



**ACTi SDK-10000**  
**ActiveX Control Edition**  
**v1.2.93**

# **Programming Guide**



[www.acti.com](http://www.acti.com)



## **ACTi SDK-10000**

This document is copyrighted, 2003 - 20012 by ACTi Corporation. All rights are reserved. ACTi Corporation reserves the right to make improvements to the products described in this manual at any time without notice.

No part of this manual may be reproduced, copied, translated or transmitted in any form or by any means without the prior written permission of the original manufacturer. Information provided in this manual is intended to be accurate and reliable. However, the original manufacturer assumes no responsibility for its use, or for any infringements upon the rights of third parties that may result from its use.

All other product names or trademarks are properties of their respective owners.

V1.2 Edition Aug, 2012

# Table of Contents

<b>1</b>	<b>OVERVIEW</b>	<b>1-1</b>
	Introduction .....	1-1
	SDK Function Groups	1-1
	Architecture	1-2
	Application Type	1-3
	Topics      1-4	
	What's New?	1-5
	Register ActiveX Control .....	1-6
	Runtime DLL Files \${SDK DIR}\Bin	1-6
	Sample Codes \${SDK DIR}\Samples .....	1-7
	Preview Sample Program	1-8
	Preview HTML Sample Code: \${SDK DIR}\Samples\HTML\Preview.htm ....	1-8
	Record Sample Program	1-9
	Record HTML Sample Code: \${SDK DIR}\Samples\HTML\Record.htm .....	1-9
	Playback Sample Program	1-10
	Playback HTML Sample Code: {SDK DIR}\Samples\HTML\Playback.htm ..	1-10
	MD Sample Program	1-1
	MD HTML Sample Code: {SDK DIR}\Samples\HTML\Motion.htm .....	1-1
	PTZ Sample Program	1-2
	PTZ HTML Sample Code: \${SDK DIR}\Samples\HTML\PTZ.htm .....	1-2
	PrivacyMask Sample Program	1-3
	PrivacyMask HTML Sample Code: \${SDK	
	DIR}\Samples\HTML\PrivacyMask.html .....	1-3
	Asynchronous Preview Sample Program	1-4
	Asynchronous Preview HTML Sample Code: \${SDK DIR}\Samples\HTML\	
	Asynchronous Preview.htm .....	1-4
<b>2</b>	<b>CONFIGURE DEVICE</b>	<b>2-1</b>
	How to configure with video server .....	2-1
	System Information	2-1
	HTML Sample: .....	2-1
	System Property	2-2
	HTML Sample: .....	2-2
	Video Color Adjustments .....	2-3
	Hue, Brightness, Contrast Setting	2-3

HTML Sample: .....	2-3
Video Setting Configuration.....	2-5
Setup Resolution, Frame Rate, Bit Rate .....	2-5
HTML Sample: .....	2-5
Set the video property .....	2-5
Save and Reboot .....	2-6
Execute Save and Reboot Command .....	2-6
HTML Sample: .....	2-6
<b>3 PREVIEW / RECORD / PLAYBACK / PTZ .....</b>	<b>3-1</b>
Preview / Record Architecture.....	3-1
Connect to IP devices .....	3-1
HTML Sample: \${SDK DIR}\Samples\HTML\Preview.....	3-1
Preview(Unicast/Multicast/CaptureCard) .....	3-3
Preview with Unicast Mode .....	3-3
HTML Sample: \${SDK DIR}\Samples\HTML\Preview .....	3-3
Preview with Multicast Mode .....	3-5
HTML Sample: \${SDK DIR}\Samples\HTML\Preview .....	3-5
Preview with CaptureCard Mode .....	3-7
HTML Sample: \${SDK DIR}\Samples\HTML\Preview .....	3-7
Preview with Audio .....	3-9
HTML Sample: \${SDK DIR}\Samples\HTML\Preview .....	3-9
Preview with 2-way audio .....	3-10
HTML Sample: \${SDK DIR}\Samples\HTML\Preview .....	3-10
Preview with I-Frame Decoding only .....	3-12
HTML Sample: \${SDK DIR}\Samples\HTML\Preview .....	3-12
Record 3-13 .....	
Record with unicast mode .....	3-13
Background record with multicast mode .....	3-13
HTML Sample: \${SDK DIR}\Samples\HTML\Record .....	3-13
Record to AVI file .....	3-14
HTML Sample: \${SDK DIR}\Samples\HTML\Record .....	3-14
Alarm Recording with DI event .....	3-15
HTML Sample: \${SDK DIR}\Samples\HTML\Motion .....	3-15
Playback.....	3-16
HTML Sample: \${SDK DIR}\Samples\HTML\Playback .....	3-16
PTZ(PAN/TILT/ZOOM).....	3-17
HTML Sample: \${SDK DIR}\Samples\HTML\PTZ.....	3-17

<b>4</b>	<b>EVENT HANDLING</b>	<b>4-1</b>
	Digital I/O Architecture .....	4-1
	Receives Digital Input Event	4-1
	HTML Sample: \${SDK DIR}\Samples\HTML\Motion.....	4-1
	Send Digital Output	4-1
	HTML Sample: \${SDK DIR}\Samples\HTML\Motion.....	4-1
	Motion Detection Event Handling .....	4-2
	Sets Motion Detection parameters	4-2
	HTML Sample: \${SDK DIR}\Samples\HTML\Motion.....	4-2
	Gets Motion Detection Settings	4-2
	HTML Sample: \${SDK DIR}\Samples\HTML\Motion.....	4-2
	Receives Motion Detection Trigger Event	4-3
	HTML Sample: .....	4-3
	Status Callback – video lost, recover, disconnect event.....	4-3
	HTML Sample: \${SDK DIR}\Samples\HTML\Preview.....	4-3
<b>5</b>	<b>IP QUAD VIDEO SERVER INTEGRATION</b>	<b>5-1</b>
	IP Quad Architecture .....	5-1
	IP Quad URL Commands	5-2
<b>6</b>	<b>ADVANCED TOPICS</b>	<b>6-1</b>
	Preview Functions.....	6-1
	Preview with Full Screen	6-1
	HTML Sample: \${SDK DIR}\Samples\HTML\Preview.....	6-1
	Preview with GDI or DirectDraw render	6-2
	HTML Sample: \${SDK DIR}\Samples\HTML\Preview.....	6-2
	Preview with PCI-5100, Xvid and FFMPEG decoder	6-3
	HTML Sample: \${SDK DIR}\Samples\HTML\Preview.....	6-3
	Preview with Intel IPP Codec.	6-3
	Connect Functions .....	6-4
	Asynchronize connection	6-4
	HTML Sample: \${SDK DIR}\Samples\HTML\Asynchronous_Preview .....	6-4
<b>7</b>	<b>NEW SUPPORT</b>	<b>7-1</b>
	H.264 and Motion-JPEG .....	7-1
	Preview with H.264 or Motion-JPEG	7-1
	HTML Sample:   \${SDK DIR}\Samples\HTML\Preview .....	7-1
	Dual stream.....	7-3
	Preview with dual stream device	7-3
	HTML Sample:   \${SDK DIR}\Samples\HTML\Preview .....	7-3
	Quad Device .....	7-5

Preview with quad device 7-5

HTML Sample: Connect with Quad mode ..... 7-5

HTML Sample: Connect with Single mode ..... 7-6

HTML Sample: Connect with Sequential mode (Not support motion  
setting/Detection)..... 7-7

HTML Sample: Connect with Single channel video server ..... 7-7





# 1

## Overview

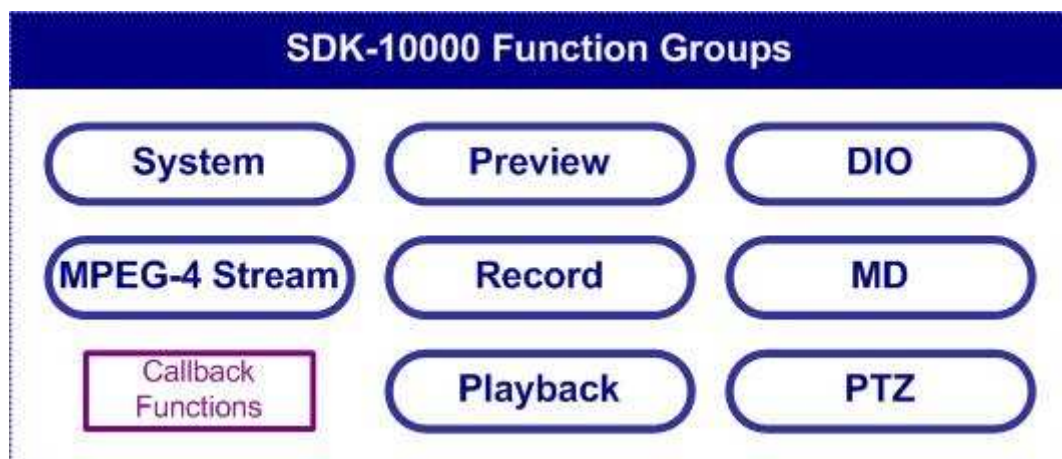
### Introduction

This material covers SDK architecture, data structure and procedures to illustrate the mechanisms to integrate the IP Surveillance devices. The content of this material is designed to lead the programmers go through the flow of the SDK and design their own application with supplied functions; they are organized in topics so that programmers may find the topics they want directly.

Please refer to Programming Guide for detailed API references.

