IVS-255

Real-time MPEG4 Industrial Video Server

Client Site ActiveX Control Programming Guide

(Version: 1.0.0)

IVS-255 Client Site ActiveX Control Programming Guide

Rev. 1.0.0: Mar 20, 2006

The information in this document is subject to change without prior notice in order to improve reliability, design and function and does not represent a commitment on the part of the manufacturer.

In no event will the manufacturer be liable for direct, indirect, special, incidental, or consequential damages arising out of the use or inability to use the product or documentation, even if advised of the possibility of such damages.

Revision	Date	Description
1.0.0	2006/03/20	First release

Contents

Chapter 1: Summarize

The ActiveX Controls of IVS-255 are software tools that help users to develop application programs on the client site. It consists of two files: **IVS255.ocx** and **lame_enc.dll**. After installation, these two files can be found in the **"windows/system32"** directory. **IVS255.ocx** contains three controls: **ViewerCtrl**, **MP3EncoderCtr**, and **VSLinkCtrl**. These controls are developed in Visual C++ 6.0, and can be called conveniently in the environment that supports windows **ActiveX**, such as **VC** and **VB**.

- The ActiveX control must be registered in the windows operating system before use. The following command can be used to register the control in command line:

  regsvr32 IVS255.ocx

- The control must be reregistered again if the path of the IVS255.ocx is changed or a new IVS255.ocx version is used.
- The file lame_enc.dll is an MP3 encoder dynamic link library, which is used to compress sampled audio data. This file must be put in 'windows/system32' or in the same directory as the user's application program.
- For playing video correctly, **DirectX 9.0 or later** and **DivX MPEG4 codec 5.01 or later** must be installed first, and the VGA display should be set as **RGB32**. Note: It is strongly recommended that the user should update their windows operation system from Microsoft's Windows update,

  http://windowsupdate.microsoft.com/

Chapter 2: Controls

2.1 ViewerCtrl

2.1.1 Description

ViewerCtrl works on:

- As a Real-time player of video/audio stream
- Saving these streams into specified avi file
- Playing previous saved avi file

The ViewerCtrl (supports) the following properties and methods

Properties	SavedFileName [*.avi]
	EnableAudioPlay
Methods	Open (*Source)
	Close()
	PutHeader(bVideo, *Data, Size)
	PutData(bVideo, VSerNumber, Key, *Data, Size)

2.1.2 Property

string SavedFileName

- Descriptions:
  - ✧ Specifying the **avi** file to save video/audio data received from the IVS-255.
  - ✧ If ***SavedFileName*** is empty (“”), ViewerCtrl just plays the A/V streams without saving them into a file
  - ✧ If ***SavedFileName*** is not empty and ***Open*** method is called with *Source* = “”, this ViewerCtrl will both play A/V streams and save them into a .**avi** file specified by ***SavedFileName***. The ViewerCtrl will finish saving when ***SavedFileName*** is set to “”. Then the user can give another new ***SavedFileName***, so that ViewCtrl starts to save the AV stream into this new file.
  - ✧ If ***Open*** method is called with *Source* not equal to “” (that is, ViewerCtrl is opened for playing **avi** file) the ***SavedFileName*** is clear to empty automatically

- Example:
  - ✧ VC:

Sample 1: Set *SavedFileName*

```
m_viewer_ctrl0.SetSavedFileName("MyChannel.avi");    // Save into MyChannel.avi
char AA[25];
sprintf(AA,"MyChannel1.avi");
m_viewer_ctrl1.SetSavedFileName(AA);                 // Save into MyChannel1.avi
m_viewer_ctrl2.SetSavedFileName("");                 // Stop saving
```

Sample 2: Get *SavedFileName*

```
CString AA = m_viewer_ctrl3.GetSavedFileName(); // Get current saving file name
```

Sample 3: Usage a check button to start/stop saving avi file

```
void CIVS255_DemoDlg::OnCheck5()
{
    m_SaveCheck1 = ! m_SaveCheck1;
    if(!m_SaveCheck1)
        m_viewer_ctrl0.SetSavedFileName(""); // Stop and finish current saving file
    else
    {
        char s[30];
        Get_Time_String(1,s);    // Use system time as file name
        m_viewer_ctrl0.SetSavedFileName(s);    // start new saving file
    }
}

void CIVS255_DemoDlg::Get_Time_String(int Channel, char *s)
{
    SYSTEMTIME current_systime;
    GetLocalTime(&current_systime);
    sprintf(s,"CH%d_%04d_%02d_%02d_%02d_%02d_%02d.avi",Channel
                                                       ,current_systime.wYear
                                                       ,current_systime.wMonth
                                                       ,current_systime.wDay
                                                       ,current_systime.wHour
                                                       ,current_systime.wMinute
                                                       ,current_systime.wSecond);
}
```

✧ VB:

Sample 1: Set *SavedFileName*

```
ViewerCtrl1(0).SavedFileName = "MyChannel.avi"    'Save into MyChannel.avi
ViewerCtrl1(1).SavedFileName = ""                 'Stop saving
```

Sample 2: Get *SavedFileName*

```
Text1.Text = ViewerCtrl1(0).SavedFileName    'Get current saving file name
```

Sample 3: Usage (of) a timer to (save) a new avi file every 30 (seconds)

```
Private Sub Timer3_Timer()                          'Timer3.Interval = 30000
    If Not Server_Link Then Exit Sub
    ViewerCtrl1(0).SavedFileName = ""
    ViewerCtrl1(0).SavedFileName = _
            Format(Date, "yyyymmdd") + "_" + Format(Time, "hhmmss") + ".avi"
End Sub
```

bool EnableAudioPlay

- Descriptions:
  - ✧ Specify if audio output enabled on client's speaker
  - ✧ If ***EnableAudioPlay*** is true, the ViewerCtrl will output audio, otherwise, no audio will be rendered on the client PC.
- Example:
  - ✧ VC:

Sample 1: Set enable/disable

```
m_viewer_ctrl0.SetEnableAudioPlay(true);    // Set Enable
m_viewer_ctrl1.SetEnableAudioPlay(false);    // Set Disable
```

Sample 2: Get enable/disable

```
if(m_viewer_ctrl3.GetEnableAudioPlay())       // If enabled, then …..
{
      …..
}
```

  - ✧ VB:

Sample 1: Set enable/disable

```
ViewerCtrl1(0).EnableAudioPlay(True);    // Set Enable
ViewerCtrl1(1).SetEnableAudioPlay(False);    // Set Disable
```

Sample 2: Get enable/disable

```
If ViewerCtrl1(2).EnableAudioPlay then       // If enabled, then …..
      …..
else
      …..
```

