxp

一、编译环境

1、 将 Sa7134Capture.lib 、mediaTransmit.lib 文件加入工程设置的“Link – General – Object/ library module”中。

2、将 Sa7134Capture.h 、mediaTransmit.h 文件加入工程。

3、 将 Sa7134Capture.dll、mediaTransmit.dll 文件拷入系统目录或其他系统能找到的目录中。

4、开始编写代码。

该 SDK 文件即 Sa7124Capture.DLL,输出功能在头文件 Sa7134Capture.H 文件中定义

二、函数说明：

函数修改：

```
//初始化SDK，增加是否初始化视频设备和初始化视频设备参数
BOOL WINAPI VCAInitSdk( HWND hWndMain, BOOL bInitVidDev = TRUE, BOOL bInitAudDev = TRUE );
```

新增函数：

```
//初始化视频
BOOL WINAPI VCAInitVidDev();
//初始化音频
BOOL WINAPI VCAInitAudDev();
```

3.1 数据常量介绍

```
#define          ERR_SUCCESS                          0
```

说明：成功

```
#define          ERR_NODEVICEFOUND                    1
```

说明：系统找不到VC404（SAA7134）设备

```
#define          ERR_UNSUPPORTFUNC          2
```

说明：不支持的函数

```
#define          ERR_ALLOCRESOURCE          3
```

说明：分配资源错误

```
#define          ERR_INITDIRECTDRAW         4
```

error

说明：初始化 Directdraw 错误

```
#define          ERR_INITDIRECTSOUND        5
```

说明：初始化 Directsound 错误

3.2 数据结构介绍:

```
typedef enum
{
       RGB32   = 0x0,
       RGB24   = 0x1,
       RGB16   = 0x2,
       RGB15   = 0x3,
       YUY2    = 0x4,
       BTYUV   = 0x5,
       Y8      = 0x6,
       RGB8    = 0x7,
       PL422   = 0x8,
       PL411   = 0x9,
       YUV12   = 0xA,
       YUV9    = 0xB,
       RAW     = 0xE
}
COLORFORMAT;
```

●说明：视频预览和视频捕捉数据流格式，系统目前这版本只支持YUY2格式.

```
typedef enum {
    BRIGHTNESS = 0,
    CONTRAST   = 1,
    SATURATION = 2,
    HUE        = 3,
```

```
    SHARPNESS  = 4,
}
COLORCONTROL;
```

● 说明:视频预览及视频捕获的显示属性,其中:

BRIGHTNESS 为亮度,value 范围:0 ~ 255, 最佳:80

CONTRAST 为对比度,value 范围: -128 ~ 127 最佳:0x44

SATURATION 为饱和度,value 范围: -128 ~ 127 最佳:0x40

HUE 为色度,value 范围: -128 ~ 127 最佳:0x0

只有当 COLORDEVICETYPE 等于 COLOR_DECODER 时 HUE 才有效。

SHARPNESS 为锐度, value 范围:-8 ~ 7 最佳:0x0

只有当 COLORDEVICETYPE 等于 COLOR_DECODER 时 SHARPNESS 才有效。

```
    typedef enum{
    COLOR_DECODER  = 0,
    COLOR_PREVIEW  = 1,
    COLOR_CAPTURE  = 2
}
COLORDEVICETYPE;
```

● 说明:显示设备的显示属性,其中:

COLOR_DECODER为解码器的显示属性,它会影响视频预览和视频捕获的显示属性

COLOR_PREVIEW 为视频预览的显示属性

COLOR_CAPTURE 为视频捕获的显示属性

```
typedef enum
{
       CAP_NULL_STREAM  = 0,
       CAP_ORIGIN_STREAM  = 1,
       CAP_MPEG4_STREAM  = 2
}
CAPMODEL;
```

● 说明:音视频捕获方式,其中:

CAP_NULL_STREAM 捕获无效

CAP_ORIGIN_STREAM 捕获为原始流回调

CAP_MPEG4_STREAM 捕获为 MPEG4

```
typedef enum
{
       MPEG4_AVIFILE_ONLY       = 0,
```

```
        MPEG4_CALLBACK_ONLY      = 1,
        MPEG4_AVIFILE_CALLBACK   =2
}
MP4MODEL;
```

● 说明：音视频 MPEG4 捕获方式,只有 CAPMODEL 等于 CAP_MPEG4_STREAM 时有效，其中:

MPEG4_AVIFILE_ONLY　　存为 MPEG4 文件

MPEG4_CALLBACK_ONLY　MPEG 数据回调

MPEG4_AVIFILE_CALLBACK 存为 MPEG 文件并回调

```
typedef enum
{
FIELD_FREQ_50HZ  = 0,
    FIELD_FREQ_60HZ  = 1,
    FIELD_FREQ_0HZ   = 2
}
eFieldFrequency;
```

● 说明：视频源的输入频率，其中:

FIELD_FREQ_50HZ　50HZ,绝大多数为 PAL 制式

FIELD_FREQ_60HZ　60HZ，绝大多数为 NTSC 制式

FIELD_FREQ_0HZ　 无信号

```
typedef enum {
    HIGH_VOLTAGE = 0,
    LOW_VOLTAGE = 1
}
eVOLTAGELEVEL;
```

● 说明：电平状态，其中:

HIGH_VOLTAGE　高电平

LOW_VOLTAGE　 低电平

2.3　SDK 导出函数说明：

● typedef void (CALLBACK *PrcCbMotionDetect) (DWORD dwCard, BOOL bMove, LPVOID lpContext)

说明：动态检测回调.

参数：dwCard：返回的卡号.

bMove：TRUE:检测到物体移动开始，FALSE:　检测到物体移动结束。

LpContext：监测上下文

返回值：无

● typedef void (CALLBACK *PrcVidCapCallBack)(DWORD dwCard, BYTE *pbuff, DWORD dwSize)

说明： 视频捕获原始数据回调.

参数： dwCard：返回的卡号.

Pbuff： 视频原始数据的指针

dwSize:视频原始数据缓冲区大小，单位字节

返回值：无

● typedef void (CALLBACK *PrcVidMpegCallBack)(DWORD dwCard, BYTE *pbuff, DWORD dwSize, BOOL isKeyFrm)

说明：视频 MPEG 压缩数据回调.

参数：dwCard：返回的卡号.

Pbuff：视频 MPEG 数据的指针

dwSize: 视频 MPEG 数据缓冲区大小，单位字节

isKeyFrm: 是否使用关键帧

返回值：无

● typedef BOOL (CALLBACK *PrcVidSaveErrCallBack)(DWORD dwCard)

说明：视频捕获存盘出错回调.

参数：dwCard：返回的卡号.

返回值：TRUE:成功

FALSE:失败.

● typedef void (CALLBACK *PrcAudCapCallBack)(DWORD dwCard, BYTE *pbuff,DWORD dwSize)

说明：音频捕获原始数据回调.

参数：dwCard：返回的卡号.

Pbuff： 音频原始数据的指针

dwSize: 音频原始数据缓冲区大小，单位字节

返回值：无

● typedef void (CALLBACK *PrcAudMpegCallBack)(DWORD dwCard, BYTE *pbuff,DWORD dwSize)

说明：音频 MPEG 压缩数据回调.

参数：dwCard： 返回的卡号.

