

# ARTRAY Camera / Capture Module Software Developer Kit

Dynamic Link Library for Windows2000,XP,Vista,7  
Functions Manual Version 1.3.0.0-16

Artray Co., Ltd.

## Contents of DLL function

DLL Initializing .....	4
ArtCam_GetDllVersion .....	4
ArtCam_GetLastError .....	6
ArtCam_Initialize .....	7
ArtCam_Release .....	7
Image capture .....	8
ArtCam_Preview .....	8
ArtCam_Record .....	8
ArtCam_CallBackPreview .....	9
ArtCam_SnapShot .....	10
ArtCam_Capture .....	11
ArtCam_Close .....	11
ArtCam_Trigger .....	12
WM_GRAPHPAINT .....	13
WM_ERROR .....	14
ArtCam_StartPreview .....	15
ArtCam_StopPreview .....	15
ArtCam_SaveImage .....	16
ArtCam_GetImage .....	17
Setting dialog .....	18
ArtCam_SetCameraDlg .....	18
ArtCam_SetImageDlg .....	18
ArtCam_SetAnalogDlg .....	19
Camera setting .....	20
ArtCam_SetPreviewWindow .....	20
ArtCam_SetCaptureWindow .....	21
ArtCam_SetCaptureWindowEx .....	22
ArtCam_GetCaptureWindowEx .....	23
ArtCam_SetColorMode .....	24
ArtCam_GetColorMode .....	24
ArtCam_SetCrossbar .....	25
ArtCam_SetDeviceNumber .....	26
ArtCam_GetDeviceName .....	26
ArtCam_EnumDevice .....	27
ArtCam_SetCameraType .....	28
ArtCam_GetCameraType .....	28
ArtCam_Width .....	29
ArtCam_Height .....	29
ArtCam_Fps .....	30
ArtCam_GetCameraInfo .....	30
ArtCam_SetIOPort .....	31
ArtCam_GetIOPort .....	31
ArtCam_SetSubSample .....	32
ArtCam_GetSubSample .....	32

ArtCam_SetWaitTime .....	33
ArtCam_GetWaitTime.....	33
ArtCam_SetMirrorV .....	34
ArtCam_GetMirrorV .....	34
ArtCam_SetMirrorH .....	35
ArtCam_GetMirrorH .....	35
ArtCam_SetHalfClock.....	36
ArtCam_GetHalfClock .....	36
ArtCam_SetAutoIris .....	37
ArtCam_GetAutoIris.....	37
ArtCam_SetSamplingRate .....	38
ArtCam_GetSamplingRate.....	38
ArtCam_GetVideoFormat.....	39
ArtCam_WriteSromID.....	40
ArtCam_ReadSromID.....	40
ArtCam_WriteRegistor .....	41
ArtCam_ReadRegister .....	41
ArtCam_SetFilterValue.....	42
ArtCam_GetFilterValue.....	42
ArtCam_Set*** .....	43
ArtCam_Get*** .....	43
ArtCam_GetRealExposureTime.....	44
 Image Filter Setting Possible Value .....	 45
For All Cameras.....	45
ARTCNVII .....	47
Grayscale Filter Setting Possible Value (DLLVer.1280 or Up) .....	48

## ArtCam\_GetDllVersion

**Definition:** **DWORD** ArtCam\_GetDllVersion(*void*)

**Function:** Obtain library's version

**Argument:** None

**Function Detail:**

Obtain version and type of DLL

Among returned DWORD (32 bits), DLL type is stored in upper 16 bits while DLL version is stored in lower 16 bits.

Before you use library, check the DLL versions you installed. So as SDK .

The version is obtained as 4 places integral number.

If the version is 1.278, 1278 is stored for lower 16 bits.