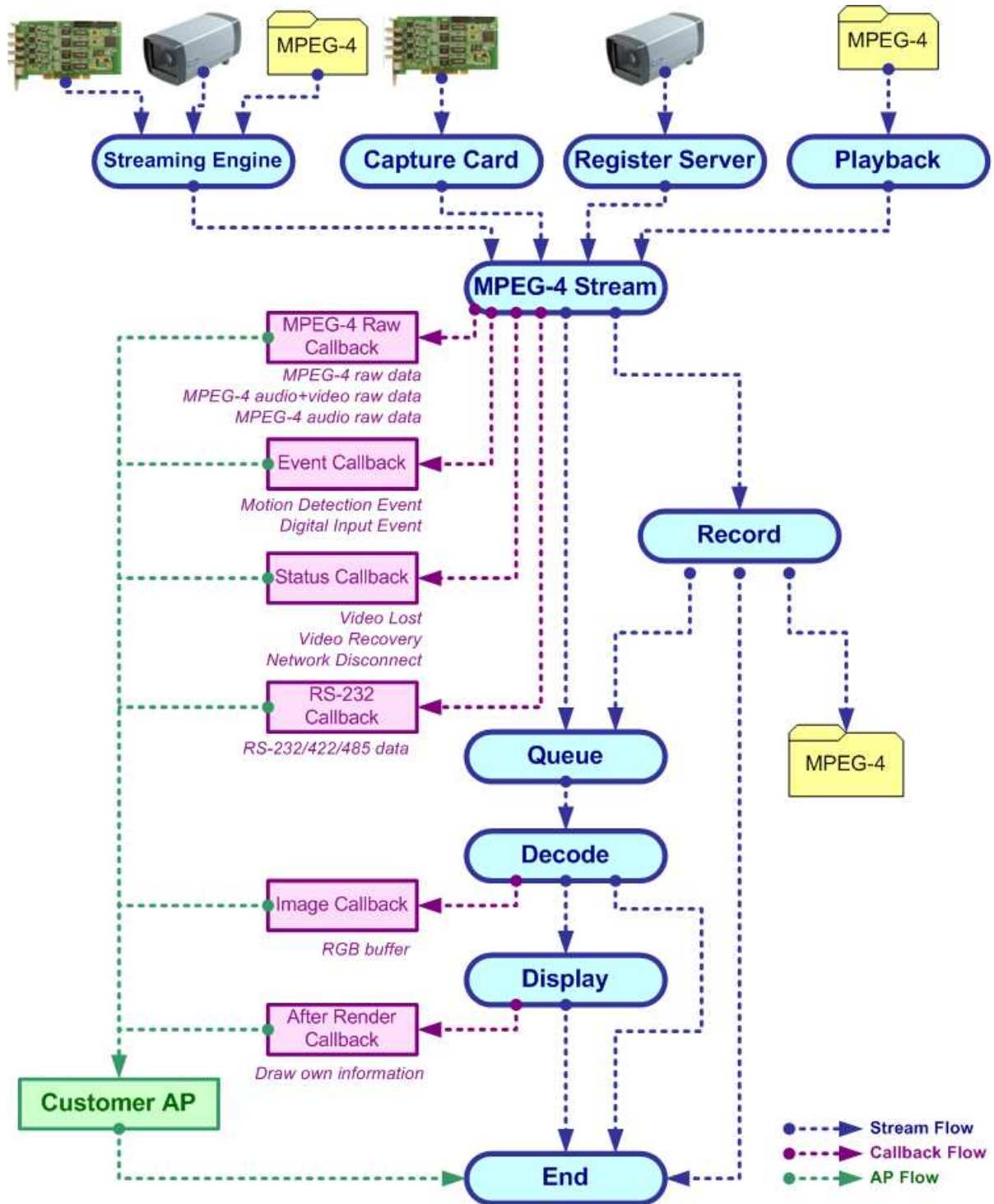
### SDK Function Groups

The whole SDK can be divided into following function groups.



## Architecture

SDK architecture and data flow is described as follow:



## Application Type

Based on the architecture and data flow, users may develop following application type:

1. **Full-featured Surveillance system:** Preview, Record, Playback, DIO event, MD event and PTZ functions
2. **Background recording:** record without preview. The stream can be configured as unicast or multicast mode
3. **Connection with event handling only:** connection only, wait for digital input or motion detection event; when the event triggered, then starts streaming and record the event
4. **Background recording with RGB buffer:** record without preview, receives RGB buffer to run user-defined motion detection algorithm at the same time
5. **Process MPEG-4 video stream:** advanced users may acquire MPEG-4 video stream and process by themselves. Related video, audio and audio+video callback functions are provided
6. **User-defined information on screen:** user may use after render callback function to draw user-defined information on preview window, including OSD text, draw video intelligence information
7. **Streaming engine support:** Control can connect to video server or remote playback through streaming engine. Now control support control port sending so that we can also send a PTZ command to video server through.

## Topics

Streaming Client Library is developed for MPEG-4/Motion JPEG/H.264 Video Network Streaming Application.

- Registration with Unicast / Multicast
- Preview / Record / Playback
- DIO Event Handling
- Motion Detection Event Handling
- PTZ Integration
- Status Callback
- IP Quad Integration
- Advanced Topics
  - ◆ Gets Video data via Video callback function
  - ◆ Gets RGB via image callback function
  - ◆ ACTi Video Time code format
  - ◆ Decode I Frame Only
  - ◆ Save ACTi Video raw data into AVI format
  - ◆ Gets RGB via image callback function

## What's New?

Following lists the new contents in this release:

### v1.2

- ◆ Auto frame rate by CPU thrash hole
- ◆ Inverse rendering image.
- ◆ New PTZ functions and absolute PTZ functions.
- ◆ Support Motion-JPEG
- ◆ Support H.264
- ◆ Support Mega-pixel mpeg4
- ◆ Jitter less function
- ◆ New Callback functions
- ◆ New Codec Type: IPP

### v1.2sp1

- ◆ New chapter about H.264, dual stream

# Register ActiveX Control

This section describes how to register the ActiveX control.

## Runtime DLL Files \${SDK DIR}\Bin

File	Description
nvUnifiedControl.ocx	Unified Control OCX

.

## Sample Codes **\${SDK DIR}\Samples**

SDK-10000 ActiveX Control v1.2 sample programs can be reached at **\${SDK DIR}\Samples**

SDK-10000 ActiveX Control v1.2 contains following sample codes:

1. Preview : preview functions with audio, event, multiple streams and quad device.
2. Record: record file, snapshot
3. Playback: playback recorded video archive
4. QuadURL: control quad video server
5. Motion: set motion detection parameters
6. PTZ: PTZ controls
7. Multiplayback: playback multiple recorded videos archive
8. PrivacyMask: reverse images and text display on video

All sample codes come with HTML, VB6 and VB.Net implementation in following directory :

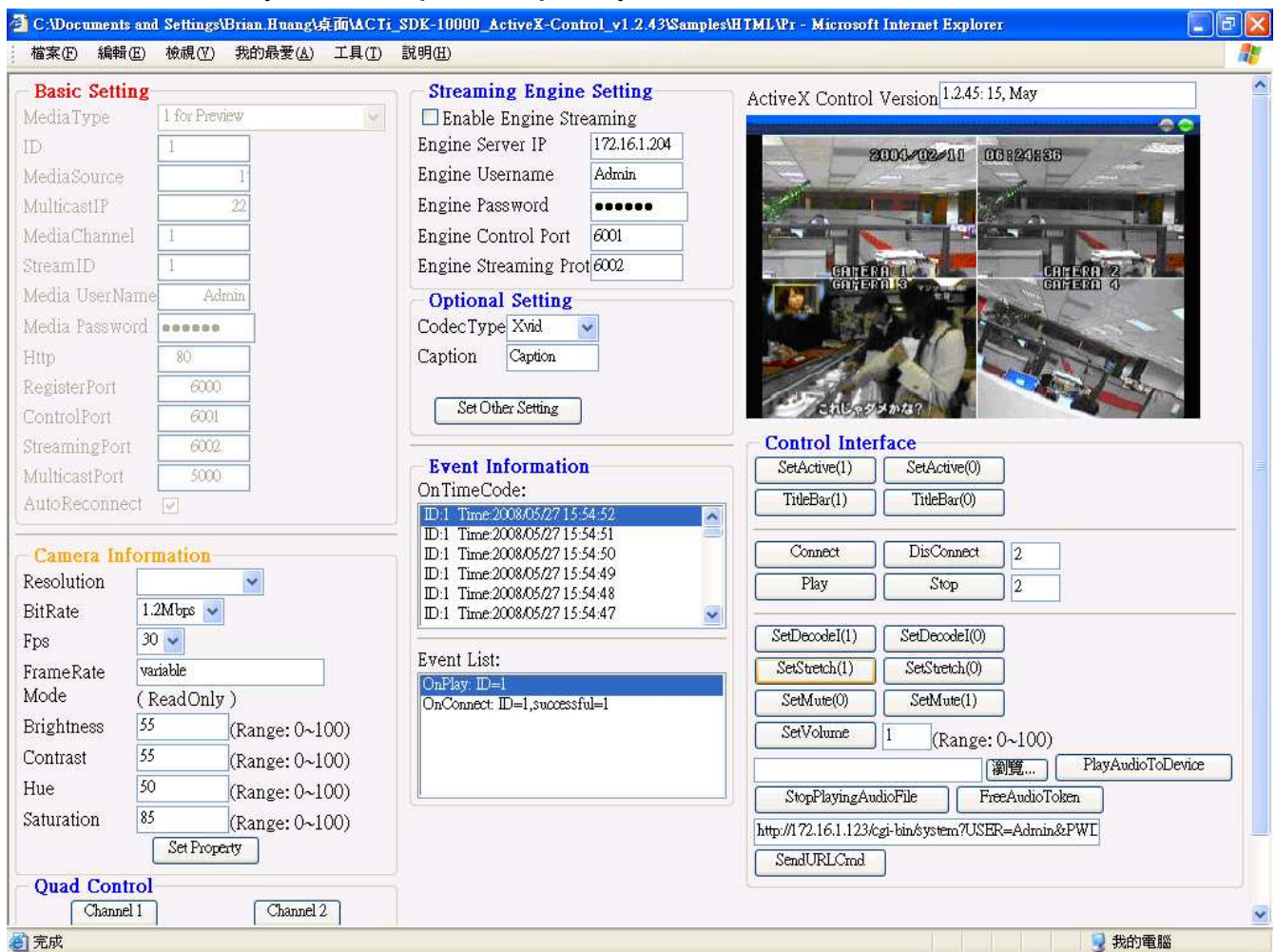
- **\${SDK DIR}\Samples\HTML**: HTML sample program

## Preview Sample Program

Preview Sample codes demonstrates following functions:

1. Connection mode: unicast, multicast
2. Audio
3. Event handling: connect, play, stop, disconnect, video loss, video recovery, network loss, connection recovery, mouse key, mouse move, Time code event
4. UI: set window active, toggle stretch mode, toggle title bar, mute
5. Decode I-Frame only
6. camera setup
7. quad device