Pbuff：音频 MPEG 数据的指针

dwSize: 音频 MPEG 数据缓冲区大小，单位字节

返回值：无

● typedef BOOL (CALLBACK *PrcAudSaveErrCallBack)(DWORD dwCard)

说明：音频捕获存盘出错回调.

参数：dwCard：返回的卡号

返回值：TRUE: 成功

FALSE: 失败.

● typedef void (CALLBACK *PrcIoAlertCallBack)(DWORD dwCard)

说明：IO 报警回调.

参数： dwCard：返回的卡号.

返回值：无

● BOOL WINAPI VCARegVidCapCallBack(DWORD dwCard, PrcVidCapCallBack ppCall)

说明：注册视频捕获原始数据回调.

参数：dwCard： 卡号.

ppCall： 需要注册的回调函数指针

返回值：TRUE: 成功

FALSE: 失败.

● BOOL WINAPI VCARegVidMpegCallBack(DWORD dwCard, PrcVidMpegCallBack ppCall)

说明：注册视频 MPEG 压缩数据回调.

参数：dwCard： 卡号.

ppCall：需要注册的回调函数指针

返回值：TRUE: 成功

FALSE: 失败.

● BOOL WINAPI VCARegVidSaveErrCallBack(DWORD dwCard, PrcVidSaveErrCallBack ppCall)

说明：注册视频捕获存盘出错回调.

参数： dwCard： 卡号.

ppCall：需要注册的回调函数指针

返回值：TRUE: 成功

FALSE: 失败.

● BOOL WINAPI VCARegAudCapCallBack(DWORD dwCard, PrcAudCapCallBack ppCall)

说明： 注册音频捕获原始数据回调.

参数：dwCard：卡号.

ppCall：需要注册的回调函数指针

返回值：TRUE: 成功

FALSE: 失败.

● BOOL WINAPI VCARegAudMpegCallBack(DWORD dwCard, PrcAudMpegCallBack ppCall)

说明：注册音频MPEG压缩数据回调.

参数：dwCard： 卡号.

ppCall：需要注册的回调函数指针

返回值：TRUE: 成功

FALSE：失败.

● BOOL WINAPI VCARegAudSaveErrCallBack(DWORD dwCard, PrcAudSaveErrCallBack ppCall)

说明：注册音频频捕获存盘出错回调.

参数：dwCard：卡号.

ppCall：需要注册的回调函数指针

返回值：TRUE:成功

FALSE：失败.

● BOOL WINAPI VCARegIoAlertCallBack(DWORD dwCard, PrcIoAlertCallBack ppCall)

说明：注册 IO 报警回调.

参数： dwCard： 卡号.

ppCall： 需要注册的回调函数指针

返回值：TRUE：成功

FALSE： 失败.

● void WINAPI VCASetLastError(DWORD dwError)

说明： 设置错误序号.

参数： dwError：错误序号.

返回值：无

● DWORD WINAPI VCAGetLastError()

说明：得到错误序号.

返回值：错误序号

● BOOL WINAPI VCAInitSdk(HWND hWndMain, BOOL bInitVidDev = TRUE, BOOL bInitAudDev = TRUE)

说明:初始化系统资源.

参数：hWndMain：应用程序主窗口句柄.

返回值：TRUE： 成功

FALSE： 失败.

参数： bInitVidDev

返回值：TRUE： 初始化视频设备

FALSE： 不初始化视频设备此时视频设备可以进行视频数据回调但是不能视频预览

参数： bInitAudDev

返回值：TRUE： 初始化音频设备

FALSE： 不初始化音频设备

● void WINAPI VCAUnInitSdk ()

说明： 释放系统资源.

参数： 无.

返回值：无

● LONG WINAPI VCAGetDevNum()

说明：返回系统当中卡号数量，即为 SAA7134 硬件数目,为 0 时表示没有设备存在.

参数：无

返回值：系统中已经安装好的 SAA7134 数量.

● BOOL WINAPI VCAOpenDevice(DWORD dwCard, HWND hPreviewWnd)

说明：打开指定卡号的设备，分配相应系统资源

参数：dwCard：卡号.

HPreviewWnd: 指定卡号将要视频预览的窗口句柄，指定卡号的预览分辨率为该窗口的尺寸大小.

返回值：TRUE: 成功

FALSE: 失败.

备注:系统将自动给定传入子窗口的大小来显示视频. 预览窗口的分辨率为该窗口的尺寸大小,需要指出的是,当视频格式为 PAL 时,显示分辨率最大为 1024×574,当视频格式为 NTSC 时显示分辨率最大为 1020×470,当窗口尺寸大于最大分辨率时，可通过调用 OVERLAY 映射函数 VCAResetMapRegion 来解决.

● BOOL WINAPI VCACloseDevice(DWORD dwCard)

说明： 关闭指定卡号的设备，释放相应系统资源.

参数： dwCard：卡号.

返回值：TRUE: 成功

FALSE: 失败.

● BOOL WINAPI VCAStartVideoPreview(DWORD dwCard)

说明：开始视频预览.

参数：dwCard：卡号.

返回值：TRUE: 成功

FALSE: 失败.

● BOOL WINAPI VCAStopVideoPreview(DWORD dwCard)

说明：停止视频预览.

参数： dwCard：卡号.

返回值：TRUE: 成功

FALSE: 失败.

● BOOL WINAPI VCAUpdateVideoPreview(DWORD dwCard, HWND hPreviewWnd)

说明： 更新视频预览。

参数：dwCard：卡号.

hPreviewWnd：新的预览窗口句柄.

返回值：TRUE： 成功.

FALSE： 失败.

备注: 用户改变该窗口的大小后想更新分辨率大小（即为改变后窗口的尺寸大小）,或这路视频希望显示到其它

窗口.

BOOL WINAPI VCAResetMapRegion(RECT* stuSrcRect, RECT* stuDesRect)

说明：设置Overlay屏幕区域到用户屏幕区域的映射关系。系统初始化时创建了一个同屏幕大小一致的Overlay屏幕区域，我们称之为逻辑屏幕区域，用户屏幕区域则是指我们所看到的实际屏幕区域。VCAOpenDevice函数中传入的窗口，它的矩形区域实际上是在逻辑屏幕区域的绝对位置，由于系统初始化时两区域的1：1关系，所以我们可以得到同坐标同大小用户屏幕区域的视频图像。利用此函数则可重建此映射关系，以改变图像大小/位置等比例关系 。

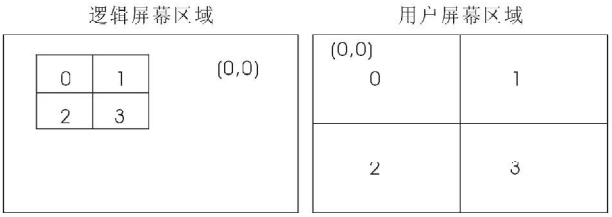
参数： stuSrcRect： Overlay 逻辑屏幕区域指针

stuDesRect： 用户屏幕区域指针。

返回值：TRUE： 成功

FALSE： 失败.

