iCVS Client SDK

Reference Specification

**Table of Contents**

[1 Overview 2](#_Toc398538932)

[2 SDK Development Guide 2](#_Toc398538933)

[2.1 Function call sequence 2](#_Toc398538934)

[3 Customized data types 3](#_Toc398538935)

[3.1 Various constants 3](#_Toc398538936)

[3.2 Resource Type 3](#_Toc398538937)

[3.3 Frame type 3](#_Toc398538938)

[3.4 Type definition 4](#_Toc398538939)

[3.5 File information 4](#_Toc398538940)

[3.6 GPS information 4](#_Toc398538941)

[4 Function interface for various resource types 5](#_Toc398538942)

[4.1 Initialize and terminate the protocol stack 5](#_Toc398538943)

[4.1.1 Initialize the protocol stack 5](#_Toc398538944)

[4.1.2 Terminate and clear the protocol stack 5](#_Toc398538945)

[4.2 Connect and disconnect platform or leading edge device 5](#_Toc398538946)

[4.2.1 Connect a platform 5](#_Toc398538947)

[4.2.2 Disconnect a platform 6](#_Toc398538948)

[4.3 Get a list of devices and resources 6](#_Toc398538949)

[4.3.1 Build a list of devices under the platform 6](#_Toc398538950)

[4.3.2 Build a single device under the platform 6](#_Toc398538951)

[4.3.3 Build resources under device 6](#_Toc398538952)

[4.4 Get audio and video data 7](#_Toc398538953)

[4.4.1 Receive the audio & video and serial data 7](#_Toc398538954)

[4.4.2 Activate transcoding stream 7](#_Toc398538955)

[4.4.3 Activate the intercom stream 7](#_Toc398538956)

[4.4.4 Stop stream 8](#_Toc398538957)

[4.4.5 Receive a frame of data 8](#_Toc398538958)

[4.4.6 Send a frame of data 8](#_Toc398538959)

[4.5 Access events 8](#_Toc398538960)

[4.5.1 Receive notice 8](#_Toc398538961)

[4.6 Various resources related orders 9](#_Toc398538962)

[4.6.1 All resources public command 9](#_Toc398538963)

[4.6.2 Site resources 11](#_Toc398538964)

[4.6.3 Enter the video resource 12](#_Toc398538965)

[4.6.4 Serial port resources 17](#_Toc398538966)

[4.6.5 PTZ resources 19](#_Toc398538967)

[4.6.6 Alarm output resource 23](#_Toc398538968)

[4.6.7 Leading end storage resources 23](#_Toc398538969)

[4.6.8 Platform storage resources 25](#_Toc398538970)

# Overview

This SDK consists of 4 header files and an import library for VC as well as a dynamic link library. Among them, iCVSC.h is a header file that declares the exported function;ICVSCTypes.h is a header file for declaring customized data type and defining various constants;iCVSCError.h is a header file that defines all error codes.

# SDK Development Guide

## Function call sequence

|  |  |
| --- | --- |
| Initialize the iCap protocol stack | **IC\_Initialize ()** |

|  |  |
| --- | --- |
| Login the platform | **IC\_Open ()** |

|  |  |
| --- | --- |
| Build a list of devices under the platform | **IC\_ForkPUList ()** |

|  |  |
| --- | --- |
| Build resources under device | **IC\_ForkPUResource ()** |

|  |  |
| --- | --- |
| Get resource name  Get resource type | **IC\_GetResourceName ()**  **IC\_GetResourceType ()** |

|  |  |
| --- | --- |
| According to the resource description, determine the types of resources, operate various resources, access or set the attributes of each resource | |
| Enable audio & video stream | **IC\_StartStream ()** |
| Receive notice | **IC\_ReceiveNotify()** |

|  |  |
| --- | --- |
| Disenable audio & video stream | **IC\_StopStream ()** |

|  |  |
| --- | --- |
| Exit the platform | **IC\_Close ()** |

|  |  |
| --- | --- |
| Terminate and clear iCAP the protocol stack | **IC\_Terminate ()** |

# Customized data types

## Various constants

enum

{

MAX\_STR\_LEN = 128, / \* Maximum string length \* /

};

## Resource Type

static const char RT\_SELF[] = "ST"; /\* Site resources \*/

static const char RT\_IV[] = "IV";/\* Input Video, IV \*/

static const char RT\_IA[] = "IA";/\* Input Audio, IA \*/

static const char RT\_OV[] = "OV";/\* Output Audio, OA \*/

static const char RT\_OA[] = "OA";/\* Output Audio, OA \*/

static const char RT\_SP[] = "SP";/\*Serial Port, SP \*/

static const char RT\_PTZ[] = "PTZ";/\* PTZ \*/

static const char RT\_IDL[] = "IDL";/\* Input Digital Line, IDL \*/

static const char RT\_ODL[] = "ODL";/\* Output Digital Line, ODL \*/

static const char RT\_DP[] = "DP";/\* Displayer \*/

static const char RT\_SG[] = "SG";/\* Storager, SG \*/

static const char RT\_WIFI[] = "WIFI";/\* WIFI module \*/

static const char RT\_WM[] = "WM";/\* Wireless Module \*/

static const char RT\_GPS[] = "GPS";/\* GPS \*/

## Frame type

enum FRAME\_TYPE

{

frame\_type\_video = 1, / \* video frame \* /

frame\_type\_audio = 2, / \* audio frame \* /

frame\_type\_data = 3, / \* data frame \* /

frame\_type\_gps = 4, / \* GPS frame \* /

frame\_type\_sp = 5, / \* Serial port frame \* /

frame\_type\_pic = 6, / \* picture frame \* /

};