### Preview HTML Sample Code: \${SDK DIR}\Samples\HTML\Preview.htm



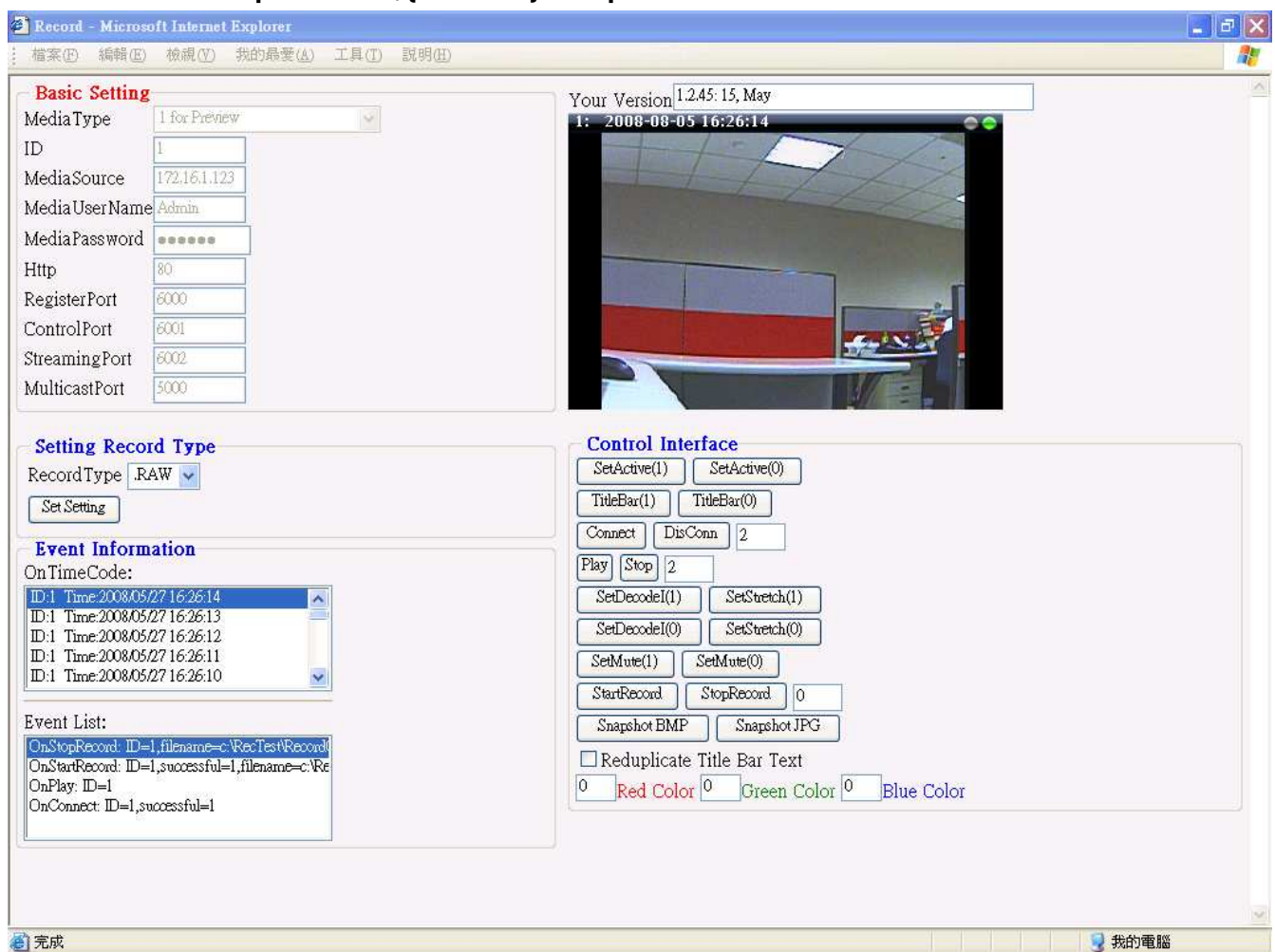


## Record Sample Program

Record sample codes demonstrate following functions:

1. Start/Stop Record
2. Snapshot
3. Decode I Frame only
4. Event handling: connect, play, stop, disconnect, record, save image, Time code event

### Record HTML Sample Code: \${SDK DIR}\Samples\HTML\Record.htm

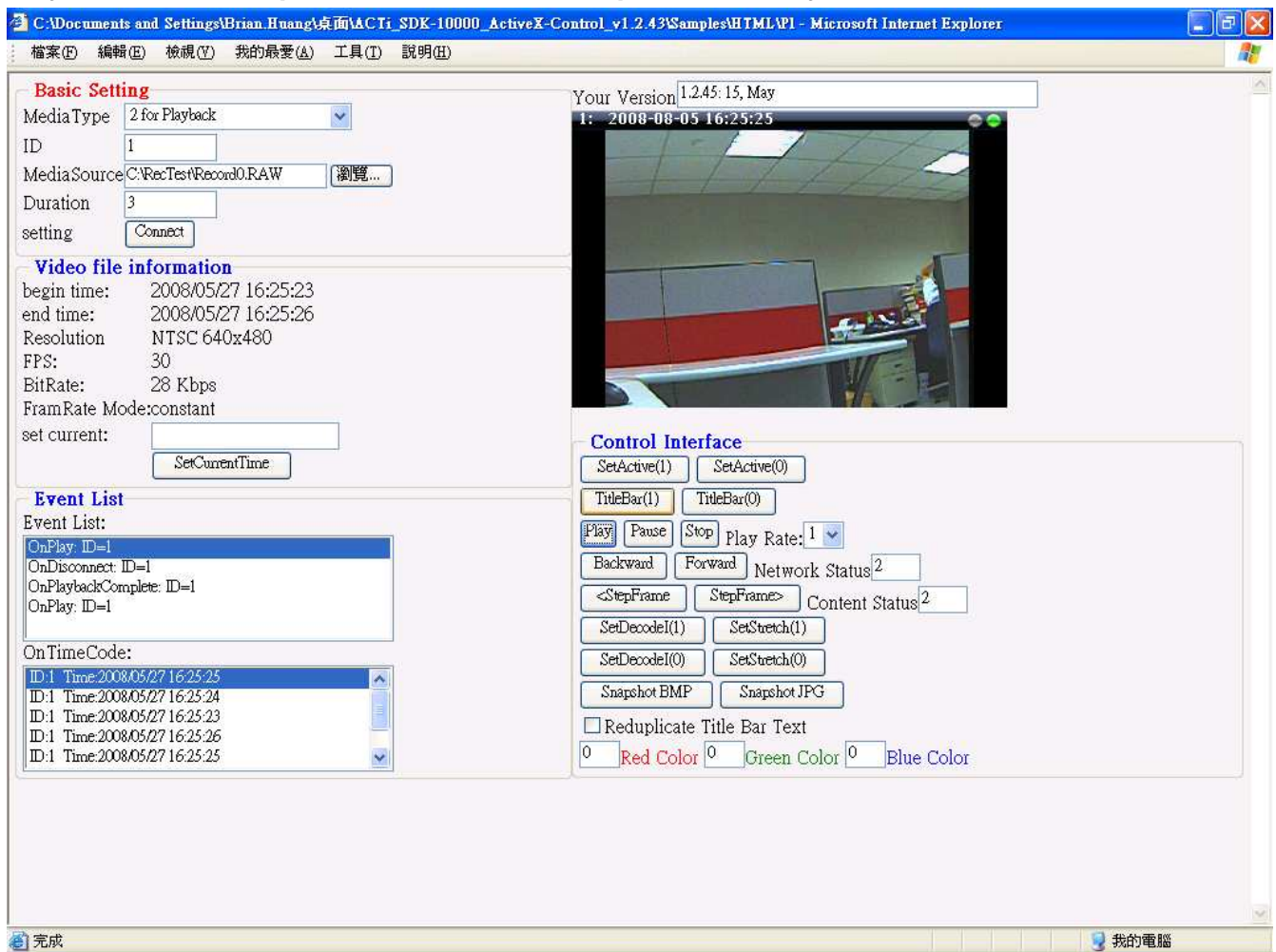


## Playback Sample Program

Playback Sample codes demonstrate following functions:

1. Open, Play, Pause, Stop
2. Play forward/backward, Fast forward/backward
3. Step-by-step forward/backward
4. Set play start time
5. Snapshot
6. UI Control: stretch, full screen mode
7. Event handling: connect, play, stop, disconnect, playback complete, save image, Time code event

### Playback HTML Sample Code: {SDK DIR}\Samples\HTML\Playback.htm



## MD Sample Program

Motion detection sample codes demonstrate following functions:

1. Set motion detection
2. Motion detection triggers
3. Event handling: connect, play, stop, disconnect, motion detect setting, motion event, record, DI event, save image, Time code event
4. DI DO event
5. title bar color setting
6. record and alarm record

### MD HTML Sample Code: {SDK DIR}\Samples\HTML\Motion.htm

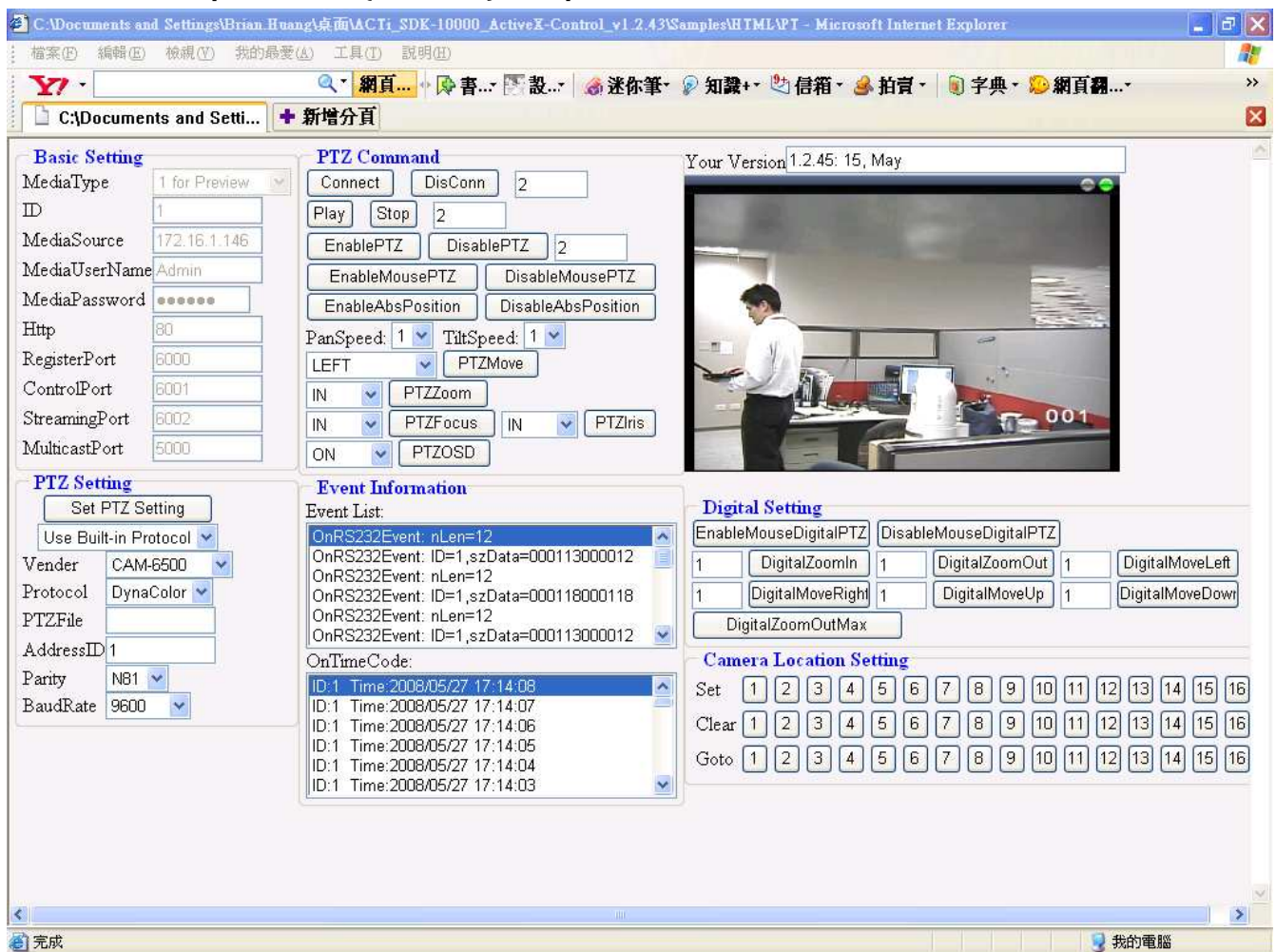


## PTZ Sample Program

PTZ sample program demonstrates following functions:

1. Connects different PTZ protocols
2. PTZ, Mouse PTZ
3. Digital zoom, Mouse digital PTZ
4. OSD Function
5. Preset position
6. Event handling: connect, play, stop, disconnect, RS232 event, Time code event