例： 我们在逻辑屏幕上相临区域创建了四路(卡)视频输出，其中有0 1 2 3窗口，四路(卡)分别输出在窗口里面，如在通常1：1映射方式下工作，这四路输出将显示到用户屏幕的左下同坐标区域，大小不变，现在我们改变映射关系，将此逻辑区域映射到全屏，则用户得到视频图像输出如右图所示。我们以第0路视频输出为例，1、改变0窗口的尺寸和位置等于0路用户屏幕区域2、调用VCAResetMapRegion函数，stuSrcRect参数应传入0路输出所占用区域在逻辑屏幕中的矩形坐标（即0路窗口改变前的区域），而stuDesRect参数则传入0路输出所占用区域在用户屏幕中的矩形坐标（即0路窗口改变后的区域）。1 2 3路雷同。



● BOOL WINAPI VCASaveBitsToBuf(DWORD dwCard, PVOID pDestBuf, DWORD& dwWidth, DWORD& dwHeight)

说明： 保存快照数据到相应的缓冲区.

参数： dwCard：卡号.

pDestBuf：缓冲区的指针.

dwWidth: 返回图像的宽度.

dwHeight: 返回图像的高度.

返回值：TRUE： 成功

FALSE： 失败.

● BOOL WINAPI VCASaveAsJpegFile(DWORD dwCard, LPCTSTR lpFileName, DWORD dwQuality = 100)

说明： 保存快照为JPEG文件.

参数： dwCard：卡号.

lpFileName：所需的文件名称及路径.

DwQuality: JPEG 文件质量,范围:1~100

返回值：TRUE： 成功

FALSE： 失败.

● BOOL WINAPI VCASaveAsBmpFile(DWORD dwCard, LPCTSTR lpFileName)

说明： 保存快照为 BMP 文件.

参数： dwCard：卡号.

pDestBuf：所需的文件名称及路径.

返回值：TRUE： 成功

FALSE： 失败.

● BOOL WINAPI VCAStartVideoCapture(DWORD dwCard,

说明： 开始视频捕获.

参数： dwCard：卡号.

lpFileName：所需的文件名称及路径.

enCapMode: 选择流捕捉模式.

enMp4Mode: MPEG 压缩模式，只有 enCapMode 等于 CAP_MPEG4_STREAM 时 enMp4Mode 才有效.

返回值：TRUE 成功,

FALSE 失败

● BOOL WINAPI VCAStartVideoCapture(DWORD dwCard,

CAPMODEL enCapMode,

MP4MODEL enMp4Mode,

LPCTSTR lpFileName)

● BOOL WINAPI StartVideoCapture(DWORD dwCard)

说明：开始视频捕获.

参数：dwCard：卡号.

返回值：TRUE 成功

FALSE. 失败

● BOOL WINAPI VCAStopVideoCapture(DWORD dwCard)

说明： 停止视频捕获.

参数： dwCard：卡号.

返回值：TRUE 成功

FALSE. 失败

● BOOL WINAPI VCASetVidCapSize(DWORD dwCard, DWORD dwWidth, DWORD dwHeight)

说明： 设置视频捕获尺寸.

参数： dwCard：卡号.

dwWidth：捕获宽度.

DwHeight：捕获高度.

返回值：TRUE 成功

FALSE 失败

● BOOL WINAPI VCAGetVidCapSize(DWORD dwCard, DWORD &dwWidth, DWORD &dwHeight)

说明： 得到视频捕获尺寸.

参数： dwCard：卡号.

dwWidth：捕获宽度.

DwHeight：捕获高度.

返回值：TRUE 成功

FALSE 失败

●BOOL WINAPI VCASetVidCapFrameRate(DWORD dwCard, DWORD dwFrameRate, BOOL bFrameRateReduction = FALSE)

说明： 设置视频捕获帧率.

参数： dwCard：卡号.

dwFrameRate：捕获帧率.

bFrameRateReduction：是否允许显示帧率随捕获帧率而减小.

TRUE：显示帧率随捕获帧率而减小，视频捕获帧率决定视频预览帧率.

FALSE：显示帧率不变，始终为最大帧率，PAL 最大帧率为 25，NTSC 最大帧率为 30.

返回值：TRUE 成功

FALSE 失败

备注：当 bFrameRateReduction 为 TRUE 时，有助缓解多路视频同时捕获时 PCI 总线的负荷量。下表是视频捕获帧率与视频预览帧率的对比

PAL捕获帧率	PAL显示帧率	NTSC捕获帧率	NTSC显示帧率
1	1	1	1
2	2 + 1/2	2	2
3	3 + 1/8	3	3
4	4 + 1/6	4	4 + 2/7
5	5	5	5
6	6 + 1/4	6	6
7	8 + 1/3	7	7 + 1/2
8	8 + 1/3	8	10
9	12 + 1/2	9	10
10	12 + 1/2	10	10
11	12 + 1/2	11	15
12	12 + 1/2	12	15
13	25	13	15
14	25	14	15
15	25	15	15
16	25	16	30
17	25	17	30
18	25	18	30
19	25	19	30
20	25	20	30
21	25	21	30
22	25	22	30
23	25	23	30
24	25	24	30
25	25	25	30
		26	30
		27	30
		28	30
		29	30
		30	30

● BOOL WINAPI VCASetBitRate(DWORD dwCard, DWORD dwBitRate)

说明： 设置MPEG压缩的位率.

参数： dwCard： 卡号.

dwBitRate： MPEG压缩的位率.范围56KBPS ~ 10MBPS

返回值：TRUE 成功

FALSE 失败

● BOOL WINAPI VCASetKeyFrmInterval(DWORD dwCard, DWORD dwKeyFrmInterval)

说明： 设置MPEG压缩的关键帧间隔，必须大于等于帧率.

参数： dwCard： 当前操作的卡号.

dwKeyFrmInterval： MPEG压缩的关键帧间隔.

返回值：TRUE 成功

FALSE 失败

● BOOL WINAPI VCAEnableMotionDetect(DWORD dwCard,

BOOL bEnaDetect,

BYTE *pAreaMap,

LONG nSizeOfMap,

LONG nPersistTime,

LPVOID lpContext,

prcCbMotionDetect OnObjectMove)

说明：设置OSD控制功能

参数：dwCard： 卡号.

bEnaDetect： TRUE：允许动态检测，后续参数有效。

FALSE：禁止动态检测，系统忽略后续参数。

pAreaMap：系统将捕获视频划分为16*16点阵的动态检测区域，按从左到右、从上到下的次序编号，依次为0、1、2、…，其中每一编号分别对应pAreaMap中的一个BYTE，pAreaMap数组中某一单元的BYTE值表示对应子块的动态检测灵敏度，灵敏度范围从0-49，0表示最高灵敏度，49表示最低灵敏度，如该值大于49，系统不对该块进行动态检测。

nSizeOfMap： pAreaMap数组的大小，pAreaMap数组应由用户分配，其大小计算公式如下：

设nCapWidth,nCapHeight分别为捕获视频的宽度和高度，nDetectWidth,nDetectHeight为以子块为单位的检测块宽度和高度,pAreaMap数组的大小定义为nDetectBlocks。

```
nDetectWidth   =  (long)(nCapWidth / 16);
nDetectHeight  =  (long)(nCapHeight / 16);
nDetectBlocks  =  nDetectWidht * nDetectHeight;
```

即：系统以16为模对捕获视频宽度和高度进行划分，多余部分则自动丢弃而不做检测。

nPersistTime： 用户设定的静止检测持续时间(以秒为单位)，即当系统检测到物体的一次运动后，当

持续检测到nPersistTime 秒没有发生物体运动才认为物体进入静止状态，该值取值范围从0 - 30s。

lpContext：它会作为回调函数的最后一个参数返回给用户，用户可以在该成员中保存一些私有信息。如果不需要的话，可以传入NULL 值。

OnObjectMove: 用于通知用户动态检测状态的回调函数指针。每当系统检测到有物体移动时,该函数被调用一次。当物体停止移动时，该函数再被调用一次，不能为NULL值。

返回：TRUE： 成功

FALSE.：失败

● BOOL WINAPI VCASetVidDeviceColor(DWORD dwCard, COLORCONTROL enCtlType, DWORD dwValue)

说明： 设置视频颜色属性,它将影响视频预览和视频捕获的显示属性.