DLL types are as below:

CODE	DEVICE TYPE
ARTCAM_CAMERATYPE_DS	DirectShowCamera
ARTCAM_CAMERATYPE_USTC	ARTUST
ARTCAM_CAMERATYPE_CNV	ARTCNV
ARTCAM_CAMERATYPE_130MI	ARTCAM-130MI
ARTCAM_CAMERATYPE_200MI	ARTCAM-200MI
ARTCAM_CAMERATYPE_300MI	ARTCAM-300MI
ARTCAM_CAMERATYPE_150P	ARTCAM-150P
ARTCAM_CAMERATYPE_320P	ARTCAM-320P
ARTCAM_CAMERATYPE_200SH	ARTCAM-200SH
ARTCAM_CAMERATYPE_098	ARTCAM-098
ARTCAM_CAMERATYPE_036MI	ARTCAM-036MI
ARTCAM_CAMERATYPE_500P	ARTCAM-500P
ARTCAM_CAMERATYPE_150P2	ARTCAM-150PII
ARTCAM_CAMERATYPE_036MIST	ARTCAM-036MI-TWIN
ARTCAM_CAMERATYPE_500MI	ARTCAM-500MI
ARTCAM_CAMERATYPE_T111	ARTCAM-T111
ARTCAM_CAMERATYPE_T112	ARTCAM-T112
ARTCAM_CAMERATYPE_150P3	ARTCAM-150PIII
ARTCAM_CAMERATYPE_130MI_MOUT	ARTCAM-130MI-MOUT
ARTCAM_CAMERATYPE_150P3_MOUT	ARTCAM-150PIII-MOUT
ARTCAM_CAMERATYPE_267KY	ARTCAM-267KY
ARTCAM_CAMERATYPE_274KY	ARTCAM-274KY
ARTCAM_CAMERATYPE_625KY	ARTCAM-625KY
ARTCAM_CAMERATYPE_V135MI	ARTCAM-V135MI
ARTCAM_CAMERATYPE_445KY	ARTCAM-445KY
ARTCAM_CAMERATYPE_098II	ARTCAM-098II
ARTCAM_CAMERATYPE_MV413	ARTCAM-MV413USB

ARTCAM_CAMERATYPE_OV210	ARTCAM-OV210
ARTCAM_CAMERATYPE_850SH	ARTCAM-850SH
ARTCAM_CAMERATYPE_1251SH	ARTCAM-1252SH
ARTCAM_CAMERATYPE_D131	ARTCAM-D131
ARTCAM_CAMERATYPE_900MI	ARTCAM-900MI
ARTCAM_CAMERATYPE_1000MI	ARTCAM-1000MI
ARTCAM_CAMERATYPE_500P2	ARTCAM-500P2
ARTCAM_CAMERATYPE_1000MI_HD2	ARTCAM-1000MI-HD2
ARTCAM_CAMERATYPE_006MAT	ARTCAM-006MAT
ARTCAM_CAMERATYPE_SATA	SATA Camera
ARTCAM_CAMERATYPE_036MI2_WOM	ARTCAM-036MI2-WOM
ARTCAM_CAMERATYPE_130MI_WOM	ARTCAM-130MI-WOM
ARTCAM_CAMERATYPE_300MI_WOM	ARTCAM-300MI-WOM
ARTCAM_CAMERATYPE_500MI_WOM	ARTCAM-500MI-WOM
ARTCAM_CAMERATYPE_1000MI_WOM	ARTCAM-1000MI-WOM
ARTCAM_CAMERATYPE_1400MI_WOM	ARTCAM-1400MI -WOM
ARTCAM_CAMERATYPE_IMX035_WOM	ARTCAM-035IMX-WOM
ARTCAM_CAMERATYPE_130HP_WOM	ARTCAM-130HP-WOM
ARTCAM_CAMERATYPE_150P5_WOM	ARTCAM-150P5-WOM
ARTCAM_CAMERATYPE_274KY_WOM	ARTCAM-274KY-WOM
ARTCAM_CAMERATYPE_285CX_WOM	ARTCAM-285CX-WOM
ARTCAM_CAMERATYPE_424KY_WOM	ARTCAM-424KY-WOM
ARTCAM_CAMERATYPE_445KY2_WOM	ARTCAM-445KY2-WOM
ARTCAM_CAMERATYPE_500P2_WOM	ARTCAM-500P2-WOM
ARTCAM_CAMERATYPE_655KY_WOM	ARTCAM-655KY-WOM
ARTCAM_CAMERATYPE_407UV_WOM	ARTCAM-407UV-WOM
ARTCAM_CAMERATYPE_USB3_500MI	ARTCAM-500MI-USB3
ARTCAM_CAMERATYPE_USB3_900MI	ARTCAM-900MI-USB3
ARTCAM_CAMERATYPE_USB3_1000MI	ARTCAM-1000MI-USB3
ARTCAM_CAMERATYPE_USB3_1400MI	ARTCAM-1400MI-USB3
ARTCAM_CAMERATYPE_USB3_267KY	ARTCAM-267KY-USB3
ARTCAM_CAMERATYPE_USB3_274KY	ARTCAM-274KY-USB3
ARTCAM_CAMERATYPE_USB3_424KY	ARTCAM-424KY-USB3
ARTCAM_CAMERATYPE_USB3_655KY	ARTCAM-655KY-USB3
ARTCAM_CAMERATYPE_USB3_810KAI	ARTCAM-810KAI-USB3
ARTCAM_CAMERATYPE_USB3_2900KAI	ARTCAM-2900KAI-USB3
ARTCAM_CAMERATYPE_USB3_1600KAI	ARTCAM-1600KAI-USB3
ARTCAM_CAMERATYPE_USB3_410KAI	ARTCAM-410KAI-USB3
ARTCAM_CAMERATYPE_USB3_100KAI	ARTCAM-100KAI-USB3
ARTCAM_CAMERATYPE_USB3_210KAI	ARTCAM-210KAI-USB3