## Type definition

typedef void\* IC\_HANDLE; / \* Handle \* /

typedef char IC\_STR[MAX\_STR\_LEN];/ \* String type \* /

## File information

typedef struct \_FILE\_INFO

{

char szName[32];/\* file name \*/

char szPath[256];/ \* file path \* /

int nSize;/\* file size \*/

int nBeginTime;/\* beginning time \*/

int nEndTime;/\* end time \*/

char szReasons[256];/ \* Recording reason \* /

} IC\_FILE\_INFO;

## GPS information

typedef struct \_GPS\_INFO

{

float fLatitude;/ \* Longitude, unit degree \* /

float fLongitude;/ \* Latitude, unit degree \* /

float fBearing;/ \* Direction, unit degree, [0,360), due north 0, increasing clockwise \* /

float fSpeed;/ \* Speed ​​in km / h \* /

float fAltitude;/ \* Elevation,unit:meters \* /

int nUTC;/ \* UTC time of this GPS data, unit:second \* /

int nTimestamp;/ \* Timestamp of this GPS data, unit:millisecond \* /

} IC\_GPS\_INFO;

# Function interface for various resource types

## Initialize and terminate the protocol stack

### Initialize the protocol stack

Function:int IC\_Initialize (void)

Parameter:none

Return: Please refer to the error code and description

Description: This function needs to be initialized before calling other methods.

### Terminate and clear the protocol stack

Function:int IC\_Terminate (void)

Parameter:none

Return: Please refer to the error code and description

Description: This function should be called to release resources before the program exits.

## Connect and disconnect platform or leading edge device

### Connect a platform

Function: int IC\_Open (IN const char\* pszAddress, IN const char\* pszUsername, IN const char\* pszPassword, IN const char\* pszEpid, OUT IC\_HANDLE \*hSession)

Parameter: const char \* pszAddress [in] The address of the target platform (in the format "ip: port");

const char \* pszUsername [in] login user name;

const char \* pszPassword [in] login password;

const char \* pszEpid [in] Enterprise ID;

IC\_HANDLE \* hSession [out] Returns the connection index after the connection is successful

Return: Please refer to the error code and description

Description: Use this function to connect to a remote site.

### Disconnect a platform

Function: int IC\_Close (IN IC\_HANDLE hSession)

Parameter: IC\_HANDLE hSession [in] connected index ;

Return: Please refer to the error code and description

Description: Before disconnecting a platform, please stop all the data streams that are being acquired under this platform .

## Get a list of devices and resources

### Build a list of devices under the platform

Function: int IC\_ForkPUList(IN IC\_HANDLE hSession, OUT IC\_HANDLE \*hPUArray, IN OUT int \*nCount, IN int nOffset)

Parameter: UINT uiSession [in] The index of the connection;

IC\_HANDLE \* hPUArray [out] an array consisting of device resource handles;

int \* nCount [in, out] the size of the incoming data in array while being input, the number of data actually get while being output.

int nOffset [in] offset;

Return: Please refer to the error code and description

Description: the consistence between nCount input and output indicates that the list has not been fully obtained, then it is needed to obtain again, nOffset should be equal to the total number of lists obtained previously. While this function is called each time, hPUArray will return different values, therefore it is usually called once after connecting to the platform.

### Build a single device under the platform

Function: int IC\_ForkOnePU(IN IC\_HANDLE hSession, IN const char\* pszPUID, OUT IC\_HANDLE \*hPU)

Parameter: IC\_HANDLE hSession [in] connected index ;

const char \* pszPUID [in] The PUID of the device;

IC\_HANDLE \* hPU [out] Pointer of the device's resource handle;

Return: Please refer to the error code and description

Description: While this function is called each time, hPUArray will return different values, therefore it is usually called once after connecting to the platform.

### Build resources under device

Function: int IC\_ForkPUResource(IN IC\_HANDLE hPU, OUT IC\_HANDLE \*hResourcesArray, IN OUT int \*nArrayCount)

Parameter: IC\_HANDLE hPU [in] Handle of device resource;

IC\_HANDLE \* hResourcesArray [out] An array of resource handles under the device;

int \* nArrayCount [in, out] The size of the array that is being input, the number of arrays actually get while being output.

Return: Please refer to the error code and description

Description: While this function is called each time, hResourcesArray will return different values, therefore it is usually called once after connecting to the platform.

## Get audio and video data

### Receive the audio & video and serial data

Function: int IC\_StartStream(IN IC\_HANDLE hResource, IN int nStreamID, OUT IC\_HANDLE \*hStream)

Parameter: IC\_HANDLE hResource [in] The target resource handle, if video data is required, please input video resource handle;

int nStreamID [in] Target stream index;

IC\_HANDLE \* hStream [out] Get the stream handle returned after successful getting;

Return: Please refer to the error code and description

Description: if this function is used to start the audio & video and get serial data, the function will returns a hStream stream handle after successful receipt and this handle will be input in IC\_ReceiveFrame () function for collecting data.

### Activate transcoding stream

Function: int IC\_StartTranscodeStream(IN IC\_HANDLE hResource, IN const char\* szAlg, IN const char\* szResolution, IN int nBitRate, IN int nFrameRate, OUT IC\_HANDLE \*hStream)

Parameter: Temporarily empty.

Return:

Description:

### Activate the intercom stream

Function:int IC\_StartTalk(IN IC\_HANDLE hResource, OUT IC\_HANDLE \*hStream)

Parameter:IC\_HANDLE hResource [in] The target resource handle, where the OA resource handle is required;

IC\_HANDLE \* hStream [OUT] stream handle returned after function is successfully performed

Return: Please refer to the error code and description.

Description:hStream is used to be input in IC\_ReceiveFrame () and IC\_SendFrame () to achieve intercom data reception and transmission.

### Stop stream

Function:IC\_StopStream(IN IC\_HANDLE hStream)

Parameter:IC\_HANDLE hStream [in] Handle of stream to be stopped;

Return: Please refer to the error code and description.