参数： dwCard：卡号.

EnCtlType：颜色类型.

DwValue：颜色属性的值.

返回值：TRUE： 成功

FALSE： 失败.

● BOOL WINAPI VCAGetVidFieldFrq(DWORD dwCard, eFieldFrequency &eVidSourceFieldRate)

说明： 得到视频源输入频率，即可得到视频源输入制式，参考eVidSourceFieldRate 类型说明

参数： dwCard： 卡号.

eVidSourceFieldRate：视频源输入频率.

返回值： TRUE： 成功

FALSE： 失败.

● BOOL WINAPI VCAStartAudioCapture(DWORD dwCard,

CAPMODEL enCapMode,

MP4MODEL enMp4Mode)

说明： 开始音频捕获.

参数： dwCard：卡号.

lpFileName：所需的文件名称及路径.

enCapMode: 选择流捕捉模式.

enMp4Mode: MPEG 压缩模式，只有enCapMode 等于CAP_MPEG4_STREAM 时enMp4Mode 才有效.

返回值：TRUE 成功,

FALSE 失败

备 注：

1、当音频enCapMode 等于 CAP_MPEG4_STREAM 时，视频捕获必须比音频捕获先运行，而且视频enCapMode 必须等于CAP_MPEG4_STREAM,

2、当音频enMp4Mode 等于 MPEG4_AVIFILE_ONLY 或MPEG4_AVIFILE_CALLBACK 时，视频enMp4Mode 必须等于 MPEG4_AVIFILE_ONLY 或MPEG4_AVIFILE_CALLBACK.

3.音频原始数据频率为32KHZ.

● BOOL WINAPI VCAStopAudioCapture(DWORD dwCard)

说明：停止音频捕获。

参数：dwCard： 卡号.

返回：TRUE： 成功

FALSE： 失败

备注：当 dwCard >= 16 时，为静音；开始音频捕获时有效.

● BOOL WINAPI VCASetAudioCardOutOn (DWORD dwCard)

说明：停在音频捕获。

参数：dwCard： 卡号.

返回：TRUE： 成功

FALSE： 失败

备注：当音频 enCapMode 等于 CAP_MPEG4_STREAM 时，音频捕获必须比视频捕获先停止

● BOOL WINAPI VCASetIOAlertLevelIn(DWORD dwCard, eVOLTAGELEVEL enCurrentLevel)

说明：设置 IO 输入报警当前电平。

参数：dwCard：卡号。

enCurrentLevel: IO 输入电平状态

返回：TRUE： 成功

FALSE： 失败

● BOOL WINAPI VCAStartIOAlertIn(DWORD dwCard, eVOLTAGELEVEL enSpringLevel)

说明：开始 IO 输入报警。

参数：dwCard： 卡号.

enSpringLevel：IO 输入触发电平.

返回：TRUE： 成功

FALSE： 失败

● BOOL WINAPI VCAStopIOAlertIn(DWORD dwCard)

说明：停止 IO 输入报警.

参数：dwCard: 卡号.

返回：TRUE： 成功

FALSE： 失败

● BOOL WINAPI VCAIOAlertOut(DWORD dwCard, eVOLTAGELEVEL enLevel)

说明：I O 电平输出。

参数：dwCard: 卡号.

EnLevel: IO 电平输出状态。

返回：TRUE 成功

FALSE 失败.

● BOOL WINAPI VCASetVidCapTextOSD(DWORD dwCard, BOOL bEnableOSD, TCHAR* tcText, POINT ptTopLeft, BOOL bTransparent, OSDPARAM* pOSDParm)

说明： 设置视频文本叠加.

参数： dwCard： 卡号.

bEnableOSD： 是否允许叠加.

tcText： 叠加文本.

ptTopLeft： 叠加位置.

bTransparent：是否透明.

pOSDParm: 叠加参数.

返回值：TRUE： 成功

FALSE： 失败.

备注：视频捕获停止时或捕获暂停时调用有效

● BOOL WINAPI VCASetVidCapDateTimeOSDParam(OSDPARAM* pOSDParm)

说明： 设置视频时间、日期叠加参数.

参数： pOSDParm:叠加参数.

返回值：TRUE： 成功

FALSE： 失败.

备注：该函数用于所有的视频时间、日期叠加，一旦设置，所有视频时间、日期叠加参数将一直。视频捕获停止时或捕获暂停时调用有效

● BOOL WINAPI VCASetVidCapDateTimeOSD(DWORD dwCard, BOOL bEnableOSD, POINT ptTopLeft, BOOL bTransparent)

说明： 设置视频时间、日期叠加.

参数： dwCard： 卡号.

bEnableOSD： 是否允许叠加.

tcText： 叠加文本.

ptTopLeft：叠加位置.

bTransparent：是否透明.

返回值：TRUE： 成功

FALSE： 失败.

备注：在VCASetVidCapDateTimeOSDPARAM()函数设置后并且视频捕获停止时或捕获暂停时调用有效调用有效。

第四章　VC404P_401P 网络传输函数说明

4.1 系统常量

系统错误常量定义

#define	ERR_SYS_NOTMCARD	128	没有检测到 VC404 卡
#define	ERR_SYS_UNINITIALIZE	129	系统未初始化
#define	ERR_SYS_OUTOFMEMORY	130	内存资源请求错误
#define	ERR_SYS_NONEWCALL	131	没有申请 Call 资源
#define	ERR_SYS_NOCONNCALL	132	没有连接的呼叫
#define	ERR_SYS_NONSTANDPARAM	133	使用了非标准输入参数
#define	ERR_SYS_SETSOCKETBUF	134	设置网络收发缓冲时出错
#define	ERR_SYS_ROLEACT	135	执行动作与角色冲突
#define	ERR_SYS_OUTOFIDRANGE	136	非法呼叫 id 号
#define	ERR_SYS_WANMULTICAST	137	广域网多播错误
#define	ERR_SYS_MULTICASTADDR	138	多播地址错误
#define	ERR_SYS_JOINMULTIGROUP	139	加入多播组错误
#define	ERR_SYS_CREATWINDOW	140	显示子窗体创建出错
#define	ERR_SYS_CREATEOVERLAY	141	overlay 表面创建错误
#define	ERR_SYS_INVALIDWINDOW	142	创建视频设备时输入参数错误
#define	ERR_SYS_CAPMODE	143	捕获模式错误
#define	ERR_SYS_CREATECAPFILE	144	创建捕获文件时出错
#define	ERR_SYS_CAPAUDIO	145	音频捕获存盘错误
#define	ERR_SYS_CAPVIDEO	146	视频捕获存盘错误
#define	ERR_SYS_MPEG4	147	MPEG4 初始化错误
#define	ERR_SYS_INVALDOBJECTID	48	无效的对象 ID,对象未初始化
#define	ERR_SYS_OPENPLAYFILE	149	打开文件时操作异常
#define	ERR_SYS_PLAYFILETYPE	150	未知的文件类型
#define	ERR_SYS_PLAYFILEFORMAT	151	播放文件格式错误
#define	ERR_SYS_GETBASETIME	152	不能获取播放文件基准时间
#define	ERR_SYS_UNINITPLAY	153	回放操作未成功初始化
#define	ERR_SYS_PLAYLOCATE	154	回放文件定位错误
#define	ERR_SYS_PLAYMODE	155	无效的播放模式