### PTZ HTML Sample Code: \${SDK DIR}\Samples\HTML\PTZ.htm



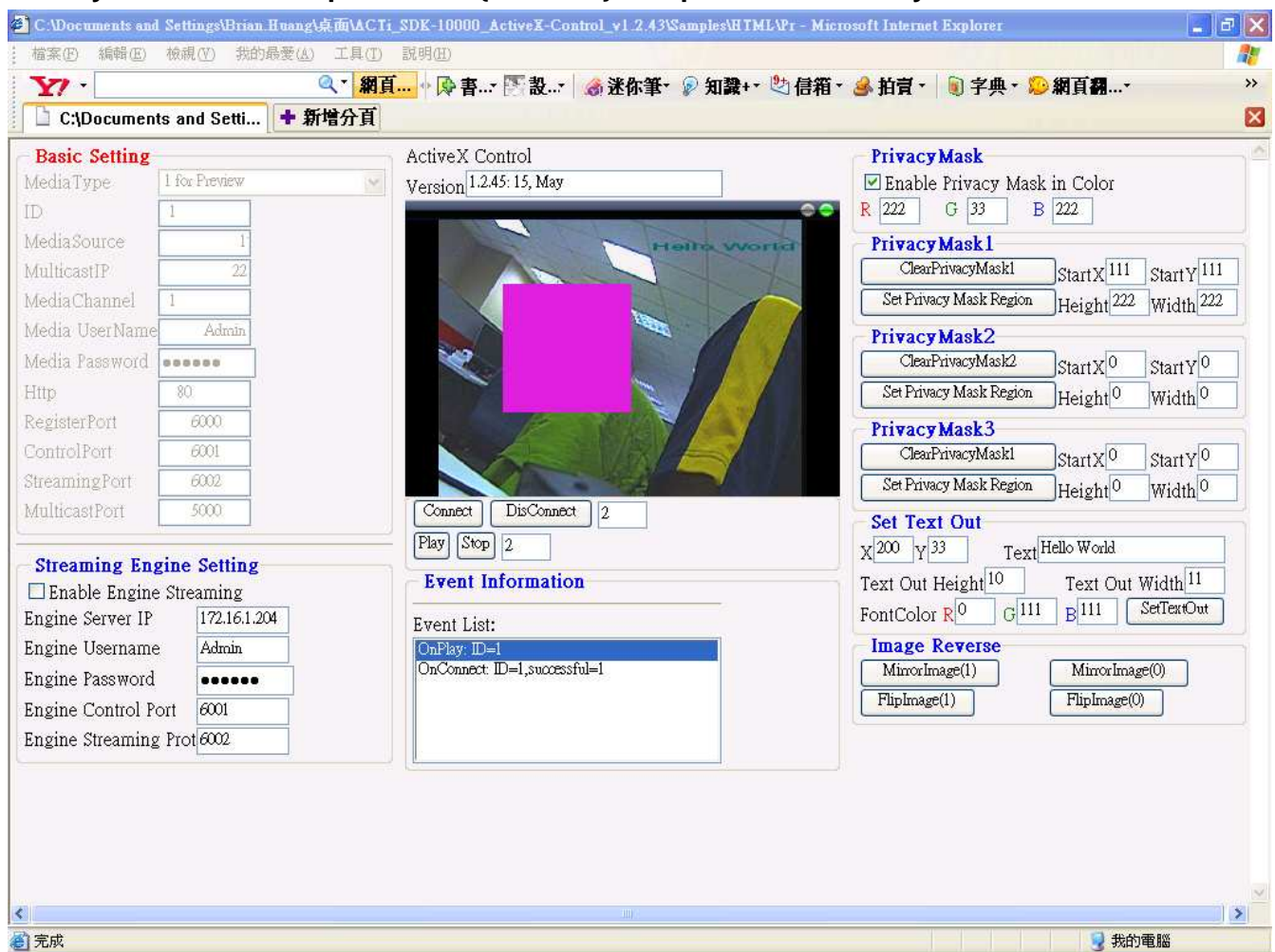


## PrivacyMask Sample Program

PrivacyMask sample program demonstrates following functions:

1. Privacy Mask setting
2. Image reverse
3. Set text out
4. Event handling: connect, play, stop, disconnect, mouse key double click

**PrivacyMask HTML Sample Code:** `${SDK DIR}\Samples\HTML\PrivacyMask.html`

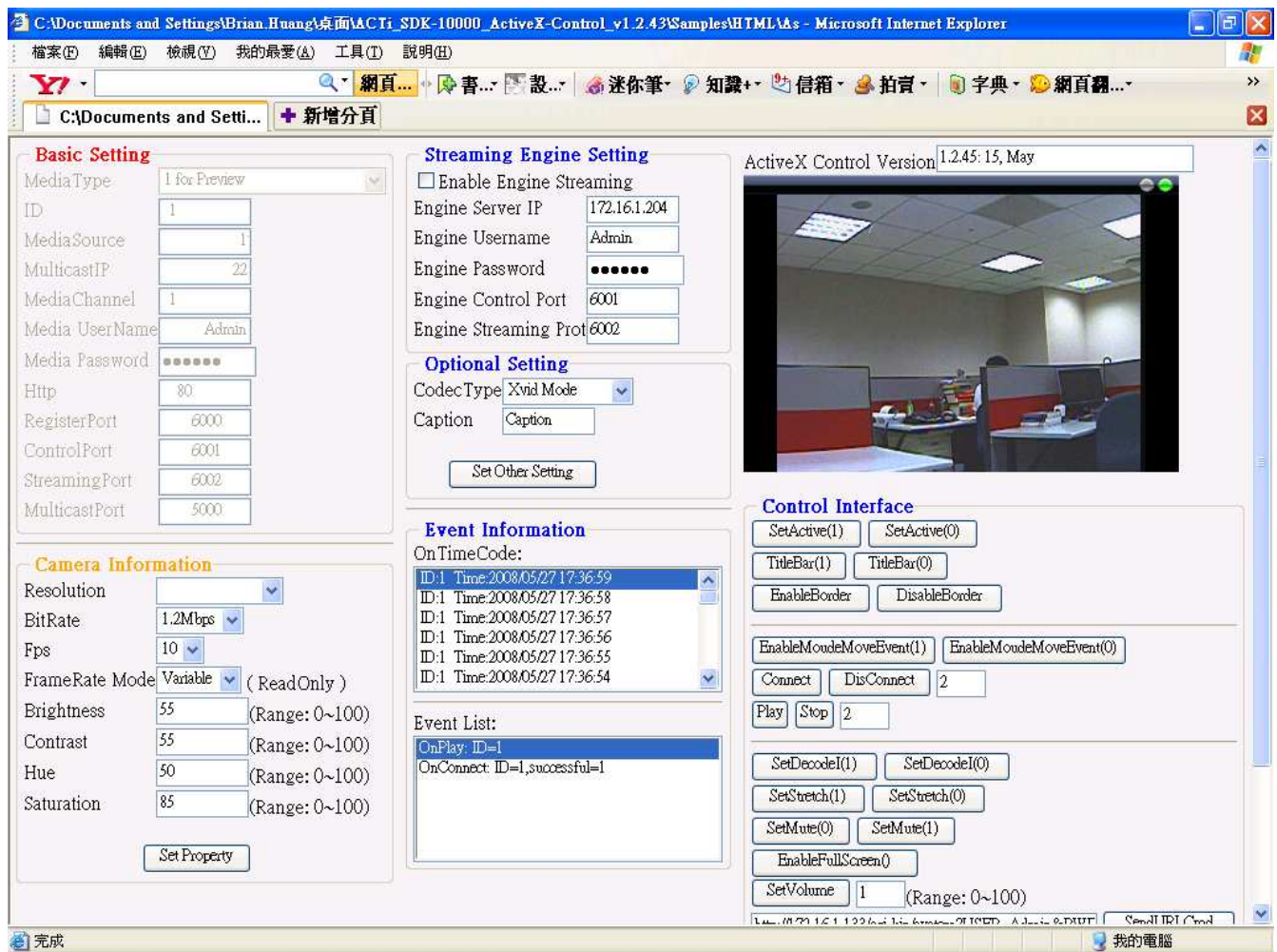


## Asynchronous Preview Sample Program

Asynchronous Preview sample program demonstrates following functions:

1. asynconnect
2. quad device
3. camera setting
4. Event handling: connect, play, stop, disconnect, video loss/recovery, connection loss/recovery, mouse key down, mouse double click, mouse move, time code
5. full screen
6. send URL command

### Asynchronous Preview HTML Sample Code: \${SDK DIR}\Samples\HTML\ Asynchronous Preview.htm



# 2

## Configure Device

### How to configure with video server

Before we want to use the video server or device. We should configure it to appropriate setting by URL commands\*. This section will describe how to detect, manage and configure IP devices. We can use any method that can send the URL command to IP devices. We are going to use the XMLHTTP object to get information from IP devices and how to adjust the video quality, resolution, bitrate... and so on.

\*Please also refer to the [Appendix for the complete ACTi URL Command listing](#).

### System Information

We can get the system information of IP devices by 3 steps as follow:

1. Initial **XMLHTTP** Object
2. Send the URL Command of request to IP device
3. Receive with **XMLHTTP.responseText**

#### HTML Sample:

```
var xmlhttp = new ActiveXObject("Microsoft.XMLHTTP") ;

strURL = 'http://192.168.1.100:80' ;
strURL = '/cgi-bin/system?USER=Admin&PWD=123456&SYSTEM_INFO' ;

xmlhttp.Open("get", strURL, false);
xmlhttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
xmlhttp.send();

szRet = xmlhttp.responseText;

//  Firmware Version = A1D-M2N-V2.13.02-NB
//  MAC Address = 00:0F:7C:00:1A:47
//  Production ID = SED2400-05I-1-00034
//  Factory Default Type = NTSC, Composite, Two ways Audio (0x71)
```

## System Property

We can get the system property of IP devices by 3 steps as follow:

1. Initial **XMLHTTP** Object
2. Send the URL Command of request to IP devise
3. Receive with **XMLHTTP.responseText**

### HTML Sample:

```
var xmlhttp = new ActiveXObject("Microsoft.XMLHTTP") ;

strURL = 'http://192.168.1.100:80' ;
strURL = '/cgi-bin/system?USER=Admin&PWD=123456&SYSTEM_PROPERTY' ;

xmlhttp.Open("get", strURL, false);
xmlhttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
xmlhttp.send();

szRet = xmlhttp.responseText;

//  SYSTEM='E'
//  TYPE='A'
//  NO_OF_CHANNEL='01'
//  MULTIPLEXING='X'
//  NO_OF_AUDIO_WAYS='2'
//  AUDIO_TYPE='PCM'
//  MOTION_TYPE='0'
//  PROTOCOL_TYPE='2'
```



# Video Color Adjustments

## Hue, Brightness, Contrast Setting

Steps to Gets/Sets product Video Property are listed as follow:

1. Initial **XMLHTTP** Object
2. Send the request URL Command
3. Receive with **XMLHTTP.responseText**

### HTML Sample:

```
var xmlhttp = new ActiveXObject("Microsoft.XMLHTTP") ;

// To Get the Video Property
strURL = 'http://192.168.1.100:80' ;
strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_BRIGHTNESS' ;
// strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_CONTRAST' ;
// strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_HUE' ;
// strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_SATURATION' ;

xmlhttp.Open("get", strURL, false);
xmlhttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
xmlhttp.send();

szRet = xmlhttp.responseText;

// Receive data format.
// VIDEO_BRIGHTNESS='55'
// VIDEO_CONTRAST='55'
// VIDEO_HUE='50'
// VIDEO_SATURATION='85'

// To Set the Video Property
strURL = 'http://192.168.1.100:80' ;
strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_BRIGHTNESS=10' ;
// strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_CONTRAST=20' ;
// strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_HUE=30' ;
// strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_SATURATION=40' ;

xmlhttp.Open("get", strURL, false);
```

```
xmlhttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");  
xmlhttp.send();
```