Description:None

### Receive a frame of data

Function:int IC\_ReceiveFrame(IN IC\_HANDLE hStream, OUT char \*pFrame, IN OUT int \*nLen, OUT int \*nFrameType, OUT int \*nKeyFrameFlag)

Parameter:IC\_HANDLE hStream [in] Target stream handle;

char \* pFrame [out] Data receive buffer pointer;

int \* nLen [in, out] Size of the data entered in buffer while being input, the length of the data actually received while being output;

int \* nFrameType [out] Frame data type, please refer to the frame type;

int \* nKeyFrameFlag [out] the key frame or not, 1 is true, 0 is false;

Return: Please refer to the error code and description.

Description: This function needs to be called regularly, and remains being called during a one-off call until the function returns an error (no data).

### Send a frame of data

Function:int IC\_SendFrame(IN IC\_HANDLE hStream, IN char \*pFrame, IN int nLen, IN int nFrameType)

Parameter:IC\_HANDLE hStream [in] Target stream handle;

char \* pFrame [in] data transmission buffer pointer;

int nLen [int] The length of data to be actually sent;

int nFrameType [int] Frame type of the data to be sent;

Return: Please refer to the error code and description.

Description: none.

## Access events

### Receive notification

Function:int IC\_ReceiveNotify(IN IC\_HANDLE hSession, OUT IC\_STR szID, OUT IC\_STR szSrcID, OUT int \*nIndex, OUT int \*nUTC)

Parameter:IC\_HANDLE hSession [in] connected index;

IC\_STR szID [out] Received notification ID;

IC\_STR szSrcID [out] Received notification source ID;

int \* nIndex [out] Received notification source resource index;

int \* nUTC [out] The time when the notification received occurs;

Return: Please refer to the error code and description.

Description: Temporarily empty.

## Various resources related orders

### All resources public command

#### Get the device handle through the resource handle

Function:int IC\_GetPUHandle(IN IC\_HANDLE hResource, OUT IC\_HANDLE \*hPU)

Parameter:IC\_HANDLE hResource [in] Sub-resource handle;

IC\_HANDLE \* hPU [in] Device handle;

Return: Please refer to the error code and description.

Description: This command is supported by all device subresources.Through this method, the handle of a device subresource can be obtained through the handle of the device itself.

#### Get the resource handle through the device handle

Function:int IC\_GetResourceHandle(IN IC\_HANDLE hPU, IN const char\* pszType, IN int nIndex, OUT IC\_HANDLE \*hResource)

Parameter:IC\_HANDLE hPU [in] Device handle;

const char \* pszType [in] The type of target subresource;

int nIndex [in] The index of the target subresource;

IC\_HANDLE \* hResource [out] Handle of subresource

Return: Please refer to the error code and description.

Description: The index of resources refers to the index between the devices when they have multiple resources at the same type. For example, if a device has 4 ways of input video, nIndex should substitute 3 to get the 4th way of resource (Index starting from 0).

#### Get the resource PUID

Function:int IC\_GetResourcePUID(IN IC\_HANDLE hResource, OUT IC\_STR szPUID)

Parameter:IC\_HANDLE hResource [in] Resource handle

IC\_STR szPUID [out] resource PUID

Return: Please refer to the error code and description.

Description: Through this method, obtain the unique identification of resources;

#### Get resource type

Function:int IC\_GetResourceType(IN IC\_HANDLE hResource, OUT IC\_STR stipe)

Parameter:IC\_HANDLE hResource [in] Resource handle

IC\_STR szType [out] The type of the resource. For the value range, please check the resource type.

Return: Please refer to the error code and description.

Description: none;

#### Get the resource index

Function:IC\_GetResourceIndex(IN IC\_HANDLE hResource, OUT int \*nIndex)

Parameter:IC\_HANDLE hResource [in] Resource handle

int \* nIndex [out] The index of the resource.

Return: Please refer to the error code and description.

Description: none;

#### Get the resource usability

Function:int IC\_GetResoureUsable(IN IC\_HANDLE hResource, OUT int \*nUsable)

Parameter:IC\_HANDLE hResource [in] Resource handle

int \* nUsable [out] Whether resources is usable or not, 1 is true, 0 is false.

Return: Please refer to the error code and description.

Description: This method is generally used to determine whether the resource is online;

#### Get resource name

Function:int IC\_GetResourceName(IN IC\_HANDLE hResource, OUT IC\_STR szName)

Parameter:IC\_HANDLE hResource [in] Resource handle

IC\_STR szName [out] Resource name.

Return: Please refer to the error code and description.

Description: none;

#### Set resource name

Function:int IC\_SetResourceName(IN IC\_HANDLE hResource, IN const char\* pszName)

Parameter:IC\_HANDLE hResource [in] Resource handle

IC\_STR pszName [in] resource name.

Return: Please refer to the error code and description.

Description: none;

#### Get resource description

Function:int IC\_GetResourceDescription(IN IC\_HANDLE hResource, OUT IC\_STR szDescription)

Parameter:IC\_HANDLE hResource [in] Resource handle

IC\_STR szDescription [out] Description of the resource.

Return: Please refer to the error code and description.

Description: none;

#### Set the resource description

Function:int IC\_SetResourceDescription(IN IC\_HANDLE hResource, IN const char\* szDescription)

Parameter:IC\_HANDLE hResource [in] Resource handle

IC\_STR szDescription [out] Description of the resource.

Return: Please refer to the error code and description.

Description: none;

#### Get the resource enablement

Function:int IC\_GetResoureEnable(IN IC\_HANDLE hResource, OUT int \*nEnable)

Parameter:IC\_HANDLE hResource [in] Resource handle

int \* nEnable [out] Whether resources is enabled or not, 1 is true, 0 is false.