请求操作返回值定义：

#define	ERR_CMD_SUCCESS	0	请求成功
#define	ERR_CMD_TIMEOUT	1	请求超时
#define	ERR_CMD_INVALID	2	无效的请求命令
#define	ERR_CMD_OUTOFMEMORY	3	请求操作时资源分配错误
#define	ERR_CMD_OUTOFUSER	4	该卡的用户请求已超过最大值
#define	ERR_CMD_REQ_EXIST	5	请求操作重复
#define	ERR_CMD_OUTOFRESOURCE	6	服务端网络资源创建错误

请求操作返回值范围为（0-63）。

其中：(0-31）由系统内部使用，(32-63）可由用户自定义

请求操作常量定义

#define	REQ_AV_STREAM	1	请求视频+音频流
#define	REQ_VI_STREAM	2	请求视频流
#define	REQ_AU_STREAM	3	请求音频流
#define	REQ_CLEAR_AV	4	请求关闭 AV 流
#define	REQ_CLEAR_VI	5	请求关闭视频流
#define	REQ_CLEAR_AU	6	请求关闭音频流
#define	NTF_FREE_CALL	16	服务端强行终止该路呼叫

请求命令取值范围为：(0-127)

其中： 0-15 供客户端系统请求使用；

16-31 供服务端/代理端通知请求使用；

32-127 用户自定义

媒体流捕获模式定义：

#define	SAVE_VIDEO_DATA	0x1	存储视频流
#define	SAVE_AUDIO_DATA	0x2	存储音频流

媒体流播放模式定义：

#define	PLAY_VIDEO_DATA	0x1	播放视频流
#define	PLAY_AUDIO_DATA	0x2	播放音频流

4.2 函数参考说明

API 列表 ()

强烈建议：请不要在回调函数中作MTAClearCall操作以及其他等待（如MessageBox等）操作，否则可能会导致未知结果。

发送端(服务端)专用函数

函数	功能
MTARegServer	注册服务器配置信息到代理端。
MTANotify MTAWriteVideo	服务端向客户端发送命令请求或通知。 发送视频数据帧。该函数须实时调用。
MTAWriteAudio	发送音频数据块。该函数须实时调用。
MTADiscardCall	服务端强制停止一路呼叫。

MTARegServer

注册服务器配置信息到代理端

```
BOOL MTARegServer ( const char        *lpRegUserName,
                    const char        *lpRegPassword,
                    const char        *lpSupplyAddr,
                    WORD               usSupplyPort,
                    EvCmdRespond      evCallNotify);
```

参数：lpRegUserName：　　注册用户名

lpRegPassword：　　注册密码

lpSupplyAddr：　　代理端地址

usSupplyPort：　　代理端端口

evCallNotify　　注册结果回调处理函数

返回:布尔值,该值仅表明当前命令是否正确执行,返回真值并不表明注册成功,最终须通过回调函数取得注册结果值。返回错误时可调用MTAGetLastError得到错误代码。该函数为服务器端专用。

MTANotify

服务端向客户端发送命令请求或通知。

```
LONGLONG MTANotify(char          *pClientAddr,
WORD                        usClientPort,
BYTE                        biReqType,
BYTE                        biMaxRepeat,
BYTE                        *pAnnexData,
WORD                        inDataSize,
BYTE                        *pRetBuffer,
WORD                        retBufSize,
HANDLE                      hEventNotify,
```

EvCmdRespond evCallNotify);

参数：

pClientAddr： 客户端 IP 地址。

usClientPort： 客户端端口号。

biReqType： 通知码。可为自定义的通知。

biMaxRepeat： 最大超时重发次数。

pAnnexData： 附带发送数据的缓冲指针。

inDataSize: 附带数据长度。

pRetBuffer： 用于接收返回结果数据的缓冲指针。

retBufSize： 结果缓冲区大小。

hEventNotify： 获取返回结果的通知事件句柄，可为 NULL。设置此事件句柄后，一个参数（回调函数指针）将被忽略。

evCallNotify： 获取返回结果的回调函数指针。

返回：成功返回请求命令 ID 号，否则返回 -1，可用 MTAGetLastError 得到错误码。

备注：EvCmdRespond 类型定义请参见 MTAMakeCall 中说明。该函数实现实际上是 MTARequest 函数的另一版本，发送方向为服务端 ->客户端。

MTAWriteVideo

发送视频数据帧。该函数须实时调用。

```
void MTAWriteVideo(    BYTE    nCardNo,
                       BYTE     *pData,
                       LONG  lSize,
                       BOOL  bIFrm
                   );
```

参数：

nCardNo： 卡号，编号以 0 为基数，0- 第一块卡，1- 第二块卡，依次类推。

pData： 视频数据的缓冲区指针。

lSize： 视频数据大小。

bIFrm： 是否 I 帧。

返回：无。

MTAWriteAudio

发送音频数据块。该函数须实时调用。

```
void MTAWriteAudio(BYTE  nCardNo, BYTE *pData, LONG  lSize);
```

参数：nCardNo： 卡号，编号以 0 为基数，0- 第一块卡，1- 第二块卡，依次类推。

pData： 音频数据的缓冲区指针。

lSize： 音频数据大小。

返回：无。

MTADiscardCall

服务端强制停止一路呼叫。

```
void MTADiscardCall(   BYTE          biCard,
                       DWORD       dwRemoteIp,
                       WORD        usRemotePort,
                       BOOL          bWait);
```

参数：biCard： 卡号，编号以 0 为基数。

dwRemoteIp： 要强制终止的客户端呼叫的 IP 地址。

usRemotePort： 要强制终止的客户端呼叫的端口号。

bWait： 函数是否等待结果完成后才返回。

返回：无。

接收端（客户端）专用函数

函数	功能
MTAGetSrvAddr	取回呼叫服务器的配置信息。
MTAConfigNetwork	设置单路呼叫的网络收发缓冲区大小。
MTANewCall	申请呼叫资源。
MTAMakeCall	发起呼叫，请求媒体传送服务。
MTAClearCall	清除呼叫，释放系统资源。
MTAGetNetTraffic	取得对应的呼叫的视频传输流量统计值。
MTAStartCapture	始捕获媒体流数据，存入指定文件。
MTAStopCapture	停止音视频数据的捕获操作，关闭存储文件。
MTARequest	向服务端发送一个请求或命令。
MTARequestStream	请求打开（/关闭）音频（/视频）流传送服务。

MTAGetSrvAddr

取回呼叫服务器的配置信息

```
BOOL MTAGetSrvAddr( int   nCallID,
                    const char    *lpServerName,
                    const char    *lpSupplyAddr,
                    WORD          usSupplyPort);
```

参数：nCallID： Call ID。

lpServerName： 待呼叫服务器的注册名

lpSupplyAddr： 代理端地址

usSupplyPort：　　代理端端口

返回：

布尔型，表明执行结果。如果成功返回，则紧接着可调用 MTAMakeCall 发起呼叫。如返回 FALSE,调用 MTAGetLastError 得到错误代码。该函数为客户端专用。