2.1.3 Method

long Open(char Source)*

- Descriptions:
  - ✧ Open a file source or a stream source
    - **Open a file source:** If a **divX** format **avi** file name is specified by *Source*, then, the ViewerCtrl starts to play this **avi** file.
    - **Open stream source:** If “” is specified by *Source*, then ViewerCtrl becomes ready for header and data input. User may call ***PutHeader*** and ***PutData*** method after this.
  - ✧ If a stream source is opened, and, ***SavedFileName*** is not empty, the received stream data will be both displayed on screen and saved into a .avi file specified by ***SavedFileName***.
- Parameters:

Parameter	Type	Description
Source	char*	**avi** file name or “” for streaming data source

- Return value: if successfully, the function returns 0. If some error occurs, it returns the error code. The error code could be:

  Value = *1→ The ‘Open’ function is called already*

  Value = *2→ Can’t open the avi file for play. (Only occurs when ‘Source’ is not empty)*

  Value = *3→ Can’t open the avi file specified by ‘SavedFileName’ for writing.*

- Example:
  - ✧ VC:

Sample 1: Open existing avi file

```
void CIVS255_DemoDlg::OnButton3() // Start or Stop Play back
{
    if (!b_IfPlayBack)
    {
        m_commondialog.ShowOpen();
        m_viewerctrl[0]->Open(m_commondialog.GetFileName());
        b_IfPlayBack = true;
        ….
    }
    else
    {
        m_viewerctrl[0]->Close();
        b_IfPlayBack = false;
        ….
    }
}
```

Sample 2: Open stream source

```
void CIVS255_DemoDlg::OnLinkButton() // Link to Server
{
    ….
    for(int i=0;i<4;i++)
        m_viewerctrl[i]->Open("");
    ….
}
```

✧ VB:

Sample 1: Open existing avi file

```
Private Sub Command5_Click()    'Start/Stop Play back
    Static AAA As Integer
    If AAA = 0 Then
        CommonDialog1.ShowOpen
        ViewerCtrl1(0).Open CommonDialog1.FileName
        AAA = 1
    Else
        ViewerCtrl1(0).Close
        AAA = 0
    End If
End Sub
```

Sample 2: Open stream source

```
Private Sub Command1_Click()
    Dim retval As Integer
    Server_Link = True
    ViewerCtrl1.Open ""
    Timer3.Enabled = True
    Counter = 10
    Timer3_Timer
    retval = VSLinkCtrl1.VS_Link
End Sub
```

long Close()

- Descriptions: Close the currently playing **avi** file or stream source
- Parameters: None.
- Return value: if successful, the function returns 0. If some error occurs, it returns the error code. The error code could be:

  Value = *1→ Open is not called yet.*

- Example:

  ✧ VC:

  Sample :

  ```
  m_viewerctrl[0]->Close();
  ```

  ✧ VB:

  Sample:

  ```
  ViewerCtrl1(0).Close
  ```

long PutHeader(bool bVideo, long Data, long Size)

- Descriptions: Put Video / Audio header information into ViewerCtrl
  - ✧ It is the VSLinkCtrl that prepares header information. When it is ready, VSLinkCtrl trigger a ***HeadStreamReady*** event, and all necessary parameters are available in this event's parameters.
- Parameters:

Parameter	Type	Description
bVideo	Boolean	True, for Video, False, for Audio
Data	Long	Address of pointer to data
Size	Long	Size of Data

- Return value: if successfully, the function returns 0. If some error occurs, it returns the error code. The error code could be:

  Value = *1→ User didn't call the 'Open' method successfully before calling PutHeader*

  Value = *2→ User called the 'Open' method with the parameter '*Source*' not equals "", it means the control is playing an avi file now.*

  Value = *3→ The size of the Data buffer is wrong.*

  Value = *4→ Can't open the DivX driver.*

  *If both **DivX** and **DirectX 9.0** are already correctly installed, this error may be caused by Screen color depth. Try to change it to RGB32 to solve this problem*

  Value = *5→ Error in writing to avi File. (Occurs only when the 'SavedFileName' property is not empty)*

  Value = *6→ Open the Waveform Output device failed. (Often because the device is occupied by another program)*

- Example:
  - ✧ VC:

Sample : Call PutHeader in HeadStreamReady event of VSLinkCtrl

```
void CIVS255_DemoDlg::OnHeadStreamReadyVslinkctrlctrl1(long Channel, BOOL bVideo, long Data, long Size)
{
    m_viewerctrl[Channel]->PutHeader( bVideo, Data, Size );
}
```

  - ✧ VB:

Sample: Call PutHeader in HeadStreamReady event of VSLinkCtrl

```
Private Sub VSLinkCtrl1_HeadStreamReady(ByVal Channel As Long, ByVal bVideo As Boolean, ByVal Data As Long, ByVal Size As Long)
    ViewerCtrl1(Channel).PutHeader bVideo, Data, Size
End Sub
```

long PutData(bool bVideo, long VSerNumber, bool Key, long Data, long Size)

- Descriptions: Put Video / Audio stream data into ViewerCtrl
  - ✧ It is VSLinkCtrl that prepares stream data. When it is ready, VSLinkCtrl triggers a ***DataStreamReady*** event, and all necessary parameters are available in this event's parameters.
- Parameters:

Parameter	Type	Description
bVideo	Boolean	True, for Video, False, for Audio
VSerNumber	Long	Serial Number of Frame
Key	Boolean	True, key frame (I-frame) False, not key frame (F-frame)
Data	Long	Address of pointer to data
Size	Long	Size of Data

- Return value: If successful, the function returns 0. If some error occurs, it returns the error code. The error code could be:

  Value = *1→ User didn't call the function 'PutHeader' successfully before calling this function or the control is closed.*

  Value = *2→ Error in writing to avi File. (Occurs only when the 'SavedFileName' property is not empty)*

  Value = *3→ Decode the Video/Audio Data Error*

- Example:
  - ✧ VC:

Sample : Call PutData in DataStreamReady event of VSLinkCtrl

```
void CIVS255_DemoDlg::OnDataStreamReadyVslinkctrlctrl1(long Channel, BOOL bVideo, long VSerNumber, BOOL Key, long Data, long Size)
{
    // TODO: Add your control notification handler code here
    m_viewerctrl[Channel]->PutData( bVideo, VSerNumber, Key, Data, Size );
}
```

  - ✧ VB:

Sample: Call PutData in DataStreamReady event of VSLinkCtrl

```
Private Sub VSLinkCtrl1_DataStreamReady(ByVal Channel As Long, ByVal bVideo As Boolean, ByVal VSerNumber As Long, ByVal Key As Boolean, ByVal Data As Long, ByVal Size As Long)
    ViewerCtrl1(Channel).PutData bVideo, VSerNumber, Key, Data, Size
End Sub
```

2.2 MP3EncoderCtrl

2.2.1 Description

This ActiveX control is used to encode a microphone input signal into MP3 frames on the client PC. The sample rate is 44.1kHz, bit rate: 128 Kbps, mono-channel. It also enumerates all possible audio input sources for the user's selection.