# Video Setting Configuration

## Setup Resolution, Frame Rate, Bit Rate

Steps to Get/Set product Video Setting are listed as follow:

1. Initial **XMLHTTP** Object
2. Send the request URL Command
3. Receive with **XMLHTTP.responseText**

### HTML Sample:

```
var xmlhttp = new ActiveXObject("Microsoft.XMLHTTP") ;

// To Get the Video Property
strURL = 'http://192.168.1.100:80' ;
strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_RESOLUTION' ;
// strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_FPS_NUM' ;
// strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_BITRATE' ;

xmlhttp.Open("get", strURL, false);
xmlhttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
xmlhttp.send();

szRet = xmlhttp.responseText;

// Receive data format.
// VIDEO_BRIGHTNESS='55'
// VIDEO_FPS_NUM='30'
// VIDEO_BITRATE='1.2M'
```

### Set the video property

```
// To Set the Video Property
strURL = 'http://192.168.1.100:80' ;
strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_RESOLUTION=N720x480' ;
// strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_FPS_NUM=6' ;
// strURL = '/cgi-bin/mpeg4?USER=Admin&PWD=123456&VIDEO_BITRATE=3M' ;

xmlhttp.Open("get", strURL, false);
xmlhttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
xmlhttp.send();
```

# Save and Reboot

When we configured the IP devices. Sometimes we need to save and reboot the IP devices. Below will describe how to save and reboot the IP devices with XMLHTTP object.

## Execute Save and Reboot Command

Steps to execute Save and Reboot Video device are listed as follow:

1. Initial **XMLHTTP** Object
2. Send the request URL Command
3. Receive with **XMLHTTP.responseText**

### HTML Sample:

```
var xmlhttp = new ActiveXObject("Microsoft.XMLHTTP") ;

// To Get the Video Property
strURL = 'http://192.168.1.100:80' ;
strURL = '/cgi-bin/system?USER=Admin&PWD=123456&SAVE_REBOOT' ;

xmlhttp.Open("get", strURL, false);
xmlhttp.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
xmlhttp.send();
```

# 3

## Preview / Record / Playback / PTZ

### Preview / Record Architecture

The control was integrated Preview, Record, Playback and PTZ functions that can be used very easy. There are most important for IP devices. The following will describe how to operate those.

### Connect to IP devices

We need to connect to network or Internet first before we want to get a video streaming. After we make sure the network or Internet is available. We can try to create a simple HTML to use control that can help us to easy get start.

First step we have to use the method, **connect()**, to register to IP devices. Below is the list of steps:

1. Get the device's IP address, account, password
2. Prepare port number
3. Call **connect()** method
4. Check if **OnConnect()** Event is successful

#### HTML Sample: \${SDK DIR}\Samples\HTML\Preview

```
<OBJECT id="oMedia" height=240 width=352
data=data:application/x-oleobject;base64,P2/zpuA6i0wvxKqCVl4L+BAHAABhJAAAzhgAA
A== classid=CLSID:F8E691A0-C92E-4E42-9CDA-62FC07A9483B VIEWASTEXT></OBJECT>

<script for="oMedia" event="OnConnect(nID,successful)">OnConnect(nID, successful);
</script>
<script for="oMedia" event="OnPlay(nID)">OnPlay(nID);</script>
<script for="oMedia" event="OnDisconnect(nID)">OnDisconnect(nID); </script>

oMedia.MediaType = 1 ;
oMedia.ID = 1 ;
oMedia.MediaSource = '192.168.0.100' ;
oMedia.MulticastIP = '228.5.6.1' ; // Set this property in Multicast Only
oMedia.MediaUserName = 'Admin' ;
oMedia.MediaPassword = '123456' ;
oMedia.Caption = 'Streaming Sample' ;
oMedia.Httpport = 80 ;
oMedia.RegisterPort = 6000 ;
```

```

oMedia.ControlPort      = 6001 ;
oMedia.StreamingPort    = 6002 ;
oMedia.MulticastPort    = 5000 ;
oMedia.ConnectTimeOut = 3 ;

oMedia.Connect(0) ;
oMedia.StartStream();

if (oMedia.NetworkStatus >= 2) {
    oMedia.Play();
}

function OnConnect(nID,successful) {
    switch (successful) {
        case 0 :
            alert('connect failed');          break ;
        case 1 :
            alert('connect success') ;          break ;
    }
}

function OnPlay(nID) {
    alert('camera' + nID + 'Play') ;
}

function OnDisconnect(nID) {
    alert('camera' + nID + 'disconnect') ;
}

```

# Preview(Unicast/Multicast/CaptureCard)

Preview is most important thing. After we understood how to connect to an IP devices, We are going to get a live video what include different media source e.g. Unicast, Multicast and CaptureCard mode at the ACTi Unified Control (AUC). Besides video preview we will show how to support audio also.

## Preview with Unicast Mode

Steps to start preview with unicast mode include:

1. Set MediaType to 1
2. Prepare IP address, account, password
3. Prepare port number
4. Set the MediaSource ONLY
5. Call **Connect()**
6. Check if **OnConnect()** Event is successful
7. Call **Play()**

### HTML Sample: \${SDK DIR}\Samples\HTML\Preview

```
<OBJECT id="oMedia" height=240 width=352
data=data:application/x-oleobject;base64,P2/zpuA6i0wvxKqCVl4L+BAHAABhJAAAzHgAA
A== classid=CLSID:F8E691A0-C92E-4E42-9CDA-62FC07A9483B VIEWASTEXT></OBJECT>

<script for="oMedia" event="OnConnect(nID,successful)">OnConnect(nID, successful);
</script>
<script for="oMedia" event="OnPlay(nID)">OnPlay(nID);</script>
<script for="oMedia" event="OnDisconnect(nID)">OnDisconnect(nID); </script>

oMedia.MediaType = 1 ;
oMedia.ID = 1 ;
oMedia.MediaSource = '192.168.0.100' ;
oMedia.MediaUserName = 'Admin' ;
oMedia.MediaPassword = '123456' ;
oMedia.Caption = 'Streaming Sample' ;
oMedia.Httpport = 80 ;
oMedia.RegisterPort = 6000 ;
oMedia.ControlPort = 6001 ;
oMedia.StreamingPort = 6002 ;
oMedia.MulticastPort = 5000 ;
oMedia.ConnectTimeOut = 1 ;
```

```

oMedia.Connect(0) ;
oMedia.StartStream();
if (oMedia.NetworkStatus >= 2) {
    oMedia.Play();
}

function OnConnect(nID,successful) {
    switch (successful) {
        case 0 :
            alert('connect failed');          break ;
        case 1 :
            alert('connect success') ;          break ;
    }
}

function OnPlay(nID) {
    alert('camera' + nID + 'Play') ;
}

function OnDisconnect(nID) {
    alert('camera' + nID + 'disconnect') ;
}

```



## Preview with Multicast Mode

Steps to start preview with multicast mode include:

1. Set MediaType to 100 (Multicast without control) or 101(Multicast with control)
2. Prepare IP address, account, password
3. Prepare port number
4. Set the MediaSource and MulticastIP
5. Call **connect()**
6. Check if **onConnect()** Event is successful
7. Call **Play()**

### HTML Sample: \${SDK DIR}\Samples\HTML\Preview

```
<OBJECT id="oMedia" height=240 width=352
data=data:application/x-oleobject;base64,P2/zpuA6i0wvxKqCV14L+BAHAABhJAAAzHgAA
A== classid=CLSID:F8E691A0-C92E-4E42-9CDA-62FC07A9483B VIEWASTEXT></OBJECT>

<script for="oMedia" event="OnConnect(nID,successful)">OnConnect(nID, successful);
</script>
<script for="oMedia" event="OnPlay(nID)">OnPlay(nID);</script>
<script for="oMedia" event="OnDisconnect(nID)">OnDisconnect(nID); </script>

oMedia.MediaType = 101 ; // Multicast with control
oMedia.ID = 1 ;
oMedia.MediaSource = '192.168.0.100' ;
oMedia.MulticastIP = '228.5.6.100' ;
oMedia.MediaUserName = 'Admin' ;
oMedia.MediaPassword = '123456' ;
oMedia.Caption = 'Streaming Sample' ;
oMedia.Httpport = 80 ;
oMedia.RegisterPort = 6000 ;
oMedia.ControlPort = 6001 ;
oMedia.StreamingPort = 6002 ;
oMedia.MulticastPort = 5000 ;
oMedia.ConnectTimeOut = 1 ;

oMedia.Connect(0) ;
oMedia.StartStream();
if (oMedia.NetworkStatus >= 2) {
    oMedia.Play();
}
```

```
function OnConnect(nID,successful) {  
    switch (successful) {  
        case 0 :  
            alert('connect failed');          break ;  
        case 1 :  
            alert('connect success') ;          break ;  
    }  
}  
  
function OnPlay(nID) {  
    alert('camera' + nID + 'Play') ;  
}  
  
function OnDisconnect(nID) {  
    alert('camera' + nID + 'disconnect') ;  
}
```

## Preview with CaptureCard Mode

Steps to start preview with unicast mode include:

1. Set channel number
2. Set the MediaType to 41
3. Call **connect()**
4. Check if **onConnect()** Event is successful
5. Call **Play()**

### HTML Sample: \${SDK DIR}\Samples\HTML\Preview

```
<OBJECT id="oMedia" height=240 width=352
data=data:application/x-oleobject;base64,P2/zpuA6i0wvxKqCVl4L+BAHAABhJAAAzhgAA
A== classid=CLSID:F8E691A0-C92E-4E42-9CDA-62FC07A9483B VIEWASTEXT></OBJECT>

<script for="oMedia" event="OnConnect(nID,successful)">OnConnect(nID, successful);
</script>
<script for="oMedia" event="OnPlay(nID)">OnPlay(nID);</script>
<script for="oMedia" event="OnDisconnect(nID)">OnDisconnect(nID); </script>

oMedia.MediaType = 101 ; // Multicast with control
oMedia.ID = 1 ;
oMedia.MediaSource = '192.168.0.100' ;
oMedia.MulticastIP = '228.5.6.100' ;
oMedia.MediaUserName = 'Admin' ;
oMedia.MediaPassword = '123456' ;
oMedia.Caption = 'Streaming Sample' ;
oMedia.Httpport = 80 ;
oMedia.RegisterPort = 6000 ;
oMedia.ControlPort = 6001 ;
oMedia.StreamingPort = 6002 ;
oMedia.MulticastPort = 5000 ;
oMedia.ConnectTimeOut = 1 ;

oMedia.Connect(0) ;
oMedia.StartStream();
if (oMedia.NetworkStatus >= 2) {
    oMedia.Play();
}

function OnConnect(nID,successful) {
    switch (successful) {
```

```
        case 0 :  
            alert('connect failed');          break ;  
        case 1 :  
            alert('connect success') ;          break ;  
    }  
}  
  
function OnPlay(nID) {  
    alert('camera' + nID + 'Play') ;  
}  
  
function OnDisconnect(nID) {  
    alert('camera' + nID + 'disconnect') ;  
}
```

## Preview with Audio

Steps to start preview with audio mode include:

1. Prepare IP address, account, password
2. Prepare port number
3. Call **connect()**
4. Check if **onConnect()** Event is successful
5. Call **Play()**
6. Set property **Mute = 0**

### HTML Sample: \${SDK DIR}\Samples\HTML\Preview

```
// Must Connect to Device FIRST.  
    oMedia.Mute = 0 ; // Audio On  
    // oMedia.Mute = 1; // Audio Off
```

## Preview with 2-way audio

Steps to preview with 2-way audio include:

1. Prepare IP address, account, password
2. Prepare port number
3. Call **connect()**
4. Check successful in **onConnect()** Event
5. Call **Play()**
6. Set property **Mute = 0**
7. Call **GetAudioToken()**
8. Check **AudioToken** Property.
9. To Start audio out to call **StartAudioTransfer()**
10. To Stop audio out to call **StopAudioTransfer()**



**IMPORTANT:** One IP device has only 1 audio token; if the token is taken by one application, then no other application may acquire the audio token again. Remember to free audio token after the 2-way audio function is done.