MTANewCall

申请呼叫资源。

```
int   MTANewCall(char *pRemoteIp,WORD usRemotePort,BYTE biCardNo);
```

参数：

pRemoteIp：　　远端主机 IP 地址。

usRemotePort：　　远端主机帧听端口号。

biCardNo：　　请求卡号。

返回：成功返回呼叫号，否则返回 -1。调用 MTAGetLastError 返回错误码

备注：调用此函数之后，如果成功，须调用 MTAClearCall 函数释放相应的资源。

MTAMakeCall

发起呼叫，请求媒体传送服务。

```
BOOL  MTAMakeCall(int nCallID,
BYTE biReqType,
BOOL bWANCall,
BYTE *lpSndData,
BYTE biSndSize,
HANDLE hEventNotify,
EvCmdRespond evCallNotify);
```

参数：

nCallID：　　呼叫号，由 MTANewCall 返回。

biReqType：　　请求类型，取值如下：

REQ_AV_STREAM ：同时请求音视频传送

REQ_VI_STREAM ：仅请求视频传送

REQ_AU_STERAM ：仅请求音频传送

bWANCall：　　是否广域网呼叫。（主要用于改善广域网上音频流播放的连续性）

lpSndData：　　发送到服务器端的附带数据缓冲指针。

biSndSize：　　附带数据长度。(不能超过 1200 字节)

hEventNotify：　　获取请求结果的通知事件句柄，如果设置，下一参数将被忽略。

evCallNotify：　　得到请求结果的回调函数指针。

◆ EvCmdRespond 回调函数类型定义

```
typedef void (*EvCmdRespond)(  int                 nCallId,
                               unsigned char biCmd,
                               int             nResult,
                               BYTE            *pRetData);
```

nCallID： 呼叫号，由 MTANewCall 分配

biCmd： 用户发起的请求命令类型；

nResult： 请求返回的结果；返回 ERR_CMD_SUCCESS 表示成功

pRetData：接收端处理请求后发送回来的附带数据。该缓冲是由用户

在调用请求操作时分配，返回数据在缓冲中的布局如下：

■ 2 字节后续数据总长度（不包括本身 2 字节长度）

■ 2 字节返回结果，等同于参数 nResult

■ 实际返回数据（有效长度 = 总长度 – 2）

具体到本函数调用，由于没有提供输出缓冲参数定义，则用户不能（也不必）从服务端获取附加返回数据。

返回：成功返回 TRUE，否则 FALSE。调用 MTAGetLastError 得到错误码。

备注：在 evCallNotify 回调函数中调用 MTAClearCall 函数会导致未知结果。

MTAConfigNetword

设置单路呼叫的网络收发缓冲区大小。

```
BOOL MTAConfigNetwork(int nCallID,int &iSndBufSize,int &iRcvBufSize);
```

参 数 ：

nCallID： 呼叫号；

iSndBufSize： 输入新的发送缓冲区大小，返回上一次的大小配置。

iRcvBufSize 输入新的接收缓冲区大小，返回上一次的大小配置。

单位：字节（BYTE）

返回：成功返回 TRUE，失败返回 FALSE。调用 MTAGetLastError 得到错误码。

备注：当输入参数为 0 时，系统返回当前配置（发送 / 接收），一般情况下不使用该函数。

MTAClearCall

清除呼叫，释放系统资源。

```
void MTAClearCall(int nCallId,BOOL bWait);
```

参数：nCallId ：呼叫号，由 MTANewCall 返回。

bWait： 是否阻塞直到清除操作完成。

返回：无

MTAGetNetTraffic

取得对应的呼叫的视频传输流量统计值。

void MTAGetNetTraffic(int nCallID,TRAFFIC_PARAM &TrafficBuf);

参数：nCallID：　　呼叫号，由MTANewCall返回。

　　TrafficBuf：　　输出参数，返回传输流量统计值。

◆　结构TRAFFIC_PARAM定义如下：

typedef struct tagTrafficStatus

{

DWORD Rxbitrate;　　当前位率统计值

DWORD Framerate;　　当前帧率统计值

DWORD IFramerate;　　当前I帧帧率统计值

DWORD PFramerate;　　当前P帧帧率统计值,暂不用

}

TRAFFIC_PARAM;

返回：无

备注：

用户须以1秒为单位周期性调用该函数才能获得正确的统计值，上述参数值会一直累加，直到用户调用才会清零。

MTAStartCapture

开始捕获媒体流数据，存入指定文件。

BOOL MTAStartCapture(int　nCallID,

　　const char *　pFileName,

　　DWORD　　dwMode,

　　EvCaptureAbnormity evCapAbnormity);

参数：

nCallID：　　呼叫号，由MTANewCall返回。

pFileName：　　捕获文件名，捕获数据将保存到该文件中。

dwMode：　　操作类型，取值如下：

SAVE_VIDEO_DATA：保存视频数据

SAVE_AUDIO_DATA：保存音频数据

可利用或操作实现音视频同时存储。

evCapAbnormity：捕获异常时的回调函数指针，当捕获数据过程中发生错误时，该回调函数被调用，可为空。

◆　EvCaptureAbnormity 回调函数类型定义如下：

typedef void (*EvCaptureAbnormity)(int nCallID,int nError);

参数： nCallID：呼叫号，标识客户端与发送端的一个请求；

nError： 异常媒体类型

可能错误：ERR_SYS_CAPAUDIO 音频流捕获存盘错误

ERR_SYS_CAPVIDEO 视频流捕获存盘错误

返回：成功返回 TRUE，否则 FALSE，调用 MTAGetLastError 返回错误码。

MTAStopCapture

停止音视频数据的捕获操作，关闭存储文件。

```
void MTAStopCapture(int nCallID);
```

参数：nCallID：呼叫号，由 MTANewCall 分配。

返回：无

MTARequest

向服务端发送一个请求或命令。

```
LONGLONG MTARequest(int      nCallID,
BYTE                 biReqType,
BYTE                 biMaxRepeat,
BYTE                 *pAnnexData,
WORD                 inDataSize,
BYTE                 *pRetBuffer,
WORD                 retBufSize,
HANDLE               hEventNotify,
EvCmdRespond         evCallNotify);
```

参数：

nCallID	呼叫号，由 MTANewCall 分配
biReqType	请求类型。可以为自定义的请求。
biMaxRepeat	请求数据最大超时重发次数。
pAnnexData	请求操作附带发送数据的缓冲指针。
inDataSize	附带数据大小。
pRetBuffer	用于接收远端响应数据的缓冲区指针。（在事件通知或回调函数中会被用到，为 NULL 则不返回数据）
retBufSize	结果缓冲的大小。
hEventNotify	获取请求结果的通知事件句柄，可为 NULL。设置此事件句柄后，下一个参数（回调函数指针）将被忽略。可通过 pRetBuffer 得到附加远端返回数据（如果有）。
evCallNotify	获取请求响应结果的回调函数指针。

返回：成功返回此请求的 ID 号，否则为 -1，可用 MTAGetLastError 得到错误码。

备注：EvCmdRespond 类型定义请参见 MTAMakeCall 中说明。

MTARequestStream

请求打开（/ 关闭）音频（/ 视频）流传送服务。

```
BOOL  MTARequestStream(int          nCallID,
BOOL                   bVStream,
BOOL                    bOpen,
BYTE                     *lpSndData,
BYTE                    biSndSize,
HANDLE                hEventNotify,
EvCmdRespond      evCallNotify);
```