Return: Please refer to the error code and description.

Description: Resource Enablement indicates whether the resource is currently online, but it is available to set whether the resource is functional or not.

#### Set the resource enablement

Function:int IC\_SetResourceEnable(IN IC\_HANDLE hResource, IN int nEnable)

Parameter:IC\_HANDLE hResource [in] Resource handle

int nEnable [in] Whether resources is enabled or not, 1 is true, 0 is false.

Return: Please refer to the error code and description.

Description: none;

### Site resources

#### Attribute access

##### Get Vendor ID

Function:IC\_GetProducerID(IN IC\_HANDLE hResource, OUT IC\_STR szProducerID)

Parameter:IC\_HANDLE hResource [in] Handle of device resource;

IC\_STR szProducerID [out] Vendor ID, like 00001, up to 63 characters;

Return: Please refer to the error code and description.

Description: none.

#### Control commands

##### Restart the video server

Function:int IC\_Reboot(IN IC\_HANDLE hResource)

Parameter:IC\_HANDLE hResource [in] Handle of device resource;

Return: Please refer to the error code and description.

Description: none.

### Enter the video resource

#### Attribute access

##### Get the camera status

Function:int IC\_GetCameraStatus(IN IC\_HANDLE hResource, OUT int \*nCameraStatus)

Parameter:IC\_HANDLE hResource [in] Input video resource handle;

int \* nCameraStatus [out] camera's current status, 0 for video available, 1 for video unavailable;

Return: Please refer to the error code and description.

Description: none.

##### Get the brightness

Function:IC\_GetBrightness(IN IC\_HANDLE hResource, OUT int \*nBrightness)

Parameter:IC\_HANDLE hResource [in] Input video resource handle;

int \* nBrightness [out] The brightness of the video;

Return: Please refer to the error code and description.

Description: Value range 0-100.

##### Get the contrast

Function:int IC\_GetContrast(IN IC\_HANDLE hResource, OUT int \*nContrast)

Parameter:IC\_HANDLE hResource [in] Input video resource handle;

int \* nContrast [out] Video contrast;

Return: Please refer to the error code and description.

Description: Value range 0-100.

##### Get the hue

Function:int IC\_GetHue(IN IC\_HANDLE hResource, OUT int \*nHue)

Parameter:IC\_HANDLE hResource [in] Input video resource handle;

int \* nHue [out] The hue of the video;

Return: Please refer to the error code and description.

Description: Value range 0-100.

##### Get the saturation

Function:int IC\_GetSaturation(IN IC\_HANDLE hResource, OUT int \*nSaturation)

Parameter:IC\_HANDLE hResource [in] Input video resource handle;

int \* nSaturation [out] Video saturation;

Return: Please refer to the error code and description.

Description: Value range 0-100.

##### Get target code rate

Function: int IC\_GetBitRate(IN IC\_HANDLE hResource, IN int nStreamID, OUT int \*nBitRate)

Parameter: IC\_HANDLE hResource [in] Input video resource handle;

int nStreamID [in] stream index;

int \* nBitRate [out] Target code rate;

Return: Please refer to the error code and description

Description: under VBR mode, the code rate does not exceed this value

under CBR mode, the code rate is constant at this value

Value range: 16Kbps ~ 10Mbps.

##### Get the target frame rate

Function: int IC\_GetFrameRate(IN IC\_HANDLE hResource, IN int nStreamID, OUT IC\_STR szFrameRateType, OUT int \*nFrameRate)

Parameter: IC\_HANDLE hResource [in] Input video resource handle;

int nStreamID [in] stream index;

IC\_STR szFrameRateType [out] Target frame rate type;

int \* nFrameRate [out] The target frame rate;

Return: Please refer to the error code and description

Description: The device will try to encode at this frame rate, and its rate is not higher than this frame rate.

##### Get the target clarity

Function: int IC\_GetImageDefinition(IN IC\_HANDLE hResource, IN int nStreamID, OUT int \*nImageDefinition)

Parameter: IC\_HANDLE hResource [in] Input video resource handle;

int nStreamID [in] stream index;

int \* nImageDefinition [out] the target definition, 0 ~ 100,100 for the clearest, 0 for the most fuzzy,;

Return: Please refer to the error code and description

Description: Under VBR, the device will try its best to encode at this definition, and its definition will not be higher than this definition, but it will be invalid under CBR.

##### Gets whether to add time

Function: int IC\_GetAddTime(IN IC\_HANDLE hResource, IN int nStreamID, OUT int \*nAddTime)

Parameter: IC\_HANDLE hResource [in] Input video resource handle;

int nStreamID [in] stream index;

int \* nAddTime [out] whether to add the time, 1 means that the time is added, while 0 means the time is not added;

Return: Please refer to the error code and description

Description: none.

##### Get added text

Function:int IC\_GetTextAdd(IN IC\_HANDLE hResource, OUT IC\_STR szTextAdd)

Parameter:IC\_HANDLE hResource [in] Input video resource handle;

IC\_STR szTextAdd [out] add text

Return: Please refer to the error code and description.

Description: none.

#### Property setup

##### Set the brightness

Function:int IC\_SetBrightness(IN IC\_HANDLE hResource, IN int nBrightness)

Parameter:IC\_HANDLE hResource [in] Input video resource handle;

int nBrightness [in] The brightness to be set.

Return: Please refer to the error code and description.

Description: Value range: 0-100, in case the client is realized, if a user frequently changes control attribute by moving a scroll bar, it is recommended to preview and then set the attributes .

##### Set the contrast

Function:int IC\_SetContrast(IN IC\_HANDLE hResource, IN int nContrast)

Parameter:IC\_HANDLE hResource [in] Input video resource handle;

int nContrast [in] Video contrast;

Return: Please refer to the error code and description.