### HTML Sample: \${SDK DIR}\Samples\HTML\Preview

```
// Must Connect to Device FIRST.  
If (oMedia.GetAudioToken()) {  
    oMedia.StartAudioTransfer() ;  
}  
oMeia.StopAudioTransfer();
```



## Preview with I-Frame Decoding only

This chapter describes a mechanism on how to decrease CPU loading. With this mechanism, MPEG-4 software decoder will decode I-Frame only and drops all P-Frame before decoding.

Steps to preview with I-Frame decoding only include:

1. Register to the IP device
2. Preview with start stream
3. Set to I-Frame decoding only with **setDecodeI()** function



**NOTE:** With **setDecodeI()** function, the CPU loading can be decreased dramatically.



**IMPORTANT:** **setDecodeI()** function only affects preview and CPU loading; recording still records with I-frame and P-frame as setup.

### HTML Sample: \${SDK DIR}\Samples\HTML\Preview

```
// Must Connect to Device and Play stream FIRST.  
oMedia.SetDecodeI(1); // Decode I-frame Only  
// oMedia.SetDecodeI (0); // Decode I-frame and P-frame
```



# Record

The recording job can be selected with live video preview or without preview (Background Recording). AUC support two file types that are saved to disk. One is RAW data file that is standard format of ACTi, another is AVI that can be played on MS MediaPlayer.

We can record the video without Streaming Client Library is developed for MPEG-4 Video Network Streaming Application.

## Record with unicast mode

## Background record with multicast mode

Streaming Client Library is developed for MPEG-4 Video Network Streaming Application.

Steps to start preview with multicast mode without preview include:

1. Set MulticastIP Property;
2. Call **connect()**
3. Check if **onConnect()** Event is successful
4. Call **startRecord()**



**NOTE:** Application may start recording without preview.

### HTML Sample: \${SDK DIR}\Samples\HTML\Record

```
// Must Connect to Device FIRST.  
oMedia.RecordType = 0; // 0 = RAW, 1 = AVI, 2 = RAW+idx  
oMedia.StartRecord('c:\\ABC.RAW');  
oMedia.StopRecord();
```

## Record to AVI file

### HTML Sample: \${SDK DIR}\Samples\HTML\Record

```
// Must Connect to Device FIRST.  
oMedia.RecordType = 1; // 0 = RAW, 1 = AVI, 2 = RAW + idx  
oMedia.StartRecord('c:\\ABC.AVI');
```

## Alarm Recording with DI event

Steps to start alarm recording include:

1. Setup pre-event recording time and post-event recording time
2. Register to the IP devices
3. Setup Enable DIEvent
4. Start alarm recording
5. Stop alarm recording

### HTML Sample: \${SDK DIR}\Samples\HTML\Motion

```
// Must Connect to Device FIRST.
// Enable DI Event
oMedia.EnableDigitalInput()

// In OnDIEvent Function
<script for="oPreview0" event="OnDIEvent(nDI)">OnDIEvent(nDI);</script>
function OnDIEvent( nDI ) {
    oMedia.StartAlarmRecord('c:\ABC.RAW');
}
```

# Playback

Steps to operate playback functions include:

1. Open file
2. Sets playback play speed
3. Calls playback operation, including play forward, play backward, seed operation

## HTML Sample: \${SDK DIR}\Samples\HTML\Playback

```
<OBJECT                id="oMedia"                height=240                width=352
data=data:application/x-oleobject;base64,P2/zpuA6i0wvxKqCVl4L+BAHAABhJAAAzhgAA
A== classid=CLSID:F8E691A0-C92E-4E42-9CDA-62FC07A9483B VIEWASTEXT></OBJECT>

oMedia.MediaType          = 2 ;
oMedia.ID                 = 1 ;
oMedia.MediaSource        = 'C:\\ABC.RAW' ;

oMedia.Connect(0) ;
oMedia.StartStream();
if (oMedia.NetworkStatus >= 2) {
    oMedia.Play();
}
```

# PTZ(PAN/TILT/ZOOM)

This material covers how to integrate PTZ protocol with prepared information.

In the product architecture, the PTZ operation is defined as transparent tunnel; in this way, the PTZ protocol information does not keep in the firmware, and user's application has to parse and prepare PTZ commands in the application side.

To shorten the integration process, SDK provides implemented and tested PTZ protocol files, so that application may just utilize the PTZ protocols that has been prepared.

## HTML Sample: \${SDK DIR}\Samples\HTML\PTZ

```
oMedia.MediaType = 4 ;  
// Must Connect to Device and Play Streaming FIRST.  
  
oMedia.Vendor = 'Pelco' ;  
oMedia.Protocol = 'Pelco-P' ;  
oMedia.AddressID = 1 ;  
oMedia.PTZPostMode = 1 ;  
oMedia.Parity = 'N81' ;  
oMedia.BaudRate = 9600 ;  
  
oMedia.EnablePTZ  
  
oMedia.PTZMove("LEFT") ;  
// oMedia.PTZMove("STOP") ;  
// oMedia.PTZZoom("IN");  
// oMedia.PTZFocus("IN");
```



# 4

## Event Handling

### Digital I/O Architecture

This material covers SDK architecture, data structure and sample programs to illustrate the methods to integrate ACTi's IP Surveillance products.

#### Receives Digital Input Event

HTML Sample: \${SDK DIR}\Samples\HTML\Motion

```
// Enable DI Event
oMedia.EnableDigitalInput()

<script for="oPreview0" event="OnDIEvent(nID, nDI)">OnDIEvent(nDI);</script>
function OnDIEvent( nID, nDI ) {
    AddOption( 'OnDIEvent: nDI=' + nDI ) ;
}
```

#### Send Digital Output

Steps to receive digital input event include:

1. Register to the IP devices
2. Call **DigitalOutput()** function to send event to the digital output device

HTML Sample: \${SDK DIR}\Samples\HTML\Motion

```
oMedia.DigitalOutput(1,0,0,0) '1,0,0,0 for DO1
                                , 0,1,0,0 for DO2
                                , 1,1,0,0 for DO1 and DO2
```

# Motion Detection Event Handling

## Sets Motion Detection parameters

Steps to setup motion detection parameters include:

1. Register to the IP devices
2. Setup motion detection callback function
3. Sets motion detection parameters
4. Process motion detection event in the callback function



**NOTE:** The parameter to set the range of the motion detection window has to be the multiplier of 16, if not, the number will be aligning to the multiplier of 16. For example, if the application set the range as 125, then it will be align to 128.

### HTML Sample: \${SDK DIR}\Samples\HTML\Motion

```
oMedia.StartMDSetup();  
oMedia.SetMotionSetting(1,1,1,100,100,50) ;  
oMedia.StopMDSetup();
```

## Gets Motion Detection Settings

### HTML Sample: \${SDK DIR}\Samples\HTML\Motion

```
oMedia.StartMotionSetting();  
<script for="oPreview0" event="OnMDSetting(nMD, x1, y1, x2, y2,  
Sens)">OnMDSetting(nMD, x1, y1, x2, y2, Sens);</script>  
  
function OnMDSetting(nMD, x1, y1, x2, y2, Sens) {  
    AddOption( 'OnMDSetting: '+nMD+', '+x1+', '+y1+', '+x2+', '+y2+', '+Sens ) ;  
}
```



## Receives Motion Detection Trigger Event

### HTML Sample:

```
<script for="oPreview0" event="OnMDEventStart(nID,nMD)">OnMDEventStart(nID,nMD);
</script>
<script for="oPreview0" event="OnMDEventEnd(nID)">OnMDEventEnd(nID); </script>

var nFile = 0 ;
var bMDRec = 0 ;

function OnMDEventEnd(nID) {
    AddOption( 'OnMDEventEnd:' ) ;

    oPreview0.StopAlarmRecord() ;
    bMDRec = 0 ;
}

function OnMDEventStart(nID ,nMD ) {
    if (bMDRec == 0) {
        if (sTime) {
            AddOption( 'OnMD:' + nMD + '/' + sTime ) ;
        } else {
            AddOption( 'OnMD:' + nMD ) ;
        }
    }

    if (nFile > 200) nFile=0;
    oPreview0.StartAlarmRecord('c:\\RecTest\\AlarmRecord' + nFile + '.RAW');
    nFile++ ;
    bMDRec = 0 ;
}
}
```

## Status Callback – video lost, recover, disconnect event

### HTML Sample: \${SDK DIR}\Samples\HTML\Preview

```
<script for="oPreview0" event="OnVideoLoss(nID)">OnVideoLoss(nID);
</script>
<script for="oPreview0" event="OnVideoRecovery(nID)">OnVideoRecovery(nID);
</script>
<script for="oPreview0" event="OnConnectionLoss(nID)">OnConnectionLoss(nID);
</script>
```

```
<script                                                                    for="oPreview0"
event="OnConnectionRecovery(nID)">OnConnectionRecovery(nID); </script>

function OnConnectionLoss(nID) {
    AddOption( 'OnConnectionLoss:' ) ;
}
function OnConnectionRecovery(nID) {
    AddOption( 'OnConnectionRecovery:' ) ;
}
function OnVideoLoss(nID) {
    AddOption( 'OnVideoLoss:' ) ;
}
function OnVideoRecovery(nID) {
    AddOption( 'OnVideoRecovery:' ) ;
}
```