## ArtCam\_GetLastError

**Definition:** LONG ArtCam\_GetLastError(**HACAM** *hACam*)

**Function:** Obtained error

**Argument:**

**HACAM**

*hACam*

Handle for distinguish cameras

### Function Detail:

When error occurs in return value of function, please call this function to obtain details of error.

Error is stored in stack type of data configuration.

Errors can be called in sequential order.

ERROR CODE	ERROR DETAIL
ARTCAMSDK_NOERROR	Normal
ARTCAMSDK_NOT_INITIALIZE	Not Initialized
ARTCAMSDK_DISABLEDDEVICE	Tray to access to unusable device
ARTCAMSDK_CREATETHREAD	Failureure to create a thread for image capture
ARTCAMSDK_CREATEWINDOW	Failureure to create a window
ARTCAMSDK_OUTOFMEMORY	Not enough memory for image transferring. Or Failureure to obtain memory
ARTCAMSDK_CAMERASET	Error at camera (device) setting
ARTCAMSDK_CAMERASIZE	Error at camera (device) size setting
ARTCAMSDK_CAPTURE	Failureure at image capture
ARTCAMSDK_PARAM	Wrong argument
ARTCAMSDK_DIRECTSHOW	DirectShow Initializing error
ARTCAMSDK_UNSUPPORTED	This function is not supported
ARTCAMSDK_UNKNOWN	Unidentified error
ARTCAMSDK_CAPTURELOST	Lost device
ARTCAMSDK_FILENOTFOUND	Cannot find specified file
ARTCAMSDK_FPGASET	Error at FPGA setting
ARTCAMSDK_TRANSIMAGEFAILED	Failure of image transferring

## ArtCam\_Initialize

Definition: **HACAM** ArtCam\_Initialize(**HWND** *hWnd*)

Function: Initialize library

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for distinguish cameras
--------------	--------------	--------------------------------

Function Detail:

Initialize library.

Call this function first when you use this library

Once this function is succeeded, handle for camera identification is obtained in return value.

On the other hand, if it is Failed, NULL or 0 is returned.

By setting window handle to *hWnd*, [WM\\_ERROR](#) is sent to window procedure when an error occurs.

Also whenever this function is called, the last parameter setting is read from registry.

Each parameter setting is saved under the below registry key.

(Some parameters are not saved)

*HKEY\_CURRENT\_USER\Software\Artray\ArtCam[MODEL NAME]Sdk*

## ArtCam\_Release

Definition: **BOOL** ArtCam\_Release(**HACAM** *hACam*)

Function: Release library

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for distinguish cameras
--------------	--------------	--------------------------------

Function Detail:

Release all plugged cameras, and initialize all data within class.

Call this function when you end application or stop operation of cameras.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

To display image again, call [ArtCam\\_Initialize](#).

Also whenever this function is called, the last parameter setting is read from registry.

(Some parameters are not saved)

### ArtCam\_Preview

Definition: **BOOL** ArtCam\_Preview(**HACAM** *hACam*)

Function: Display image

Argument: **HACAM** *hACam* Handle for camera identification

Function Detail:

Image display is controlled by SDK.