Description: Value range: 0-100, in case the client is realized, if a user frequently changes control attribute by moving a scroll bar, it is recommended to preview and then set the attributes .

##### Set the hue

Function:int IC\_SetHue(IN IC\_HANDLE hResource, IN int nHue)

Parameter:IC\_HANDLE hResource [in] Input video resource handle;

int nHue [in] The hue of the video;

Return: Please refer to the error code and description.

Description: Value range: 0-100, in case the client is realized, if a user frequently changes control attribute by moving a scroll bar, it is recommended to preview and then set the attributes .

##### Set the saturation

Function:int IC\_SetSaturation(IN IC\_HANDLE hResource, IN int nSaturation)

Parameter:IC\_HANDLE hResource [in] Input video resource handle;

int nSaturation [in] Saturation of the video;

Return: Please refer to the error code and description.

Description: Value range: 0-100, in case the client is realized, if a user frequently changes control attribute by moving a scroll bar, it is recommended to preview and then set the attributes .

##### Set the encoding resolution

Function: int IC\_SetResolution(IN IC\_HANDLE hResource, IN int nStreamID, IN const char\* pszResolution)

Parameter: IC\_HANDLE hResource [in] Input video resource handle;

int nStreamID [in] stream index;

const char \* pszResolution [in] encoding resolution;

Return: Please refer to the error code and description

Description: none.

##### Set the target code rate

Function: int IC\_SetBitRate(IN IC\_HANDLE hResource, IN int nStreamID, IN int nBitRate)

Parameter: IC\_HANDLE hResource [in] Input video resource handle;

int nStreamID [in] stream index;

int nBitRate [in] Target code rate;

Return: Please refer to the error code and description

Description: Value range: 16Kbps ~ 10Mbps.

##### Set the target frame rate

Function: int IC\_SetFrameRate(IN IC\_HANDLE hResource, IN int nStreamID, IN const char\* pszFrameRateType, IN int nFrameRate)

Parameter: IC\_HANDLE hResource [in] Input video resource handle;

int nStreamID [in] stream index;

const char \* pszFrameRateType [in] The target frame rate type;

int nFrameRate [in] The target frame rate;

Return: Please refer to the error code and description

Description: The device will try to encode at this frame rate, and its rate is not higher than this frame rate.

##### Set the target definition

Function: int IC\_SetImageDefinition(IN IC\_HANDLE hResource, IN int nStreamID, IN int nImageDefinition)

Parameter: IC\_HANDLE hResource [in] Input video resource handle;

int nStreamID [in] stream index;

int nImageDefinition [in] Target definition;

Return: Please refer to the error code and description

Description: Under VBR, the device will try its best to encode at this definition, and its definition will not be higher than this definition, but it will be invalid under CBR.

##### Set whether to add time

Function: int IC\_SetAddTime(IN IC\_HANDLE hResource, IN int nStreamID, IN int nAddTime)

Parameter: IC\_HANDLE hResource [in] Input video resource handle;

int nStreamID [in] stream index;

int nAddTime [in] whether to add time;

Return: Please refer to the error code and description

Description: 1 means that the time is added, while 0 means the time is not added;

##### Set whether to add text

Function: IC\_SetAddText(IN IC\_HANDLE hResource, IN int nStreamID, IN int nAddText)

Parameter: IC\_HANDLE hResource [in] Input video resource handle;

int nStreamID [in] stream index;

Set whether to add text

Return: Please refer to the error code and description

Description: 1 means that the text is added, while 0 means the text is not added;

##### Set added text content

Function:int IC\_SetTextAdd(IN IC\_HANDLE hResource, IN const char\* pszTextAdd)

Parameter:IC\_HANDLE hResource [in] Input video resource handle;

const char \* pszTextAdd [in] the contents of text added;

Return: Please refer to the error code and description.

Description: The string to be added, up to 63 characters, and it is needed to be automatically cut off if not displayed as a whole

### Serial port resources

#### Attribute access

##### Get the current baud rate

Function:int IC\_GetBaudRate(IN IC\_HANDLE hResource, OUT int \*nBaudRate)

Parameter:IC\_HANDLE hResource [in] Handle of serial port resources;

int \* nBaudRate [out] baud rate;

Return: Please refer to the error code and description.

Description: The value range covers all supported baud rates.

##### Get the current data bit

Function:int IC\_GetDataBits(IN IC\_HANDLE hResource, OUT int \*nDataBits)

Parameter:IC\_HANDLE hResource [in] Handle of serial port resources;

int \* nDataBits [out] data bits;

Return: Please refer to the error code and description.

Description: The value range covers all supported data bits.

##### Get the current parity bit

Function:int IC\_GetParity(IN IC\_HANDLE hResource, OUT IC\_STR szParity)

Parameter:IC\_HANDLE hResource [in] Handle of serial port resources;

IC\_STR szParity [out] Name of the parity bit;

Return: Please refer to the error code and description.

Description: The value range covers all supported parity bits.

##### Get the current stop bit

Function:int IC\_GetStopBits(IN IC\_HANDLE hResource, OUT int \*nStopBits)

Parameter:IC\_HANDLE hResource [in] Handle of serial port resources;

int \* nStopBits [out] Stop bit;

Return: Please refer to the error code and description.

Description: The value range covers all supported stop bits.

#### Attribute setup

##### Set the current baud rate

Function:int IC\_SetBaudRate(IN IC\_HANDLE hResource, IN int nBaudRate)

Parameter:IC\_HANDLE hResource [in] Handle of serial port resources;

int nBaudRate [in] baud rate;

Return: Please refer to the error code and description.

Description: The value range covers all supported baud rates.

##### Set the current data bit

Function:int IC\_SetDataBits(IN IC\_HANDLE hResource, IN int nDataBits)

Parameter:IC\_HANDLE hResource [in] Handle of serial port resources;

int nDataBits [in] data bits;

Return: Please refer to the error code and description.