The MP3EncoderCtrl supports the following property, methods and event

Property	AudioInputSource
Methods	Mp3EncodeStart()
	Mp3EncodeStop()
Event	StreamReady(*Data,Size)

2.2.2 Property

long AudioInputSource

- Descriptions:
  - ✧ Specifies the audio input source on the client site. User can set it during the design phase.
  - ✧ The MP3EncoderCtrl automatically enumerates possible audio input sources.
  - ✧ In most cases, this property is set during the design phase and needs no change during run time

2.2.3 Method

long Mp3EncodeStart ()

- Descriptions:
  - ✧ After calling this method, MP3EncoderCtrl starts encoding Audio input from ***AudioInputSource*** into MP3 stream data, and, every 26 ms MP3EncoderCtrl triggers a ***StreamReady*** event until ***Mp3EncodeStop*** is called.
  - ✧ If no audio input source is available, this method returns a false and never triggers a ***StreamReady*** event.
- Parameters: None
- Return value: If successful, the function returns 0. If some error occurs, it returns the error code. The error code could be:

Value = *1*→ *Can not find the lame_enc.dll*

Value = *2*→ *Can't open audio input device*

- Example:
  - ✧ VC:

Sample : Use a check button to start/stop MP3 encoder

```
void CIVS255_DemoDlg::OnCheck9() // Audio Out
{
    m_AudioOutCheck = ! m_AudioOutCheck;
    if(!m_AudioOutCheck)
        m_mp3encoderctrl.Mp3EncodeStop();
    else
        if(b_IfLinked && (i_SelectModel > 1))
            m_mp3encoderctrl.Mp3EncodeStart();
}
```

  - ✧ VB:

Sample: Call PutData in DataStreamReady event of VSLinkCtrl

```
Private Sub Command6_Click()
    If Command6.Caption = "Start" Then
        MP3EncoderCtrl1.Mp3EncodeStart
        Command6.Caption = "Stop"
    Else
        MP3EncoderCtrl1.Mp3EncodeStop
        Command6.Caption = "Start"
    End If
End Sub
```

long Mp3EncodeStop ()

- Descriptions:
  - ✧ After calling this method, MP3EncoderCtrl stops encoding Audio input from ***AudioInputSource*** into the MP3 stream data, and, no ***StreamReady*** event is further triggered.
- Parameters: None
- Return value: Always returns 0.
- Example:
  - ✧ VC:

Sample : Use a check button to start/stop MP3 encoder

```
void CIVS255_DemoDlg::OnCheck9() // Audio Out
{
    m_AudioOutCheck = ! m_AudioOutCheck;
    if(!m_AudioOutCheck)
        m_mp3encoderctrl.Mp3EncodeStop();
    else
        if(b_IfLinked && (i_SelectModel > 1))
            m_mp3encoderctrl.Mp3EncodeStart();
}
```

✧ VB:

Sample: Call PutData in DataStreamReady event of VSLinkCtrl

```
Private Sub Command6_Click()
    If Command6.Caption = "Start" Then
        MP3EncoderCtrl1.Mp3EncodeStart
        Command6.Caption = "Stop"
    Else
        MP3EncoderCtrl1.Mp3EncodeStop
        Command6.Caption = "Start"
    End If
End Sub
```

2.2.4 Event

void StreamReady(long Data, long Size)

- Descriptions:
  - ✧ This Event is triggered every 26 ms, after **Mp3EncodeStart** method is called. The encoded MP3 frame data is put in ***Data*, and *Size* tells the number of bytes.
  - ✧ In the trigger routine, the user could use the ***SendMP3Stream*** method of VSLinkCtrl to send MP3 frame data to IVS-255.
- Parameters:

Parameter	Type	Description
Data	Long	Address of pointer to MP3 stream data
Size	Long	Size of Data

- Example:
  - ✧ VC:

Sample : Call SendMP3Stream method in StreamReady routine

```
void CIVS255_DemoDlg::OnStreamReadyMp3encoderctrlctrl1(long Data, long Size)
{
    // TODO: Add your control notification handler code here
    m_vslinkctrl.SendMP3Stream(Data,Size);
}
```

  - ✧ VB:

Sample: Call SendMP3Stream method in StreamReady routine

```
Private Sub MP3EncoderCtrl1_StreamReady(ByVal Data As Long, ByVal Size As Long)
    If Server_Link Then
        VSLinkCtrl1.SendMP3Stream Data, Size
    End If
End Sub
```

2.3 VSLinkCtrl

2.3.1 Description

The VSLinkCtrl is used to:

- Define the Server IP address and the client local ports (LocalPort and AV_LocalPort) for receiving UDP sockets from the server
- Send encoded Mp3 frame data to IVS-255 for playing.
- Receive MP4 Video stream and ADPCM Audio stream data from IVS-255 for playing on the client side.
- Monitor the status of IVS-255
- Control and Configure IVS-255
- Reset IVS-255
- Send data to RS232, COM2
- Receiving data that input to RS232, COM2 of IVS-255.

And, it supports the following properties, methods and events

Properties	IP_Address
	LocalPort
	AV_LocalPort
	ServerReady
	Video_Mode
	Video_0_Enable, Video_1_Enable, Video_2_Enable, Video_3_Enable
	Audio_0_Enable, Audio_1_Enable, Audio_2_Enable, Audio_3_Enable
	VideoFrameRate
	VideoQuantization
	VideoIPInterval
	BitRate
	RS232DataType
	MDSense
Methods	VS_Link()
	VS_unLink()
	SendMP3Stream(*Data,Size)
	ConfigurationChange()
	ResetServer()
	SetRS232SendBuffer(Index, Data)
	SendRS232(Length)
	ReadRS232Data()
Events	HeadStreamReady(Channel, bVideo, *Data, Size)

	DataStreamReady(Channel, bVideo, VSerNumber, Key, *Data, Size)
	ServerDown()
	FrameDrop(Channel)
	ReceiveRS232String(Data, Size)
	ReceiveRS232Binary(Length)
	MotionDetected(Channel)

2.3.2 Property

string IP_Address

- Description: IP Address of the IVS-255.
- Example:
  - ✧ VC:

    Sample:

    ```
    char aa[20],bb[20];
    m_IPaddresscombo.GetWindowText(aa,20);
    m_vslinkctrl.SetIP_Address(aa);     // Set IVS-255's IP address
    ```

  - ✧ VB:

    Sample:

    ```
    VSLinkCtrl1. IP_Address = "192.168.0.218"
    ```

short LocalPort

- Descriptions:
  - ✧ Define the local socket port number for receiving data from IVS-255.
  - ✧ When more than one IVS-255 is linked, it needs exactly the same number of VSLinkCtrl controls as IVS-255. In this case, each VSLinkCtrl should be assigned different ***LocalPort*** and ***AV_LocalPort*** numbers.

short AV_LocalPort

- Descriptions:
  - ✧ Define the local socket port number for receiving MP4 Video and ADPCM Audio stream data from IVS-255.
  - ✧ When more than one IVS-255 is linked, it needs exactly the same number of VSLinkCtrl controls as IVS-255. In this case, each VSLinkCtrl should be

assigned different ***LocalPort*** and ***AV_LocalPort*** numbers.

bool ServerReady;

- Descriptions:
  - ✧ An indicator that tells if IVS-255 is ready for the client to link on.
  - ✧ Read only property
  - ✧ Before IP is assigned, the ***ServerReady*** is *False*.
  - ✧ After IP is correctly assigned, and no other client links to this specified server, ***ServerReady*** is *True*.
  - ✧ If some other client has linked to the IVS-255, ***ServerReady*** becomes *False*.
  - ✧ The ***ServerReady*** keeps in *True* state, after successfully linking by calling ***VS_Link***.
- Example:
  - ✧ VC:

Sample: Check if server is ready on button click

```
void CIVS255_DemoDlg::OnCheckServerReady()
{
    char aa[20],bb[20];
    m_IPaddresscombo.GetWindowText(aa,20);
    m_vslinkctrl.SetIP_Address(aa);     // Set IVS-255's IP address
    if(m_vslinkctrl.GetServerReady())
    {
        AfxMessageBox("Server is Ready!!");
        ….
    }
    else
        AfxMessageBox("Server is Not Ready!!");
}
```

  - ✧ VB:

Sample: Check if server is ready on button click

```
Private Sub Command6_Click()
    If VSLinkCtrl1.ServerReady Then
        MsgBox "Server is ready!!"
    Else
        MsgBox "Server is NOT ready!!"
    End If
End Sub
```

long Video_Mode

- Descriptions:
  - ✧ Mode of Video Channel:

Value = 0: NTSC CIF, 320 * 240

Value = 1: NTSC Full, 640 * 480

Value = 2: PAL CIF, 352 * 288

Value = 3: PAL Full, 702 * 576

✧ Only one channel can be set to full mode. So, if some ***Video_Mode*** is set to 1 or 3, the other 3 video channels must be disabled.

✧ Changing ***Video_Mode*** after linking won't result in an immediate response until the **ConfigurationChange** method is successfully called.

♦ Example:

✧ VC:

Sample: Change video mode on button click

```
void CIVS255_DemoDlg::OnLargeSmallCH1()
{
	// TODO: Add your control notification handler code here
	SaveFileAllStop();
	if (!b_IfLarged)
	{
		Full_Display(0);
	}
	else
		Cif_Display(0);
}
void CIVS255_DemoDlg::Cif_Display(int Ch)
{
	b_IfLarged = false;
	m_viewerctrl[Ch]->SetSavedFileName("");
	m_viewerctrl[Ch]->Close();
	….
	if(b_IfNTSC)
		m_vslinkctrl.SetVideo_Mode(0);	// CIF NTSC
	else
		m_vslinkctrl.SetVideo_Mode(2);	// CIF PAL
	for(int i=0;i<4;i++)
		m_viewerctrl[i]->Open("");
	m_vslinkctrl.ConfigurationChange();
	….
}

void CIVS255_DemoDlg::Full_Display(int Ch)
{
	int i;
	b_IfLarged = true;
	i_LargeChannel =Ch;
	for(i = 0 ; i< 4; i++)
	{
		m_viewerctrl[i]->SetSavedFileName("");
		m_viewerctrl[i]->Close();
		….
	}
	….

	if(b_IfNTSC)
		m_vslinkctrl.SetVideo_Mode(1);	// Full NTSC
	else
		m_vslinkctrl.SetVideo_Mode(3);	// Full PAL
	m_viewerctrl[Ch]->Open("");
```

```
        m_vslinkctrl.ConfigurationChange();
        ….
    }
```

✧ VB:

Sample: Change to full video mode on button click

```
Private Sub Command2_Click()
    ViewerCtrl1(0).Width = ViewerCtrl1(0).Width * 2
    ViewerCtrl1(0).Height = ViewerCtrl1(0).Height * 2
    Me.Width = Me.Width * 2 - 270
    Me.Height = Me.Height * 2 – 1120
    ….
    VSLinkCtrl1.Video_Mode = 3          'Full PAL
    VSLinkCtrl1.ConfigurationChange
    Command3.Enabled = True
    Command2.Enabled = False
End Sub
```

long MDSense

- Descriptions:
  - ✧ Sensitivity of motion detection: value 0 ~ 63

    Value = 0: Extremely sensitive

    Value = 40: Default

    Value = 63: No MD
  - ✧ Changing ***MDSense*** after linking won't result in an immediate response until the **ConfigurationChange** method is successfully called.
- Example:
  - ✧ VC:

Sample1: Set MD sensitivity value

```
m_vslinkctrl.SetMDSense(20); // Set MD sensitivity value to be 20
```

Sample 2: Get current MD sensitivity value

```
if (m_vslinkctrl.GetMDSense () == 20)
{
    ….
}
```

✧ VB:

Sample1: Set MD sensitivity value

```
VSlinkCtrl1.MDSense = 20
```

Sample 2: Get current MD sensitivity value

```
If VSlinkCtrl1.MDSense = 20 then
    ….
End if
```

bool Video_0_Enable, Video_1_Enable, Video_2_Enable, Video_3_Enable

- Descriptions:
  - ✧ Flag of Video Channel Enable.
  - ✧ *True* for channel enabled, *False* for disabled. Default value = *False*
  - ✧ If some ***Video_Mode*** is set to 1 or 3, other Video channels must be disabled
  - ✧ Changing ***Video_#_Enable*** after linking won't result in an immediate response until the **ConfigurationChange** method is successfully called.
  - ✧
- Example:
  - ✧ VC:

Sample1: Set video channel (ON/OFF)

```
m_vslinkctrl.SetVideo_0_Enable(true);
m_vslinkctrl.SetVideo_1_Enable(false);
```