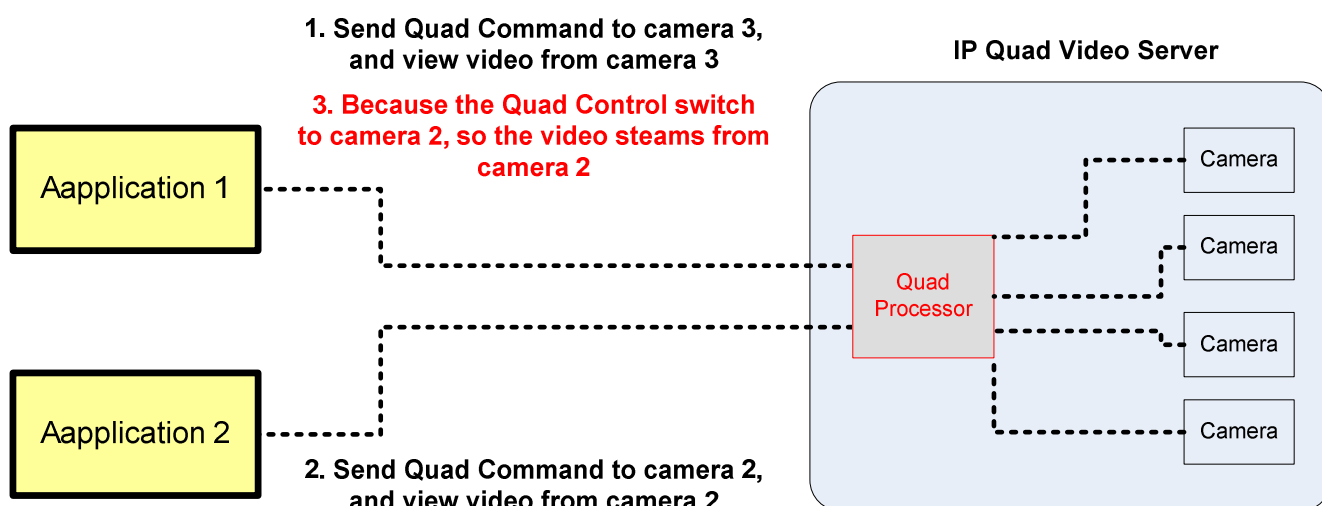
# 5

## IP Quad Video Server Integration

### IP Quad Architecture

IP Quad is a Quad processor which connects to 4 analog video sources then multiplexed by a quad processor; in this way, an IP Quad video server may generate 1 Full D1 video stream or 4 CIF video streams at the same time

IP Quad video server firmware contains URL commands, so that application may simply send out the URL command to control the behavior of it.



**NOTE:** There is only one quad processor in the device, so when an application sends a URL command to the IP Quad video server, then the quad processor will execute the commands specified, and all connected application will receive the same result from quad processor.

## IP Quad URL Commands

Application may just use URL Command to perform these tasks to setup and control SED-2300Q; for information that needs to retrieve from SED-2300Q (e.g. Retrieve MPEG-4 stream, record to files, motion detection event, digital input event), the calling methods are all the same as SDK-10000 v1.0.

IP Quad's quad control is based on URL Command, which means that you need to send out the URL Command to IP Quad to set certain parameters.

#### HTTP Code Status

HTTP Code	HTTP Text	Description
200	OK	The request has succeeded, but an application error can still occur, which will be returned as an application error code.
204	No Content	The server has fulfilled the request, but there is no new information to send back.
400	Bad Request	The request had bad syntax or was inherently impossible to be satisfied.
401	Unauthorized	The request requires user authentication or the authorization has been refused.
404	Not Found	The server has not found anything matching the request.
409	Conflict	The request could not be completed due to a conflict with the current state of the resource.
500	Internal Error	The server encountered an unexpected condition which prevented it from fulfilling the request.
503	Service Unavailable	The server is unable to handle the request due to temporary overload.

Example :

#### Return success http context

```
HTTP/1.0 200 OK\r\n
Content-Type: text/plain\r\n
\r\n
```

#### Return failed http context

```
HTTP/1.0 200 OK\r\n
Content-Type: text/plain\r\n
\r\n
ERROR: error description
```

#### How to set display mode

<b>Syntax</b>	http://192.168.1.1/cgi-bin/quad?DISPLAY=n
---------------	---

#### How to get display mode

<b>Syntax</b>	http://192.168.1.1/cgi-bin/quad?DISPLAY
---------------	---

<parameter>	<values>	Description
DISPAY	n: 0~4	0: quad display 1: display channel 1 2: display channel 2 3: display channel 3 4: display channel 4

#### How to set osd enabled

<b>Syntax</b>	<code>http://192.168.1.1/cgi-bin/quad?OSD_ENABLED=0xnn</code>
---------------	---

#### How to get osd enabled status

<b>Syntax</b>	<code>http://192.168.1.1/cgi-bin/quad?OSD_ENABLED</code>
---------------	--

<parameter>	<values>	Description
OSD_ENABLED	0xnn : hexadecimal	BIT0: 1:title name enabled BIT1: 1:video loss enabled BIT2: 1:motion detect enabled BIT3: 1:date time enabled BIT4: 1:DIO status enabled BIT5: Reserved BIT6: Reserved BIT7: Reserved

#### How to set motion detect enabled

<b>Syntax</b>	<code>http://192.168.1.1/cgi-bin/quad?MOTION_ENABLED=0xnn</code>
---------------	--

#### How to get motion enabled status

<b>Syntax</b>	<code>http://192.168.1.1/cgi-bin/quad?MOTION_ENABLED</code>
---------------	---

<parameter>	<values>	Description
MOTION_ENABLED	0xnn : hexadecimal	BIT0: 1:channel 1 motion detect enabled BIT1: 1:channel 2 motion detect enabled BIT2: 1:channel 3 motion detect enabled BIT3: 1:channel 4 motion detect enabled BIT4: Reserved BIT5: Reserved BIT6: Reserved BIT7: Reserved

#### How to set sensitive for motion detect

<b>Syntax</b>	<code>http://192.168.1.1/cgi-bin/quad?CHANNEL=n&amp;SENSITIVE=m</code>
---------------	--

#### How to get sensitive setting

<b>Syntax</b>	http://192.168.1.1/cgi-bin/quad?CHANNEL=n&SENSITIVE
---------------	---

<parameter>	<values>	Description
CHANNEL	n: 1~4	channel number
SENSITIVE	m: 0~15	0: more sensitive . .. 8: middle sensitive . .. 15: less sensitive

#### How to set title name

<b>Syntax</b>	http://192.168.1.1/cgi-bin/quad?CHANNEL=n&TITLE_NAME=xxxxxxxx
---------------	---

#### How to get title name setting

<b>Syntax</b>	http://192.168.1.1/cgi-bin/quad?CHANNEL=n&TITLE_NAME
---------------	--

<parameter>	<values>	Description
CHANNEL	n: 1~4	channel number
TITLE_NAME	xxxxxxxx: title name	max length: 8bytes ASCII: A~Z & 0~9 & space

#### How to set brightness

<b>Syntax</b>	http://192.168.1.1/cgi-bin/quad?CHANNEL=n&BRIGHTNESS=m
---------------	--

#### How to get brightness setting

<b>Syntax</b>	http://192.168.1.1/cgi-bin/quad?CHANNEL=n&BRIGHTNESS
---------------	--

<parameter>	<values>	Description
CHANNEL	n: 1~4	channel number
BRIGHTNESS	m: 0~255	0: -25IRE . .. 128: 0IRE . .. 255: 25IRE

#### How to set contrast

<b>Syntax</b>	http://192.168.1.1/cgi-bin/quad?CHANNEL=n&CONTRAST=m
---------------	--

#### How to get contrast setting

<b>Syntax</b>	http://192.168.1.1/cgi-bin/quad?CHANNEL=n&CONTRAST
---------------	--

<parameter>	<values>	Description
-------------	----------	-------------

CHANNEL	n: 1~4	channel number
CONTRAST	m: 0~255	0: 0% . .. 128: 100% . .. 255: 200%

#### How to set saturation

<b>Syntax</b>	<code>http://192.168.1.1/cgi-bin/quad?CHANNEL=n&amp;SATURATION=m</code>
---------------	---

#### How to get saturation setting

<b>Syntax</b>	<code>http://192.168.1.1/cgi-bin/quad?CHANNEL=n&amp;SATURATION</code>
---------------	---

<parameter>	<values>	Description
CHANNEL	n: 1~4	channel number
SATURATION	m: 0~255	0: 0% . .. 128: 100% . .. 255: 200%

#### How to set hue

<b>Syntax</b>	<code>http://192.168.1.1/cgi-bin/quad?CHANNEL=n&amp;HUE=m</code>
---------------	--

#### How to get contrast setting

<b>Syntax</b>	<code>http://192.168.1.1/cgi-bin/quad?CHANNEL=n&amp;HUE</code>
---------------	--

<parameter>	<values>	Description
CHANNEL	n: 1~4	channel number
HUE	m: 0~255	0: -180degree . .. 128: 0degree . .. 255: 180degree

#### How to get system information

<b>Syntax</b>	<code>http://192.168.1.1/cgi-bin/system?INFO</code>
---------------	---

#### Http return context

Firmware Version = SED2300Q-20050404.02-AC-D1

MAC Address = 00:0F:7C:00:00:67

Factory Default Type = NTSC (0x51)

Serial ID = SED2300-04I-8-00027

Model Number = SED-2300Q (11)



How to set factory default

<b>Syntax</b>	<code>http://192.168.1.1/cgi-bin/system?FACTORY_DEFAULT</code>
---------------	--

How to save all setting to flash and reboot system

<b>Syntax</b>	<code>http://192.168.1.1/cgi-bin/system?SAVE_REBOOT</code>
---------------	--



# 6

## Advanced Topics

### Preview Functions

#### Preview with Full Screen

Steps to start preview with full screen mode include:

1. Prepare IP address, account, password
2. Prepare port number
3. Call **connect()**
4. Check if **onConnect()** Event is success
5. Call **Play()**
6. Call **setFullscreen(1)**

**HTML Sample: \${SDK DIR}\Samples\HTML\Preview**

```
// Must Connect to Device FIRST.  
oMedia.setFullscreen(1); // Full Screen On
```

## Preview with GDI or DirectDraw render

Steps to start preview with DDraw render mode include:

1. Prepare IP address, account, password
2. Prepare port number
3. Call **connect()**
4. Check the error code in **onConnect()** Event
5. Call **Play()**

### HTML Sample: \${SDK DIR}\Samples\HTML\Preview

```
// Must Connect to Device FIRST.  
oMedia.RenderType = 1; // 0 = GDI, 1 = DirectDraw
```

## Preview with PCI-5100, Xvid and FFMPEG decoder

Steps to start preview with PCI-5100 decoder card include:

1. Prepare IP address, account, password
2. Prepare port number
3. Call **connect()**
4. Check the error code in **OnConnect()** Event
5. Call **Play()**

### HTML Sample: \${SDK DIR}\Samples\HTML\Preview

```
// Must Connect to Device FIRST.  
oMedia.DecodeType = 2;
```

## Preview with Intel IPP Codec.

Steps to start preview with PCI-5100 decoder card include:

1. Copy Bin\IPP\\*.dll to system32\\*. manually.
2. Prepare IP address, account, password
3. Prepare port number
4. Call **connect()**
5. Set CodecType = 3
6. Call **Play()**

# Connect Functions

## Asynchronize connection

Steps to start preview with full screen mode include:

1. Prepare IP address, account, password
2. Prepare port number
3. Call **Connect(1)**
4. Wait **onConnect()** event callback
5. Call **Play()** function in **onConnect()** event

### HTML Sample: `${SDK DIR}\Samples\HTML\Asynchronous_Preview`

```
// Must Connect to Device FIRST.
oMedia.Connect(1) ; // 0 = Xvid, 1 = FFMpeg, 2 = PCI-5100 decoder card

function OnConnect( nID, ConnectSuccessful ) {
    if(ConnectSuccessful==1) {
        oMedia.StartStream ();

        oMedia.Play() ;
    }
}
```