参数：

nCallID：　　呼叫号，由 MTANewCall 分配；

bVStream：　　是否请求对视频流操作，为 FALSE 表示对音频流操作。

bOpen：:　　是打开(TRUE)还是关闭(FALSE)指定的流。

lpSndData：　　附带数据指针。

biSndSize：　　附带数据大小。

hEventNotify：　　获取请求结果的通知事件句柄，可为 NULL。设置此事件句柄后，下一个参数（回调函数指针）将被忽略。

evCallNotify：　　获取请求响应结果的回调函数指针。

返回：成功返回 TRUE，否则 FALSE。调用 MTAGetLastError 返回错误码。。

备注：EvCmdRespond 类型定义请参见 MTAMakeCall 中说明。

```
BOOL  MTASetMpeg4Version(int nVersion)
```

参数：nVersion：Microsoft MPEG4 版本，取值范围 1-3，分别代表版本 V1/V2/V3。

返回：成功返回 TRUE，否则 FALSE，表示输入参数超出范围。

发送端和接收端共用函数

函数	功能
MTALoadLibrary	初始化开发包，分配相应系统资源
MTAFreeLibrary	释放开发包分配的系统资源。
MTACreateVideoDevice	创建视频显示窗口和Overlay显示表面。
MTASetSplitMode	设置视频窗口的显示切分模式。
MTASetWndPos	设定视频显示窗口的位置。
MTAPageDown	向下翻页，直到当前显示模式下的最后一页。
MTAPageUp	向上翻页，直到当前显示模式下的第一页。
MTAGetPageNo	取得当前页号。
MTASetVideoOut	将指定呼叫的视频信号输出显示到指定的位置.
MTAGetCallIDByIndex	获得显示位置所对应的呼叫。
MTAGetIndexByCallID	取得呼叫所对应的视频显示输出位置。

MTACreateAudioDevice	初始化音频设备，分配相应资源，须DirectSound支持。
MTASetAudioPass	将呼叫的音频输出到音频设备。
MTASendMessage	向视频显示窗口发送消息。
MTAGetLastError	得到上一次操作的错误码
MTASetLastError	设置当前的错误码
MTASetRequestTimeOut	设置请求超时的时间长度
MTACalculateCrc	计算数据块的16位CRC校验码。
MTARegNotifier	注册用于接收请求的回调函数。

MTALoadLibrary

初始化开发包，分配相应系统资源。

```
BOOL MTALoadLibrary(WORD usLocalPort,
                    WORD usRole);
```

参数：

usLocalPort： 本地侦听端口号。

usRole： 角色定义，取值如下：

WORK_AS_SERVER 发送端：此时 usLocalPort 为本地帧听端口号

WORK_AS_CLIENT 接收端：此时 usLocalPort 为本地套接字绑定端口号

WORK_AS_SUPPLY 代理端：同 WORK_AS_SERVER

返回：成功返回 TRUE，失败返回 FALSE，调用 MTAGetLastError 得到错误类型。

MTAFreeLibrary

释放开发包分配的系统资源。

```
void MTAFreeLibrary()
```

参数：无

返回：无

MTACreateVideoDevice

创建视频显示窗口和Overlay显示表面。

```
int  MTACreateVideoDevice( HWND hParentWnd,
                           HWND hNotifyWnd,
                           RECT  rect,
                           int      nWidth,
                           int      nHeight,
                           int      nSpace,
                           BOOL  bUseOverlay);
```

参数：hParentWnd：视频显示窗口的父窗口句柄，不能为空。

hNotifyWnd：视频显示通知消息的接收窗口句柄，不能为空。

rect：　　视频显示窗相对于父窗口的大小位置，采用相对坐标。

nWidth：　　单路视频的宽度，如320等。

nHeight：　　单路视频的高度，如240等。

nSpace：　　多路视频同屏显示时它们之间的间隔宽度。

bUseOverlay：是否使用Overlay显示方式。

返回：

成功时返回系统允许的最大显示模式，否则返回-1，调用MTAGetLastError返回错误代码。

MTASetSplitMode

设置视频窗口的显示切分模式。

int　MTASetSplitMode(int nMode);

参数：

nMode：　视频窗口的显示模式，取值为1到4之间。

1到4分别表示：一屏显示图像1*1路，2*2路，3*3路和4*4路。

返回：

成功时返回1，否则返回小于1的错误码。

MTASetWndPos

设定视频显示窗口的位置。

int　MTASetWndPos (int nPlayBackID);

参数：

rc：　视频窗口的新位置。

返回：

成功返回TRUE,失败返回FALSE

MTAPageDown

向下翻页，直到当前显示模式下的最后一页。

int　MTAPageDown();

参数：无

返回：

成功时返回前一次的页号，否则返回小于1的错误码。

MTAPageUp

向上翻页，直到当前显示模式下的第一页。

int MTAPageUp();

参数：无

返回：

成功时返回前一次的页号，否则返回小于 1 的错误码。

MTAGetPageNo

取得当前页号。

int MTAGetPageNo();

参数：无

返回：页号。

MTASetVideoOut

将 nCallID 指定呼叫的视频信号输出显示到 nIndex 指定的位置.

BOOL MTASetVideoOut(int nCallID,int nIndex);

参数：

nCallID：当前操作的呼叫号.

nIndex： 显示窗口中的相对索引号,取值为 0-15，按由左到右，由上到下的次序排列。

返回：

成功返回 TRUE,失败返回 FALSE，调用 MTAGetLastError 得到错误码。

MTAGetCallIDByIndex

获得显示位置所对应的呼叫。

int MTAGetCallIDByIndex(int nIndex);

参数：nIndex： 显示位置的相对索引号。

返回：如果该显示位置有视频输出，返回对应的呼叫号，否则返回 -1。

MTAGetIndexByCallID

取得呼叫所对应的视频显示输出位置。

int MTAGetIndexByCallID (int nCallID);

参数：nCallID： 呼叫号，调用 MTANewCall 由系统分配。

返回：显示位置，如用户未调用 MTASetVideoOut 函数设置显示输出位置，则返回 -1。

MTACreateAudioDevice

初始化音频设备，分配相应资源，须 DirectSound 支持。

BOOL MTACreateAudioDevice(HWND hMainWnd);

参数：hMainWnd：　与 DirectSound 协同工作的窗口句柄，一般为主窗口句柄。

返回：成功返回 TRUE,失败返回 FALSE。

MTASetAudioPass

将 nCallID 呼叫的音频输出到音频设备。

BOOL MTASetAudioPass(int nCallID);

参数：nCallID：　　呼叫号，调用 MTANewCall 由系统分配。

返回：成功返回 TRUE,失败返回 FALSE，调用 MTAGetLastError 得到错误码。

MTASendMessage

向视频显示窗口发送消息。主要用于父窗口移动或改变大小时通知视频子窗口。

void MTASendMessage(UINT nMessage,WPARAM wParam,LPARAM lParam);

参数：

nMessage　　消息标识，目前仅支持 WM_MOVE 消息。

wParam　　消息参数。

lParam　　消息参数。

返回：无。

详细使用请见 Demo 示例。

MTAGetLastError

得到上一次操作的错误码。

int　MTAGetLastError();

参数：无

返回：错误码。

MTASetLastError

设置当前的错误码。

void　MTAGetLastError(int ErrorNo);