Sample 2: Get video channel ON/OFF status

```
if (m_vslinkctrl.GetVideo_2_Enable())
{
    ….
}
```

  - ✧ VB:

Sample1: Set video channel (ON/OFF)

```
VSlinkCtrl1.Video_2_Enable =True
```

Sample 2: Get video channel ON/OFF status

```
If VSlinkCtrl1.Video_3_Enable then
    ….
End if
```

bool Audio_0_Enable, Audio_1_Enable, Audio_2_Enable, Audio_3_Enable

- Descriptions:
  - ✧ Flag of Audio channel enable.
  - ✧ *True* for channel enabled, *False* for disabled. Default value = *False*
  - ✧ IVS-255 has 4 CH audio input capability
  - ✧ Changing ***Audio_#_Enable*** after linking won't result in an immediate response until the **ConfigurationChange** method is successfully called.
- Example:
  - ✧ VC:

Sample1: Set audio channel (ON/OFF)

```
m_vslinkctrl.SetAudio_0_Enable(true);
m_vslinkctrl.SetAudio_1_Enable(false);
```

Sample 2: Get audio channel ON/OFF status

```
if (m_vslinkctrl.GetAudio_2_Enable())
{
    ….
}
```

✧ VB:

Sample1: Set audio channel (ON/OFF)

```
VSlinkCtrl1.Audio_2_Enable =True
```

Sample 2: Get audio channel ON/OFF status

```
If VSlinkCtrl1.Audio_3_Enable then
    ….
End if
```

long VideoFrameRate

♦ Descriptions: Video frame rate

Value = 0: 30 (NTSC) , 25 (PAL) frame/sec, default

Value = 1: 15 (NTSC) , 12.5 (PAL) frame/sec

Value = 2: 7.5 (NTSC) , 6.25 (PAL) frame/sec

Value = 3: 3.75 (NTSC) , 3.125 (PAL) frame/sec

Value = 4: 1.875 (NTSC) , 1.5625 (PAL) frame/sec

Value = 5: 0.9375 (NTSC) , 0.78125 (PAL) frame/sec

✧ Before linking, this property could be written to set a suitable video frame rate

✧ Changing ***VideoFrameRate*** after linking won't result in an immediate response until the **ConfigurationChange** method is successfully called.

♦ Example:

✧ VC:

Sample1: Set video frame rate

```
m_vslinkctrl.SetVideoFrameRate(0); // Set to be 30 (NTSC) , 25 (PAL) frame/sec,
```

Sample 2: Get current video frame rate

```
if (m_vslinkctrl.GetVideoFrameRate() == 0)
{
    ….
}
```

✧ VB:

Sample1: Set video frame rate

```
VSlinkCtrl1.VideoFrameRate =0
```

Sample 2: Get current video frame rate

```
If VSlinkCtrl1.VideoFrameRate = 0 then
    ….
End if
```

long VideoQuantization

- Descriptions:
  - ✧ Video frame rate, Value: 2~31, Default value = 5
  - ✧ Before linking, this property could be written to set a suitable video quantization value
  - ✧ Changing ***VideoQuantization*** after linking won't result in an immediate response until the **ConfigurationChange** method is successfully called.
- Example:
  - ✧ VC:

Sample1: Set video quantization

```
m_vslinkctrl.SetVideoQuantization(7); // Set Video quantization value to be 7
```

Sample 2: Get current video quantization

```
if (m_vslinkctrl.GetVideoQuantization() == 9)
{
    ….
}
```

  - ✧ VB:

Sample1: Set video quantization

```
VSlinkCtrl1.VideoQuantization = 5
```

Sample 2: Get current video quantization

```
If VSlinkCtrl1.VideoQuantization = 5 then
    ….
End if
```

long VideoIPInterval

- Descriptions:
  - ✧ To set Video IP interval, I: I frame (Key frame), P: P frame

Value = 0: IP interval = 2

Value = 1: IP interval = 4

Value = 2: IP interval = 8

Value = 3: IP interval = 16

Value = 4: IP interval = 32, default

Value = 5: IP interval = 64

Value = 6: IP interval = 128

Value = 7: IP interval = 256

- ✧ Before linking, this property can be written to set a suitable video IP interval
- ✧ Changing ***VideoIPInterval*** after linking won't result in an immediate response until the **ConfigurationChange** method is successfully called.

♦ Example:

✧ VC:

Sample1: Set video IP interval

```
m_vslinkctrl.SetVideoIPInterval(3); // Set to IP interval to be 8,
```

Sample 2: Get current video IP interval

```
if (m_vslinkctrl.GetVideoIPInterval() == 3)
{
    ….
}
```

✧ VB:

Sample1: Set video IP interval

```
VSlinkCtrl1.VideoIPInterval=5
```

Sample 2: Get current video IP interval

```
If VSlinkCtrl1.VideoIPInterval= 5 then
    ….
End if
```

long BitRate

♦ Descriptions:

- ✧ This property is a read only property that shows current AV streaming bit rate from server. It is only effective in run time.
- ✧ This value is a software calculation by VSLinkCtrl according to received package bytes during the most recent second.

♦ Example:

✧ VC:

Sample: Get current bit rate

```
char a[20];
sprintf(a,"%.3f",((float)m_vslinkctrl.GetBitRate())/8000);
((CEdit*)GetDlgItem(IDC_EDIT1))->SetWindowText(a);
```

✧ VB:

Sample: Get current bit rate

```
Private Sub Timer2_Timer()
    Text1.Text = VSLinkCtrl1.BitRate / 8000
End Sub
```

long RS232DataType

- Descriptions:
  - ✧ This property is to tell which kind of data will be received from RS232 port of IVS-255. There are 2 kind of data type, 0 for binary or 1 for string.
  - ✧ If **RS232DataType** is set to 0: Received data is in binary format, and, the event **ReceiveRS232Binary** will be fired.
  - ✧ If **RS232DataType** is set to 1: Received data is in string format, and, the event **ReceiveRS232String** will be fired.
- Example:
  - ✧ VC:

Sample: Set as binary data

```
void CRS232DemoDlg::OnRadio_Binary()
{
    m_vslink.SetRS232DataType(0); // 0: Received data is in binary format
}
```

Sample: Set as binary data

```
void CRS232DemoDlg::OnRadio_String()
{
    m_vslink.SetRS232DataType(1); // 1: Received data is in string format
}
```

  - ✧ VB:

Sample: Set as binary data

```
Private Sub OptionBinary _Click()
    VSLinkCtrl1.RS232DataType = 0
End Sub
```

Sample: Set as binary data

```
Private Sub OptionString _Click()
    VSLinkCtrl1.RS232DataType = 1
End Sub
```

2.3.3 Method

long VS_Link()

- Descriptions:
  - ✧ Used to Link to IVS-255.
  - ✧ Before ***VS_Link*** is called, the following properties should be set correctly for proper operation.