Call [ArtCam\\_Initialize](#) before using this function.

When this function succeeds, create a sub-window within the window specified by

[ArtCam\\_SetPreviewWindow](#). Image will be displayed in the sub-window.

If setting is not done by [ArtCam\\_SetPreviewWindow](#), new window will be created, and image will be displayed.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

### ArtCam\_Record

Definition: **BOOL** ArtCam\_Record(  
**HACAM** *hACam*, **LPCTSTR** *lpAviName*, **UINT** *RecTime*, **BOOL** *fShow*)

Function: Record to file

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for distinguish cameras
<b>LPCTSTR</b>	<i>lpAviName</i>	Name of file to be saved
<b>UINT</b>	<i>RecTime</i>	Recording time (milli-second)
		Continuous recording at 0
<b>BOOL</b>	<i>fShow</i>	Display image or not

Function Detail:

When *RecTime* is specified, recording will automatically end as time out. However, device will not be released, and therefore image will still be displayed.

If you like to execute some process at the end of recordings, you need to obtain timing using timer. Regarding *fShow*, hiding image will prevent loss of frames.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

Remark: This function is exclusively for ArtCamSdk.dll (Direct Show Camera)



## ArtCam\_CallBackPreview

Definition: **BOOL** ArtCam\_CallBackPreview(

**HACAM** *hACam*, **HWND** *hWnd*, **LPBYTE** *lpImage*, **LONG** *Size*, **BOOL** *TopDown*)

Function: Obtain image data while display live video

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>HWND</b>	<i>hWnd</i>	Window Handle for receiving message
<b>LPBYTE</b>	<i>lpImage</i>	Address of arrangement for receiving image data
<b>LONG</b>	<i>Size</i>	Arrangement length of lpImage
<b>BOOL</b>	<i>TopDown</i>	Determine whether image is up or down

Function Detail:

When hWnd is specified to window handle, [WM\\_GRAPHPAINT](#) is sent to specified window procedure.

When lpImage and Size are specified, image is copied to the alignment, which was specified at lpImage before [WM\\_GRAPHPAINT](#).

Image will not be copied unless the size of alignment is equal to or larger than size of image.

Do not insert address of temporary alignment to lpImage.

If bitmap prepared is DDB (top-down), specify Topdown as True.

If bitmap is DIB (bottom-up), specify Topdown as False.

Like [ArtCam\\_Preview](#), this function also has automatic display. Procedure for auto-display is same as that of [ArtCam\\_Preview](#).

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

\*1: It is relatively difficult to obtain message with VB. There may be error due to processing speed of VB. Although the function itself can be used, real-time processing by [WM\\_GRAPHPAINT](#) should be avoided.

(With the current sample, that procedure is removed, and timing of display is controlled by timer)

## ArtCam\_SnapShot

**Definition:** **BOOL** ArtCam\_SnapShot(  
    **HACAM** *hACam*, **LPBYTE** *lpImage*, **LONG** *Size*, **BOOL** *TopDown*)

**Function:** Obtain image of camera only once

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LPBYTE</b>	<i>lpImage</i>	Address of arrangement for receiving image data
<b>LONG</b>	<i>Size</i>	Arrangement length of lpImage
<b>BOOL</b>	<i>TopDown</i>	Determination of ups and down of image

### **Function Detail:**

Obtained only 1 image from a camera by soft trigger.

When function succeeds, obtained data is stored in lpImage.

Image will not be obtained unless the size of alignment is equal to or larger than size of image.

If bitmap prepared is DDB (top-down), specify Topdown as True. If bitmap is DIB (bottom-up), specify Topdown as False.

While [ArtCam\\_GetImage](#) captures a frame in preview mode, ArtCam\_SnapShot captures a frame in non-preview mode.

This function will Failure if preview is displayed with other functions such as [ArtCam\\_Preview](#)

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

\*1: Strobo signal is sent from sensor to camera's BNC pin type if you call the function by some strobo camera (like as ARTCAM-300MI-STR2).

Please refer the manual for more detail of signal timing.

There is not this function for normal camera.