参数：

ErrorNo：错误码

返回：无。

MTASetRequestTimeOut

设置请求超时的时间长度。

void MTASetRequestTimeOut(WORD usTimeOut);

参数：usTimeOut　　请求超时的时间长度，单位毫秒，范围 1000 – 10000

返回：无。

备注：一旦设置了这个时间长度，所有的请求都将以它作为超时标准。

MTACalculateCrc

计算数据块的 16 位 CRC 校验码。主要用于远程请求时对附带数据进行校验。

```
WORD MTACalculateCrc(BYTE *pData,int nSize);
```

参数：pData：　需要计算 CRC 校验码的数据的缓冲指针。

nSize：　数据长度。

返回：16 位 CRC 校验码。

MTARegNotifier

注册用于接收请求的回调函数。如果用户注册了自定义的请求接收回调函数，则当远程请求到达时该回调函数会被调用。（由于客户 / 服务端均可向远端发送请求，故该函数无角色限制）

```
BOOL MTARegNotifier(EvInComingRequest evNotifier);
```

参数：

evNotifier：　回调函数指针。

◆ EvInComingRequest 回调函数类型定义：

```
typedef BYTE (*EvInComingRequest)( DWORD          dwIp,
                                   WORD           usPort,
                                   BYTE           biCardNo,
                                   BYTE           biCmd,
                                   BYTE           *lpInData,
                                   WORD           biInSize,
                                   BYTE           *lpOutData,
                                   WORD           &biOutSize  );
```

参数：dwIP：　命令发送端的 IP 地址；

usPort：　命令发送端的端口号；

biCardNo：　请求卡号；（客户端 ->服务端请求有效）

biCmd：　请求命令；

lpInData：　收到请求的附带数据缓冲指针；

biInSize：　附带数据长度；

lpOutData：　系统提供的返回数据缓冲指针，用户可将附加返回信息置入该缓冲由底层系统发送给请求端。

biOutSize：　输入 / 输出参数，输入返回数据缓冲的最大长度，输出实际返回数据长度。

返回：ERR_CMD_SUCCESS 表示成功，否则失败，此时系统不会分配相应资源。

实现文件回放的函数

函数	功能
MTANewPlayBack	分配用于回放操作的系统资源，返回分配的资源ID号。
MTAFreePalyBack	释放相关的系统回放资源。
MTAOpenFile	打开回放文件。
MTACloseFile	关闭nPlayBackID相关的回放文件。
MTASetPlayPosition	设置文件播放位置。
MTAGetPlayPosition	得到当前回放操作的播放状态级播放位置。
MTAGetVideoSize	得到回放文件中视频图像的尺寸大小。
MTAStartPaly	从当前位置处开始播放文件。
MTAPausePaly	暂停播放。
MTAGetPlayTimeLen	得到回放文件的总时间长度

MTANewPlayBack

分配用于回放操作的系统资源，返回分配的资源 ID 号。

```
int    MTANewPlayBack(LPCTSTR lpFileName);
```

参数：lpFileName：初始化回放文件名，可为 NULL。

返回：系统分配的整型回放 ID 号，-1 表示错误，调用 MTAGetLastError 得到错误代码。

MTAFreePalyBack

释放 nPlayBackId 相关的系统回放资源。

```
void   MTAFreePlayBack(int nPlayBackId);
```

参数：nPlayBackId：要释放的回放资源 ID 号。

返回：无。

MTAOpenFile

打开回放文件。

```
BOOL   MTAOpenFile(int nPlayBackID,LPCTSTR lpFileName);
```

参数：nPlayBackID：　　回放资源 ID。

**　　lpFileName：　　回放文件名。**

返回：返回 TRUE 表示成功，FALSE 表示失败，调用 MTAGetLastError 得到错误代码。

MTACloseFile

关闭 nPlayBackID 相关的回放文件。

```
BOOL   MTACloseFile(int nPlayBackID);
```

参数：nPlayBackID：　　回放资源 ID。

返回：无。

MTASetPlayPosition

设置文件播放位置。

BOOL MTASetPlayPosition(int nPlayBackID, DWORD dwPlayTime);

参数：

nPlayBackID： 回放资源 ID。

dwPlayTime： 相对于文件开始的播放位置，单位毫秒。

返回：

布尔值。返回 TRUE 表示定位成功，FALSE 表示失败，调用 MTAGetLastError 返回错误代码。

MTAGetPlayPosition

得到当前回放操作的播放状态级播放位置，推荐以 1 秒为单位周期行调用。

DWORD MTAGetPlayPosition(int nPlayBackID,BOOL &bPlaying);

参数：nPlayBackID： 回放资源 ID。

bPlaying： 输出变量，返回当前播放状态，表征是否正在播放。

返回：当前播放位置（相对于文件起始），单位毫秒。

MTAGetVideoSize

得到回放文件中视频图像的尺寸大小。

BOOL MTAGetVideoSize(int nPlayBackID,WORD &usWidth,WORD &usHeight);

参数：nPlayBackID： 回放资源 ID。

usWidth： 输出变量，返回视频图像的宽度。

usHeight： 输出变量，返回视频图像的高度。

返回：布尔值。TRUE 表示成功获取，此时长度、宽度变量有效，FALSE 表示获取失败，调用 MTAGetLastError 返回错误代码。

MTAStartPaly

从当前位置处开始播放文件。

BOOL MTAStartPlay(int nPlayBackID,DWORD dwMode);

参数：nPlayBackID: 回放资源 ID 号。

dwMode： 播放媒体类型。（可用或操作实现音视频同时播放）

PLAY_VIDEO_DATA： 播放视频流

PLAY_AUDIO_DATA： 播放音频流。

返回：布尔值。返回 TRUE 表示操作成功，FALSE 表示失败，调用 MTAGetLastError 得到错误代码。

MTAPausePaly

暂停播放。

void MTAPausePlay(int nPlayBackID);

参数：nPlayBackID: 回放资源 ID 号。

返回：无。

MTAGetPlayTimeLen

得到回放文件的总时间长度，单位毫秒

int MTAGetPlayTimeLen(int nPlayBackID);

参数：nPlayBackID：回放资源 ID。

返回：整型值，文件总时长，返回 -1 表示错误，调用 MTAGetLastError 得到错误代码。

第五章 10MOONS VC404P_401P SDK 安装说明

安装使用说明:

系统安装完成后会在安装路径下生成如下目录

安装路径：

|----------SAA7134.SYS（SAA7134 驱动）

|----------Sa7134Capture.dll(SAA7134 二次开发所需的动态库)

|----------Sa7134Capture.h(SAA7134 二次开发所需的头文件)

|----------Sa7134Capture. lib(SAA7134 二次开发所需的 lib 库)

|----------Mpeg4Codec(Microsoft Mpeg4 压缩引擎)

|----------Mix.dll(字幕叠加所需的动态库)

|----------SAA7134Demo.exe(Demo 程序可执行文件)

|---------- Server Demo(Server Demo 源程序代码)

|---------- Client Demo(Client Demo 源程序代码)

注：如果您想使用本软件的MPEG4压缩功能，请先注册微软公司提供的MPEG4压缩引擎，安装方法为：鼠标右键单击 Mpeg4Codec 目录下的 Mpg4vki.inf 文件，在弹出式菜单中选择“ 安装 ”即可。

另：其他程序不需注册，直接使用即可。