# 7

## New Support

### H.264 and Motion-JPEG

#### Preview with H.264 or Motion-JPEG

These steps are the same as Preview by other codec type stream. First step we have to use the method, **connect()**, to register to IP devices. The following is the list of steps:

1. Get the device's IP address, account, password
2. Prepare port number
3. Call **connect()** method
4. Check if **onConnect()** Event is successful

#### HTML Sample: \${SDK DIR}\Samples\HTML\Preview

```
<OBJECT id="oMedia" height=240 width=352
data=data:application/x-oleobject;base64,P2/zpuA6i0wvxKqCVl4L+BAHAABhJAAAzHgAA
A== classid=CLSID:F8E691A0-C92E-4E42-9CDA-62FC07A9483B VIEWASTEXT></OBJECT>

<script for="oMedia" event="OnConnect(nID,successful)">OnConnect(nID, successful);
</script>
<script for="oMedia" event="OnPlay(nID)">OnPlay(nID);</script>
<script for="oMedia" event="OnDisconnect(nID)">OnDisconnect(nID); </script>

oMedia.MediaType = 1 ;
oMedia.ID = 1 ;
oMedia.MediaSource = '192.168.0.100' ;
oMedia.MulticastIP = '228.5.6.1' ; // Set this property in Multicast Only
oMedia.MediaUserName = 'Admin' ;
oMedia.MediaPassword = '123456' ;
oMedia.Caption = 'Streaming Sample' ;
oMedia.Httpport = 80 ;
oMedia.RegisterPort = 6000 ;
oMedia.ControlPort = 6001 ;
oMedia.StreamingPort = 6002 ;
oMedia.MulticastPort = 5000 ;
oMedia.ConnectTimeOut = 3 ;

oMedia.Connect(0) ;
```

```
if (oMedia.NetworkStatus >= 2) {
    oMedia.Play();
}

function OnConnect(nID,successful) {
    switch (successful) {
        case 0 :
            alert('connect failed');          break ;
        case 1 :
            alert('connect success') ;          break ;
    }
}

function OnPlay(nID) {
    alert('camera' + nID + 'Play') ;
}

function OnDisconnect(nID) {
    alert('camera' + nID + 'disconnect') ;
}
```



# Dual stream

## Preview with dual stream device

To play dual stream, we have to set MedialChannel and TCPVideoStreamID(TCP2.0) before we connect to the dual stream device. Otherwise, the control will connect to default stream. The rest of the steps are the same as Preview for other devices. Frist step we have to use the method, **connect()**, to register to IP devices. Below is the list of steps:

1. Get the device's IP address, account, password
2. Set MedialChannel
3. Set TCPvideoStreamID (TCP 2.0)/ RTPVideoTrackNumber and RTPAudioTrackNumber (RTP)
4. Prepare port number
5. Call **connect()** method
6. Check if **onConnect()** Event is successful

### HTML Sample: \${SDK DIR}\Samples\HTML\Preview

```
<OBJECT id="oMedia" height=240 width=352
data=data:application/x-oleobject;base64,P2/zpuA6i0WvxKqCVl4L+BAHAABhJAAAzhgAA
A== classid=CLSID:F8E691A0-C92E-4E42-9CDA-62FC07A9483B VIEWASTEXT></OBJECT>

<script for="oMedia" event="OnConnect(nID,successful)">OnConnect(nID, successful);
</script>
<script for="oMedia" event="OnPlay(nID)">OnPlay(nID);</script>
<script for="oMedia" event="OnDisconnect(nID)">OnDisconnect(nID); </script>

oMedia.MediaType = 1 ;
oMedia.ID = 1 ;
oMedia.MediaSource = '192.168.0.100' ;
oMedia.MulticastIP = '228.5.6.1' ; // Set this property in Multicast Only
oMedia.MediaUserName = 'Admin' ;
oMedia.MediaPassword = '123456' ;
oMedia.Caption = 'Streaming Sample' ;
oMedia.MedialChannel = 1;
oMedia.TCPVideoStreamID = 1; (0 or 1)
oMedia.Httpport = 80 ;
oMedia.RegisterPort = 6000 ;
oMedia.ControlPort = 6001 ;
oMedia.StreamingPort = 6002 ;
```

```

oMedia.MulticastPort      = 5000 ;
oMedia.ConnectTimeOut = 3 ;

oMedia.Connect(0) ;
oMedia.StartStream();
if (oMedia.NetworkStatus >= 2) {
    oMedia.Play();
}

function OnConnect(nID,successful) {
    switch (successful) {
        case 0 :
            alert('connect failed');          break ;
        case 1 :
            alert('connect success') ;          break ;
    }
}

function OnPlay(nID) {
    alert('camera' + nID + 'Play') ;
}

function OnDisconnect(nID) {
    alert('camera' + nID + 'disconnect') ;
}

```

# Quad Device

## Preview with quad device

To play with quad device, we have to set DeviceType and QuadDeviceMode before we connect to the quad device(ACD2000Q).

There are using two important properties to configuration Control for quad device and define in nvUnifiedControl.idl.

```
[id(74), helpstring("property DeviceType")] DeviceType DeviceType;  
[id(84), helpstring("property QuadDeviceMode")] QuadDeviceMode QuadDeviceMode;
```

```
[helpstring("QuadDevice Mode")]  
typedef [v1_enum] enum {  
    QUAD_MODE = 0,  
    SINGLE_MODE = 1,  
    SEQUENTIAL_MODE = 2,  
    AUTO_DETECT = 3  
}QuadDeviceMode;
```

```
[helpstring("Device Type")]  
typedef [v1_enum] enum  
{  
    _SINGLE_CHANNEL_VIDEO_SERVER = 0,  
    _ACD2000Q_VIDEO_SERVER=1,  
    _SED2300Q_VIDEO_SERVER=2,//not support now.  
    _AUTO_DETECT = 3  
}DeviceType;
```

Also you can assignment AUTO\_DETECT to properties of DeviceType/QuadDeviceMode, then ActiveX control will be get those information via URL commands automatic. It's mean there needs a bit of time to process it during connection session.

### HTML Sample: Connect with Quad mode

```
<object id="oMedia" classid="CLSID:F8E691A0-C92E-4E42-9CDA-62FC07A9483B"  
width="352" height="264"></object>
```

```

<input type="button" value="Connect" onclick="Connect()">

oMedia.MediaType          = 1 ;
oMedia.ID                 = 1 ;
oMedia.MediaSource        = '192.168.0.100' ;
oMedia.MulticastIP        = '228.5.6.1' ;    // Set this property in Multicast Only
oMedia.MediaUserName      = 'Admin' ;
oMedia.MediaPassword      = '123456' ;
oMedia.Caption            = 'Streaming Sample' ;
oMedia.MediaChannel      = 1;
oMedia.TCPVideoStreamID = 0;
oMedia.Httpport           = 80 ;
oMedia.RegisterPort       = 6000 ;
oMedia.ControlPort        = 6001 ;
oMedia.StreamingPort      = 6002 ;
oMedia.MulticastPort      = 5000 ;
oMedia.ConnectTimeOut = 3 ;
oMedia.DeviceType = 1;    //0:Single channel device, 1: Quad device
oMedia.QuadDeviceMode = 0; //0:Quad mode, 1:Single mode, 2:Sequential mode
oMedia.Connect(0) ;
oMedia.StartStream();

```

### HTML Sample: Connect with Single mode

```

<object id="oMedia" classid="CLSID:F8E691A0-C92E-4E42-9CDA-62FC07A9483B"
width="352" height="264"></object>

<input type="button" value="Connect" onclick="Connect()">

oMedia.MediaType          = 1 ;
oMedia.ID                 = 1 ;
oMedia.MediaSource        = '192.168.0.100' ;
oMedia.MulticastIP        = '228.5.6.1' ;    // Set this property in Multicast Only
oMedia.MediaUserName      = 'Admin' ;
oMedia.MediaPassword      = '123456' ;
oMedia.Caption            = 'Streaming Sample' ;
oMedia.MediaChannel      = 1;    //channel number(1~4)
oMedia.TCPVideoStreamID = 0;    // 0 - 3
oMedia.Httpport           = 80 ;
oMedia.RegisterPort       = 6000 ;
oMedia.ControlPort        = 6001 ;
oMedia.StreamingPort      = 6002 ;
oMedia.MulticastPort      = 5000 ;
oMedia.ConnectTimeOut = 3 ;
oMedia.DeviceType = 1;    //0:Single channel device, 1: Quad device

```

```
oMedia.QuadDeviceMode = 1; //0:Quad mode, 1:Single mode, 2:Sequential mode
oMedia.Connect(0) ;
```

### HTML Sample: Connect with Sequential mode (Not support motion setting/Detection)

```
<object id="oMedia" classid="CLSID:F8E691A0-C92E-4E42-9CDA-62FC07A9483B"
width="352" height="264"></object>

<input type="button" value="Connect" onclick="Connect()">

oMedia.MediaType = 1 ;
oMedia.ID = 1 ;
oMedia.MediaSource = '192.168.0.100' ;
oMedia.MulticastIP = '228.5.6.1' ; // Set this property in Multicast Only
oMedia.MediaUserName = 'Admin' ;
oMedia.MediaPassword = '123456' ;
oMedia.Caption = 'Streaming Sample' ;
oMedia.MediaChannel = 1;
oMedia.TCPVideoStreamID = 0;
oMedia.Httpport = 80 ;
oMedia.RegisterPort = 6000 ;
oMedia.ControlPort = 6001 ;
oMedia.StreamingPort = 6002 ;
oMedia.MulticastPort = 5000 ;
oMedia.ConnectTimeOut = 3 ;
oMedia.DeviceType = 1; //0:Single channel device, 1: Quad device
oMedia.QuadDeviceMode = 2; //0:Quad mode, 1:Single mode, 2:Sequential mode
oMedia.Connect(0) ;
oMedia.StartStream();
```

### HTML Sample: Connect with Single channel video server

```
<object id="oMedia" classid="CLSID:F8E691A0-C92E-4E42-9CDA-62FC07A9483B"
width="352" height="264"></object>

<input type="button" value="Connect" onclick="Connect()">

oMedia.MediaType = 1 ;
oMedia.ID = 1 ;
oMedia.MediaSource = '192.168.0.100' ;
oMedia.MulticastIP = '228.5.6.1' ; // Set this property in Multicast Only
oMedia.MediaUserName = 'Admin' ;
oMedia.MediaPassword = '123456' ;
oMedia.Caption = 'Streaming Sample' ;
oMedia.MediaChannel = 1;
oMedia.TCPVideoStreamID = 0;
oMedia.Httpport = 80 ;
oMedia.RegisterPort = 6000 ;
```

```
oMedia.ControlPort      = 6001 ;  
oMedia.StreamingPort    = 6002 ;  
oMedia.MulticastPort    = 5000 ;  
oMedia.ConnectTimeOut = 3 ;  
oMedia.DeviceType = 0;      //0:Single channel device, 1: Quad device  
oMedia.QuadDeviceMode = 1; //0:Quad mode, 1:Single mode, 2:Squential mode  
oMedia.Connect(0) ;  
oMedia.StartStream();
```