Description: The value range covers all supported data bits.

##### Set the current parity bit

Function:int IC\_SetParity(IN IC\_HANDLE hResource, IN const char\* pszParity)

Parameter:IC\_HANDLE hResource [in] Handle of serial port resources;

const char \* pszParity [in] The name of the check digit;

Return: Please refer to the error code and description.

Description: The value range covers all supported parity bits.

##### Set the stop bit

Function:int IC\_SetStopBits(IN IC\_HANDLE hResource, IN int nStopBits)

Parameter:IC\_HANDLE hResource [in] Handle of serial port resources;

int nStopBits [in] Stop bit;

Return: Please refer to the error code and description.

Description: The value range covers all supported stop bits.

#### Control commands

##### Send data

Function:int IC\_WriteData(IN IC\_HANDLE hResource, IN char \*pData, IN int nLen)

Parameter:IC\_HANDLE hResource [in] Handle of serial port resources;

char \* pData [in] send buffer pointer;

USHORT usLength [in] unsigned short, the length of the data sent;

Return: Please refer to the error code and description.

Description: Video server outputs data to the serial port.

### PTZ resources

#### Attribute access

##### Get the PTZ movement speed

Function:int IC\_GetSpeed(IN IC\_HANDLE hResource, OUT int \*nSpeed)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

int \* nSpeed ​​[out] PTZ speed of movement;

Return: Please refer to the error code and description.

Description: The value range is 0-100, representing the speed at which the PTZ moves.

#### Attribute setup

##### Set the PTZ movement speed

Function:int IC\_SetSpeed(IN IC\_HANDLE hResource, IN int nSpeed)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

int nSpeed ​​[in] PTZ speed of movement;

Return: Please refer to the error code and description.

Description: The value range is 0-100, representing the speed at which the PTZ moves.

#### Control commands

##### Turn left

Function:int IC\_StartTurnLeft(IN IC\_HANDLE hResource, IN int nDegree)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

int nDegree [in] turning angle, 0 means remaining turning status;

Return: Please refer to the error code and description.

Description: none.

##### Turn right

Function:int IC\_StartTurnRight(IN IC\_HANDLE hResource, IN int nDegree)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

int nDegree [in] turning angle, 0 means remaining turning status;

Return: Please refer to the error code and description.

Description: none.

##### Turn up

Function:int IC\_StartTurnUp(IN IC\_HANDLE hResource, IN int nDegree)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

int nDegree [in] turning angle, 0 means remaining turning status;

Return: Please refer to the error code and description.

Description: none.

##### Turn down

Function:int IC\_StartTurnDown(IN IC\_HANDLE hResource, IN int nDegree)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

int nDegree [in] turning angle, 0 means remaining turning status;

Return: Please refer to the error code and description.

Description: none.

##### Stop turning

Function:int IC\_StopTurn(IN IC\_HANDLE hResource)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

Return: Please refer to the error code and description.

Description: none.

##### Increase the aperture

Function:int IC\_AugmentAperture(IN IC\_HANDLE hResource)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

Return: Please refer to the error code and description.

Description: none.

##### Reduce the aperture

Function:int IC\_MinishAperture(IN IC\_HANDLE hResource)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

Return: Please refer to the error code and description.

Description: none.

##### Stop the aperture zoom-out

Function:int IC\_StopApertureZoom(IN IC\_HANDLE hResource)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

Return: Please refer to the error code and description.

Description: none.

##### Make the focus far away

Function:int IC\_MakeFocusFar(IN IC\_HANDLE hResource)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

Return: Please refer to the error code and description.

Description: none.

##### Make the focus close

Function:int IC\_MakeFocusNear(IN IC\_HANDLE hResource)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

Return: Please refer to the error code and description.

Description: none.

##### Stop the focus adjustment

Function:int IC\_StopFocusMove(IN IC\_HANDLE hResource)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

Return: Please refer to the error code and description.

Description: none.

##### Zoom out the picture

Function:int IC\_ZoomOutPicture(IN IC\_HANDLE hResource)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

Return: Please refer to the error code and description.

Description: none.

##### Zoom in the picture

Function:int IC\_ZoomInPicture(IN IC\_HANDLE hResource)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

Return: Please refer to the error code and description.

Description: none.

##### Stop the picture zoom-out

Function:int IC\_StopPictureZoom(IN IC\_HANDLE hResource)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

Return: Please refer to the error code and description.

Description: none.

##### Move to the preset position

Function:int IC\_MoveToPresetPos(IN IC\_HANDLE hResource, IN int nPresetPos)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

int nPresetPos [in] Preset position number;

Return: Please refer to the error code and description.

Description: none.

##### Set the preset position

Function:int IC\_SetPresetPos(IN IC\_HANDLE hResource, IN int nPresetPos, IN const char\* pszName)

Parameter:IC\_HANDLE hResource [in] Handle of PTZ resource;

int nPresetPos [in] Preset position number;

const char \* pszName [in] The name of the preset position, corresponding to the option "preset position name", up to 63 characters;

Return: Please refer to the error code and description.

Description: Set the current position of PTZ to a preset position.

### Alarm output resource

#### Attribute access

##### Get whether output is connected

Function:int IC\_GetConnectStatus(IN IC\_HANDLE hResource, OUT int \*nConnectStatus)

Parameter:IC\_HANDLE hResource [in] Input video resource handle;

int \* nConnectStatus [out] Output is connected;

Return: Please refer to the error code and description.

Description:nConnectStarus:1 for connected, 0 for disconnected.

#### Control commands

##### Set whether output is connected

Function:int IC\_SetConnectStatus(IN IC\_HANDLE hResource, IN int nConnectStatus)

Parameter:IC\_HANDLE hResource [in] Input video resource handle;

int nConnectStatus [in] Output is on;

Return: Please refer to the error code and description.