IP_Address (Can't be modified after linking)

LocalPort (Can't be modified after linking)

AV_LocalPort (Can't be modified after linking)

Video_Mode

Video_#_Enable

Audio_#_Enable

VideoFrameRate

VideoQuantization

VideoIPInterval

- ✧ Though some operation parameters can be changed on the fly, the potential cost is lost frames.
- ✧ After successfully calling ***VS_Link***, the IVS-255 specified by ***IP_Address*** is reserved by this client. No other client can link to it until **VS_unLink** or ***ResetServer*** is called or the IVS-255 reboots.
- ✧ All methods can function correctly only after ***VS_Link*** is successfully called
- ✧ No events occur before ***VS_Link*** is successfully called.
- ✧ After **ResetServer**, ***VS_Link*** must be called again to gain control of the IVS-255.

♦ Parameters: None

♦ Return value: If successful, the function returns 0. If some error occurs, it returns the error code. The error code could be:

Value = *1→ The Client has already Linked to the IVS-255.*

Value = *2→ Some properties are not set correctly: the 'IP_Address', 'LocalPort' and 'AVLocalPort' Must be set before Link to the IVS-255.*

Value = *3→ Fail on Creating 'LocalPort' Socket to receive the response data from IVS-255*

Value = *4→ Fail on Creating 'AV_LocalPort' Socket to receive video/audio data.*

Value = *5→ Time out on receiving the response packet from the IVS-255.*

Value = *6→ The response packet's format is error*

Value = *250→ Wrong VideoIPInterval*

Value = *251→ Wrong Quantization*

Value = *252→ Wrong FrameRate*

Value = *253→ Wrong VideoMode*

Value = *254→ The IVS-255 is already linked by other client*

♦ Example:

✧ VC:

Sample: Link to IVS-255

```
long ret = m_vslinkctrl.VS_Link();
if (ret ==0)            // link successfully
{
    ....
}
```

```
else                    // link failed
{
    ….
}
```

✧ VB:

Sample: Link to IVS-255

```
If VSLinkCtrl1.VS_Link = 0 then        ‘link successful
    ….
Else                                   ‘link failed
    ….
End if
```

long VS_unLink()

- Descriptions:
  - ✧ To unlink IVS-255.
  - ✧ After ***VS_unLink***, the specified IVS-255 becomes free. Any client can link to it.
- Parameters: None
- Return value: The return value is always 0
- Example:
  - ✧ VC:

Sample: unlink IVS-255 on button click

```
void CIVS255_DemoDlg::OnButton2()
{
    m_mp3encoderctrl.Mp3EncodeStop();
    m_vslinkctrl.VS_unLink();              // unlink Server
    ….
}
```

  - ✧ VB:

Sample: unlink IVS-255 on button click

```
Private Sub Command4_Click()
    VSLinkCtrl1.VS_unlink          ‘unlink server
    ViewerCtrl1(0).Close
    MP3EncoderCtrl1.Mp3EncodeStop
End Sub
```

long SendMP3Stream(long Data, long Size)

- Descriptions:
  - ✧ Send encoded MP3 stream data to IVS-255
  - ✧ When MP3EncoderCtrl fires a ***StreamReady*** event, the user can call this function to send the encoded MP3 stream to IVS-255.
  - ✧ The sent MP3 stream data will be played by IVS-255. The data that is sent first

is played first.

- Parameters

Parameter	Type	Description
Data	Long	Address of Pointer to MP3 stream data
Size	Long	Size of Data

- Return value: If successful, the function returns 0. If some error occurs, it returns the error code. The error code could be:

  Value = *1→ The Client has not linked to the IVS-255 yet*

  Value = *2→ The size of the data is not 417 or 418: the size of the MP3 frame must be 417 or 418 bytes.*

- Example:

  ✧ VC:

  Sample: Call SendMP3Stream in MP3EncoderCtrl's *StreamReady* event routine

```
void CIVS255_DemoDlg::OnStreamReadyMp3encoderctrlctrl1(long Data, long Size)
{
    m_vslinkctrl.SendMP3Stream(Data,Size);
}
```

  ✧ VB:

  Sample: Call SendMP3Stream in MP3EncoderCtrl's *StreamReady* event routine

```
Private Sub MP3EncoderCtrl1_StreamReady(ByVal Data As Long, ByVal Size As Long)
    VSLinkCtrl1.SendMP3Stream Data, Size
End Sub
```

long ResetServer()

- Descriptions:
  - ✧ Call this method to reset IVS-255.
  - ✧ Once called,
    - **The server will be reset and, after completing reboot, becomes free for all clients to link to it.**
    - *ServerReady* **becomes false, and no further events will be fired.**
    - **All methods are functionless until** *VS_Link* **is called again to link to the server.**
- Parameters: None
- Return value: The return value is always 0
- Example:

  ✧ VC:

  Sample: Reset IVS-255 on button click

```
void CIVS255_DemoDlg::OnButton3()
{
    m_mp3encoderctrl.Mp3EncodeStop();
    m_vslinkctrl.ResetServer();          // reset Server
    ….
}
```

✧ VB:

Sample: Reset IVS-255 on button click

```
Private Sub Command4_Click()
    VSLinkCtrl1.ResetServer          'reset server
    ViewerCtrl1(0).Close
    MP3EncoderCtrl1.Mp3EncodeStop
End Sub
```

long ConfigurationChange()

- (Description): Call this method to (change) properties after (a) link.
  - ✧ After (linking to a) server, (the) user can change the following properties

    Video_Mode

    Video_#_Enable

    Audio_#_Enable

    VideoFrameRate

    VideoQuantization

    VideoIPInterval
  - ✧ However, the change won't (occur) until ***ConfigurationChange*** is called.
  - ✧ This method is used to prevent (frequent configuration changes). (After user changes,) call this method to (modify?) all (canned?) properties after (the) link.
- Parameters: None
- Return value: (If successful), this method returns (a) 0. If some error occurs, it returns the error code. The error code could be:

  Value = *2→ Can't receive the response packet from the IVS-255.*

  Value = *3→ the response packet's Format is wrong*
- Example:
  - ✧ VC:

    Sample:

```
void CIVS255_DemoDlg::OnButton3()
{
    m_vslinkctrl.SetVideo_Mode(1);
    m_vslinkctrl.SetVideo_0_Enable(true);
    m_vslinkctrl.SetVideo_1_Enable(false);
    m_vslinkctrl.SetVideo_2_Enable(false);
    m_vslinkctrl.SetVideo_3_Enable(false);
    m_vslinkctrl.ResetServer();          // reset Server
    ….
}
```

  - ✧ VB:

    Sample:

```
Private Sub Command2_Click()
    ViewerCtrl1(0).Width = ViewerCtrl1(0).Width * 2
    ViewerCtrl1(0).Height = ViewerCtrl1(0).Height * 2
```

```
        Me.Width = Me.Width * 2 - 270
        Me.Height = Me.Height * 2 - 1120
        VSLinkCtrl1.Video_Mode = 3
        VSLinkCtrl1.ConfigurationChange
        ….
End Sub
```

void SetRS232SendBuffer(short Index, short Data);

- Descriptions:
  - ✧ This method is to work with **SendRS232** method to perform sending data to RS232 of IVS-255, the COM2. **SetRS232SendBuffer** is called first, to prepare data sent.
  - ✧ The size of send buffer is 256, so, the Index must not exceed 255.
  - ✧ Please note that the Data is in short type, however, only the low byte is effective. That means every byte need call **SetRS232SendBuffer** one time.
- Parameters

Parameter	Type	Description
Index	short	Index of send buffer, 0 ~ 255
Data	short	Value to be set into send buffer, valid value:0 ~ 255

Return value: void

- Example:
  - ✧ VC:

Sample: send out Binary data

```
void CRS232DemoDlg::OnSendBinary()
{
    // TODO: Add your control notification handler code here
    unsigned int i;
    char    *stopstring;
    for(i=0;i<m_binaryData.GetLength()/2;i++)
        m_vslink.SetRS232SendBuffer( i, strtoul(m_binaryData.Mid(2*i,2),&stopstring,16));
    m_vslink.SendRS232(i);
}
```

Sample: send out String data

```
void CRS232DemoDlg::OnSendString()
{
    unsigned int i;
    for(i=0;i<m_StringData.GetLength();i++)
    {
        m_vslink.SetRS232SendBuffer( i , (short)m_StringData.GetAt(i));
    }
    m_vslink.SendRS232(i);
}
```

  - ✧ VB:

Sample: send out Binary data

```
Private Sub Command1_Click()
    Dim i As Integer
    For i = 0 To 9
        VSLinkCtrl1.SetRS232SendBuffer i * 2, i + 1
        VSLinkCtrl1.SetRS232SendBuffer 1 * 2 + 1, Asc("1") + i
    Next i
    VSLinkCtrl1.SendRS232 2*i
End Sub
```

Sample: send out String data

```
Private Sub Command4_Click()
    Dim i As Integer
    For i = 0 To Len(Text1.Text) - 1
        VSLinkCtrl1.SetRS232SendBuffer i, Asc(Right(Left(Text1.Text, i + 1), 1))
    Next i
    VSLinkCtrl1.SendRS232 i
End Sub
```

long SendRS232(Length)

- Descriptions:
  - ✧ To send data to RS232 of IVS-255, the COM2.
  - ✧ Before executing this method, call "**SetRS232SendBuffer**" method to set data into send buffer. If "**SetRS232SendBuffer**" is not called, the **SendRS232** will send un-expected data to RS232.
- Parameters

Parameter	Type	Description
Length	Long	Size of Data, at most 255.

  Return value: if succeed, it return number of byte sent
  If failed, it returns –1.
- Example: refer to **SetRS232SendBuffer** method

short ReadRS232Data()

- Descriptions:
  - ✧ To read data received from RS232 of IVS-255, the COM2.
  - ✧ When **RS232DataType** is set to "0", the received data will be stored in the receiving buffer. And, this function is used to read it out. Each call gets one byte. (First come, first read) Once read, the buffer becomes empty.
  - ✧ This method is usually called after **ReceiveRS232Binary** event.
- Parameters: void
- Return value: -1: Error, read buffer is empty
  0~255: received data
- Example:
  - ✧ VC:

Sample:

```
void CRS232DemoDlg::OnReceiveRS232BinaryVslinkctrlctrl1(long Length)
{
    // TODO: Add your control notification handler code here
    char ss[1024]={0};
    for(int i = 0; i < Length; i++)
        sprintf(ss,"%s%02X",ss,m_vslink.ReadRS232Data());
```

```
        m_list1.AddString(ss);
        m_list1.SetCurSel(m_list1.GetCount()-1);
    }
```

✧ VB:

Sample:

```
Private Sub VSLinkCtrl1_ReceiveRS232Binary(ByVal Length As Long)
    Dim i As Long, s As String, data As Integer
    For i = 0 To Length - 1
        data = VSLinkCtrl1.ReadRS232Data
        If Len(Hex(data)) = 1 Then
            s = s + "0" + Hex(data)
        Else
            s = s + Hex(data)
        End If
    Next i
    List1.AddItem s
    List1.ListIndex = List1.ListCount - 1
End Sub
```

2.3.4 Event

HeadStreamReady(long Channel, bool bVideo, long Data, long Size)

- (Description): The event is fired when stream header (information) arrives from IVS-255.
- Parameters:

Parameter	Type	(Description)
Channel	Long	Channel number = 0,1,2 or 3
BVideo	Boolean	True, for Video, False, for Audio
Data	Long	Address of pointer to data
Size	Long	Size of Data

- Example:

✧ VC:

Sample : Call PutHeader in HeadStreamReady event

```
void CIVS255_DemoDlg::OnHeadStreamReadyVslinkctrlctrl1(long Channel, BOOL bVideo, long Data, long Size)
{
    m_viewerctrl[Channel]->PutHeader( bVideo, Data, Size );
}
```

✧ VB:

Sample: Call PutHeader in HeadStreamReady event

```
Private Sub VSLinkCtrl1_HeadStreamReady(ByVal Channel As Long, ByVal bVideo As Boolean, ByVal Data As Long, ByVal Size As Long)
    ViewerCtrl1(Channel).PutHeader bVideo, Data, Size
End Sub
```

DataStreamReady(long Channel, bool bVideo, long VSerNumber, bool Key, long Data, long Size)

- (Description): The event is fired when stream data arrives from IVS-255.
- Parameters:

Parameter	Type	(Description)
Channel	Long	Channel number = 0,1,2 or 3
BVideo	Boolean	True, for Video, False, for Audio
VSerNumber	Long	Serial Number of Frame
Key	Boolean	True, key frame (I-frame) False, not key frame (F-frame)
Data	Long	Address of pointer to data
Size	Long	Size of Data