Description:nConnectStarus:1 for connected, 0 for disconnected.

### Leading end storage resources

#### Control commands

##### Query video and picture files

Function: int IC\_QueryFile(IN IC\_HANDLE hResource, IN int nIndex, IN int nBeginTime, IN int nEndTime, IN int nStreamID, OUT IC\_FILE\_INFO \*rfiaFileInfos, IN OUT int \*nCount, IN int nOffset)

Parameter: IC\_HANDLE hResource [in] leading edge storage resource handle;

int nIndex [in] Index of the video resource to be queried;

int nBeginTime [in] beginning time;

int nEndTime [in] End time;

int nStreamID [in] stream index;

IC\_FILE\_INFO \* rfiaFileInfos [in] An array of file information-contained structures;

int \* nCount [in, out] the size of the incoming data in array while being input, the number of data actually get while being output.

int nOffset [in] Query offset;

Return: Please refer to the error code and description

Description: none.

##### Query historical GPS data

Function:int IC\_QueryGPSData(IN IC\_HANDLE hResource, IN int nBeginTime, IN int nEndTime, OUT IC\_GPS\_INFO \*giaGPSInfos, IN OUT int \*nCount, IN int nOffset)

Parameter:IC\_HANDLE hResource [in] leading edge storage resource handle;

int nBeginTime [in] beginning time;

int nEndTime [in] End time;

IC\_GPS\_INFO \* giaGPSInfos [out] Pointer consisting of structures storing GPS information

int \* nCount [in, out] the size of the incoming GPS in array while being input, the number of GPS actually get while being output.

int nOffset [in] offset;

Return: Please refer to the error code and description.

Description: the consistence between nCount input and output indicates that the data has not been fully obtained, then it is needed to obtain again, nOffset should be equal to the total number of data obtained previously.

##### Download files (including pictures)

Function:int IC\_DownLoadFile(IN IC\_HANDLE hResource, IN const char\* pszFile, IN int nOffset, IN int nLength, OUT IC\_HANDLE \*hStream)

Parameter:IC\_HANDLE hResource [in] leading edge storage resource handle;

const char \* pszFile [in] file path + file name;

Int nLength [in] The length of downloaded, in bytes, -1 indicates downloading until end of the file;

IC\_HANDLE \* hStream [out] Download stream handle;

Return: Please refer to the error code and description.

Description: Users need to use hStream to read the stream data.

##### On-demand documents

Function:int IC\_VODFile(IN IC\_HANDLE hResource, IN const char\* pszFile, IN int nSpeed, IN int nDirection, IN int nStartTime, OUT int \*nTotalTime, OUT IC\_HANDLE \*hStream)

Parameter:IC\_HANDLE hResource [in] leading edge storage resource handle;

const char \* pszFile [in] file path + file name;

int nSpeed ​​[in] playing speed;

int nDirection [in] playing direction;

int nStartTime [in] start time;

int \* nTotalTime [out] The total file time;

IC\_HANDLE \* hStream [out] on-demand stream handle;

Return: Please refer to the error code and description.

Description: Users need to use hStream to read the stream data.

### Platform storage resources

#### Control commands

##### Manually start the storage

Function:int IC\_CSSManualStart(IC\_HANDLE hSession, const char \*pszReason, int nDuration, int nType, int nReserveDay, int nFileLength, const char \*pszPUID, int nIndex, int nStreamID, IC\_STR szID)

Parameter:IC\_HANDLE hSession [in] Platform handle;

const char \* pszReason [in] Storage reason;

int nDuration [in] Duration;

int nType [in] storage type time, 0 for video, 1 for capture;

int nReserveDay [in] File retention days;

int nFileLength [in] file length;

const char \* pszPUID [in] storage device ID;

int nIndex [in] Video number;

IC\_STR szID [out] storage ID;

Return: Please refer to the error code and description.

Description: none.

##### Manually stop the storage

Function:int IC\_CSSManualStop(IC\_HANDLE hSession, const char \*pszID)

Parameter:IC\_HANDLE hSession [in] Platform handle;

const char \* pszID [in] storage ID;

Return: Please refer to the error code and description.

Description: none.

##### Query video / picture files

Function:int IC\_CSSQueryFile(IC\_HANDLE hSession, int nBeginTime, int nEndTime, int nType, const IC\_STR \*pReason, int nReasonCnt, const IC\_HANDLE \*pRes, int nResCnt, int nOffset, IC\_FILE\_INFO \*rfiaFileInfos, int \*nCount)

Parameter:IC\_HANDLE hSession [in] Platform handle;

int nBeginTime [in] beginning time;

int nEndTime [in] End time;

int nType [in] Query type, 0 for video, 1 for capture;

const IC\_STR \* pReason [in] Storage reason array;

int nReasonCnt [in] number of storage reasons;

const IC\_HANDLE \* pRes [in] query resource arrays;

int nResCnt t [in] Number of resources;

int nOffset [int] Query offset;

IC\_FILE\_INFO \* rfiaFileInfos [in] An array of file information-contained structures;

int \* nCount [in, out] the size of the incoming data in array while being input, the number of data actually get while being output.

Return: Please refer to the error code and description.

Description: none.

##### Delete video / picture file

Function:int IC\_CSSDeleteFile(IC\_HANDLE hSession, int nType, const IC\_STR \*szaFiles, const IC\_STR \*szaID, int nCount)

Parameter:IC\_HANDLE hSession [in] Platform handle;

int nType [in] type, 0 for video, 1 for capture;

const IC\_STR \* szaFiles [in] file path + file name;

const IC\_STR \* szaID [in] Store ID array;

int nCount [in] Number of files;

Return: Please refer to the error code and description.

Description: none.

##### Download files (including pictures)

Function:int IC\_CSSDownLoadFile(IC\_HANDLE hSession, const char \*pszID, const char \*pszFile, int nOffset, int nLength, IC\_HANDLE \*hStream)

Parameter:IC\_HANDLE hSession [in] Platform handle;

const char \* pszID [in] storage ID;

const char \* pszFile [in] file path + file name;

int nOffset [in] Download offset;

int nLength [in] The length of the download;

IC\_HANDLE \* hStream [out] Download stream handle;

Return: Please refer to the error code and description.

Description: Users need to use hStream to read the stream data.

##### On-demand documents

Function:int IC\_CSSVODFile(IC\_HANDLE hSession, const char \*pszID, const char \*pszFile, int nSpeed, int nDirection, int nStartTime, int nDuration, IC\_HANDLE \*hStream)

Parameter:IC\_HANDLE hSession [in] Platform handle;

const char \* pszID [in] storage ID;

const char \* pszFile [in] file path + file name;

int nSpeed ​​[in] playing speed;

int nDirection [in] playing direction;

int nStartTime [in] start time;

int nDuration [in] playing time;

IC\_HANDLE \* hStream [out] on-demand stream handle;

Return: Please refer to the error code and description.

Description: Users need to use hStream to read the stream data.

##### On-demand time

Function: int IC\_CSSVODTime(IC\_HANDLE hSession, IC\_HANDLE hRes, int nBeginTime, int nEndTime, int nRate, IC\_STR szVodID, IC\_HANDLE \*hStream)

Parameter: IC\_HANDLE hSession [in] Platform handle;

const char \* pszID [in] storage ID;

IC\_HANDLE hRes [in] on-demand video resource handle;

int nBeginTime [in] start time;

int nEndTime [in] End time;

int nRate [in] transmission rate, at a rate of 2 n-th-power,

IC\_STR szVodID [out] on-demand ID for on-demand control;

IC\_HANDLE \* hStream [out] on-demand stream handle;

Return: Please refer to the error code and description

Description: Users need to use hStream to read the stream data.

##### On-demand pause

Function: int IC\_CSSVODPause(IC\_HANDLE hSession, const char \*pszVodID)

Parameter: IC\_HANDLE hSession [in] Platform handle;

const char \* pszVodID [in] on-demand ID, IC\_CSSVODFile or IC\_CSSVODTime returned;

Return: Please refer to the error code and description

Description: none.

##### On-demand resume

Function: int IC\_CSSVODResume (IC\_HANDLE hSession, const char \*pszVodID)

Parameter: IC\_HANDLE hSession [in] Platform handle;

const char \* pszVodID [in] on-demand ID, IC\_CSSVODFile or IC\_CSSVODTime returned;

Return: Please refer to the error code and description

Description: none.

##### On-demand change playing location

Function: int IC\_CSSVODChangePosition(IC\_HANDLE hSession, const char \*pszVodID, int nPosition)

Parameter: IC\_HANDLE hSession [in] Platform handle;

const char \* pszVodID [in] on-demand ID, IC\_CSSVODFile or IC\_CSSVODTime returned;

int nPosition [in] Offset relative to on-demand start time, in seconds;

Return: Please refer to the error code and description

Description: none.

##### On-demand to change the transmission rate

Function: int IC\_CSSVODChangeRate(IC\_HANDLE hSession, const char \*pszVodID, int nRate)

Parameter: IC\_HANDLE hSession [in] Platform handle;

const char \* pszVodID [in] on-demand ID, IC\_CSSVODFile or IC\_CSSVODTime returned;

int nRate [in] transmission rate, at a rate of 2n-th-power,

Return: Please refer to the error code and description

Description: none.

##### Upload rate

Function: int IC\_CSSUploadFile(IC\_HANDLE hSession, const char \*pszFilename, int nTime, int nReserveDay, const IC\_STR \*szaReason, int nCount, IC\_STR szID, int \*nLength, IC\_HANDLE \*hStream)

Parameter: IC\_HANDLE hSession [in] Platform handle;

const char \* pszVodID [in] on-demand ID, IC\_CSSVODFile or IC\_CSSVODTime returned;

int nRate [in] transmission rate, at a rate of 2n-th-power,

Return: Please refer to the error code and description

Description: none.

##### Upload files

Function: int IC\_CSSUploadFile(IC\_HANDLE hSession, const char \*pszFilename, int nTime, int nReserveDay, const IC\_STR \*szaReason, int nCount, IC\_STR szID, int \*nLength, IC\_HANDLE \*hStream)

Parameter: IC\_HANDLE hSession [in] Platform handle;

const char \* pszFilename [in] file name;

int nTime [in] file time;

int nReserveDay [in] file save time;

const IC\_STR \* szaReason [in] reason array;

int nCount [in] number of reasons;

IC\_STR szID [in, out] Upload ID, if the channel is disconnected and reconnected, the returned data during the previous upload should be input;

int \* nLength [out] Number of bytes uploaded

IC\_HANDLE \* hStream [out] Upload handle;

Return: Please refer to the error code and description

Description: Users need to read documents manually, and use hStream to send data.

##### Upload video

Function: IC\_CSSUploadRecord(IC\_HANDLE hSession, const char \*pszPUID, int nIndex, int nStreamID, const char \*pszReason, int nReserveDay, int nFileLength, IC\_HANDLE \*hStream)

Parameter: IC\_HANDLE hSession [in] Platform handle;

const char \* pszPUID [in] Upload device;

int nIndex [in] Channel index;

int nStreamID [in] stream index;

const char \* pszReason [in] Upload reason;

int nReserveDay [in] File retention days;

IC\_HANDLE \* hStream [out] Upload handle;

Return: Please refer to the error code and description

Description: Users need to read real-time data, and use hStream to send data.