- Example:
  - ✧ VC:

Sample : Call PutData in DataStreamReady event

```
void CIVS255_DemoDlg::OnDataStreamReadyVslinkctrlctrl1(long Channel, BOOL bVideo, long VSerNumber, BOOL Key, long Data, long Size)
{
    // TODO: Add your control notification handler code here
    m_viewerctrl[Channel]->PutData( bVideo, VSerNumber, Key, Data, Size );
}
```

  - ✧ VB:

Sample: Call PutData in DataStreamReady event

```
Private Sub VSLinkCtrl1_DataStreamReady(ByVal Channel As Long, ByVal bVideo As Boolean, ByVal VSerNumber As Long, ByVal Key As Boolean, ByVal Data As Long, ByVal Size As Long)
    ViewerCtrl1(Channel).PutData bVideo, VSerNumber, Key, Data, Size
End Sub
```

ServerDown()

- Descriptions:
  - ✧ Fired when connection to IVS-255 is lost.
  - ✧ This event fires only after ***VS_Link*** is successfully called.
  - ✧ When it is fired, it means the IVS-255 is not functioning. This could result from:
    - **Ethernet disconnected or was unstable**
    - **Server Crash**
  - ✧ Once fired, the linking between server and client is broken already and the user needs to call **VS_Link** again to get linked after fixing the original problem that caused ***ServerDown***.

- Parameters: None
- Example:
  - ✧ VC:

Sample :

```
void CIVS255_DemoDlg::OnServerDownVslinkctrlctrl1()
{
    OnButton2(); // Button2 --> Reset
    AfxMessageBox("Server Down! Press 'Link' again after server reboot\n");
}
```

  - ✧ VB:

Sample:

```
Private Sub VSLinkCtrl1_ServerDown()
    Server_Link = False
    MP3EncoderCtrl1.Mp3EncodeStop
    ViewerCtrl1(0).SavedFileName = ""
    ViewerCtrl1(0).Close
    MsgBox "Server is down!!"
End Sub
```

FrameDrop(long Channel)

- Descriptions:
  - ✧ Fired when at least one frame is dropped
  - ✧ This event fires only after ***VS_Link*** has successfully called.
  - ✧ ***FrameDrop*** may be caused by any of the following reasons:
    - **The network quality or bandwidth is not (good) enough**
    - **An encoding error from the server. In this case, it is a good idea to call ResetServer() method.**
  - ✧ If ***FrameDrop*** is frequently fired, try setting a lower frame rate or higher Quantization value.
  - ✧ This event is triggered if a key frame(I frame) is lost. If the lost frame is not a key frame, ie P frame, this event will also be triggered, but it will not be triggered again before the next key frame arrives.
- Parameters:

Parameter	Type	Description
Channel	Long	Channel number = 0,1,2 or 3

- Example:
  - ✧ VC:

Sample :

```
void CIVS255_DemoDlg:: OnFrameDropVslinkctrlctrl1 ()
{
    i_FrameDropCounter ++;
}
```

  - ✧ VB:

Sample:

```
Private Sub VSLinkCtrl1_FrameDrop(ByVal Channel As Long)
    Text1.Text = Clng(Text1.Text) + 1
End Sub
```

ReceiveRS232Binary (long Size)

- Descriptions:
  - ✧ The IVS-255 checks RS232, COM2, every 100ms. If any input data, IVS-255 sends it to Client, then either **ReceiveRS232Binary** or **ReceiveRS232String** event will be fired.
  - ✧ If the **RS232DataType** property is set “0”, the received data will be stored into receiving buffer, and then **ReceiveRS232Binary** be fired.
  - ✧ To read the received RS232 data, use **ReadRS232Data** method.
- Parameters:

Parameter	Type	Description
Size	long	Size of string data

- Example:
  - ✧ VC:

Sample :

```
void CRS232DemoDlg::OnReceiveRS232BinaryVslinkctrlctrl1(long Length)
{
    // TODO: Add your control notification handler code here
    char ss[1024]={0};
    for(int i = 0; i < Length; i++)
        sprintf(ss,"%s%02X",ss,m_vslink.ReadRS232Data());
    m_list1.AddString(ss);
    m_list1.SetCurSel(m_list1.GetCount()-1);
}
```

  - ✧ VB:

Sample:

```
Private Sub VSLinkCtrl1_ReceiveRS232Binary(ByVal Length As Long)
    Dim i As Long, s As String, data As Integer
    For i = 0 To Length - 1
        data = VSLinkCtrl1.ReadRS232Data
        If Len(Hex(data)) = 1 Then
            s = s + "0" + Hex(data)
        Else
            s = s + Hex(data)
        End If
    Next i
    List1.AddItem s
    List1.ListIndex = List1.ListCount - 1
End Sub
```

ReceiveRS232String (LPCTSTR Data, long Size)

- Descriptions:
  - ✧ The IVS-255 checks RS232, COM2, every 100ms. If any input data, IVS-255 sends it to Client, then either **ReceiveRS232Binary** or **ReceiveRS232String** event will be fired.
  - ✧ If the **RS232DataType** property is set “1”, **ReceiveRS232String** event will be fired.
- Parameters:

Parameter	Type	Description
Data	LPCTSTR	The received string data
Size	long	Size of string data, in byte.

- Example:
  - ✧ VC:

Sample :

```
void CRS232DemoDlg::OnReceiveRS232StringVslinkctrlctrl1(LPCTSTR Data, long Size)
{
    // TODO: Add your control notification handler code here
    m_list1.AddString(Data);
    m_list1.SetCurSel(m_list1.GetCount()-1);
}
```

  - ✧ VB:

Sample:

```
Private Sub VSLinkCtrl1_ReceiveRS232String(ByVal data As String, ByVal Size As Long)
    List1.AddItem data
    List1.ListIndex = List1.ListCount - 1
End Sub
```

MotionDetected(Channel)

- Descriptions:
  - ✧ Fired when motion detected on the specific ***Channel***
  - ✧ This event fires only after ***VS_Link*** has successfully called.
  - ✧ If ***MotionDetected*** is frequently fired, try setting a higher ***MDSense*** value.
- Parameters:

Parameter	Type	Description
Channel	Long	Channel number = 0,1,2 or 3

- Example:
  - ✧ VC:

Sample :

```
void CIVS255_DemoDlg:: OnMotionDetectedVslinkctrlctrl1 ()
```

```
{
    i_MotionDetectedCounter ++;
}
```

✧ VB:

Sample:

```
Private Sub VSLinkCtrl1_MotionDetected(ByVal Channel As Long)
    Text1.Text = Clng(Text1.Text) + 1
End Sub
```