## ArtCam\_Capture

Definition: **BOOL** ArtCam\_Capture(**HACAM** hACam)

Function: Initialize camera for continuous snapshot

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
--------------	--------------	----------------------------------

Function Detail:

Initialize camera to use [ArtCam\\_SnapShot](#) continuously.

Normally when [ArtCam\\_SnapShot](#) is used, procedure proceeds as following:

Initialize - Obtain - Release

However, if you initialize beforehand with this function, the process of "Initialize" and "Release" will be ignored when [ArtCam\\_SnapShot](#) is called. Hence the image can be obtained with high-speed.

To stop [ArtCam\\_SnapShot](#) and release camera, call [ArtCam\\_Close](#).

The main flow is as following:

Initialize

[ArtCam\\_Capture](#)

Can be used unlimitedly

[ArtCam\\_SnapShot](#)

Release

[ArtCam\\_Close](#)

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

## ArtCam\_Close

Definition: **BOOL** ArtCam\_Close(**HACAM** hACam)

Function: Release device

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
--------------	--------------	----------------------------------

Function Detail :

Stop preview screen, and release device. Use this function to release device when you obtain images with the following functions.

[ArtCam\\_Preview](#)

[ArtCam\\_Record](#)

[ArtCam\\_CallBackPreview](#)

[ArtCam\\_Capture](#)

[ArtCam\\_Trigger](#)

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

## ArtCam\_Trigger

Definition: **BOOL** ArtCam\_Trigger(**HACAM** *hACam*, **HWND** *hWnd*, **LPBYTE** *lpImage*, **LONG** *Size*, **BOOL** *TopDown*)

Function: Obtain image of camera in external trigger mode

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>HWND</b>	<i>hWnd</i>	Window Handle for receiving message
<b>LPBYTE</b>	<i>lpImage</i>	Address of arrangement for receiving image data
<b>LONG</b>	<i>Size</i>	Arrangement length of lpImage
<b>BOOL</b>	<i>TopDown</i>	Determination of up and down of image

### Function Detail:

Procedure of this function is similar to that of [ArtCam\\_CallBackPreview](#).

Timing of capturing depends on camera's clock speed with [ArtCam\\_CallBackPreview](#).

With this function, capturing is processed when triggered with external trigger.

When you initialize with this function, updates and obtaining message of image is sent only after the trigger is sent to camera.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

\*1: Use pulse signal from 0-5V to 0-12V for trigger signal.

Timing from input trigger to take a picture is different from each cameras.

Please refer the manual for more detail.

# WM\_GRAPHPAINT

**Definition:** `#define WM_GRAPHPAINT WM_APP + 2`

**Function:** Message is issued when a camera image is updated.

<b>WPARAM</b>	<i>wParam</i>	LPGP_INFO
<b>LPARAM</b>	<i>lParam</i>	Always NULL

## Function Detail:

LPGP\_INFO in which is received by *wParam* is pointer to the structure that stores image data

```
typedef struct GP_INFO {  
    LONG        lSize;  
    LONG        lWidth;  
    LONG        lHeight;  
    LONG        lBpp;  
    LONG        lFps;  
    LPBYTE pImage;  
} *LPGP_INFO;
```

LPGP\_INFO lpGPIF = (LPGP\_INFO)wParam

This message is sent to the callback procedure of the window when Window Handle is set to hWnd at [ArtCam\\_CallbackPreview](#) and [ArtCam\\_Trigger](#).

This message is sent when image is updated.

To obtain image data, assign pointer and array length of alignment to lImage and Size of [ArtCam\\_CallbackPreview](#). Then image data is stored in specified alignment when this message is sent.

When wParam is NULL, WM\_GRAPHPAINT becomes error.  
wParam and lParam will mean [WM\\_ERROR](#).

WM\_GRAPHPAINT is defined as 0x8002

## WM\_ERROR

**Definition:** #define WM\_GRAPHPAINT WM\_APP + 3

**Function:** Receive error message

**WPARAM**    *wParam*                      Always 0  
**LPARAM**    *lParam*                      Error Code

### Function Detail:

When Window Handle is specified at [ArtCam\\_Initialize](#), error code is sent to Window Procedure in case error occurs within SDK.

WM\_ERROR is defined as 0x8003.

Error codes are as below:

ERROR CODE	STATUS
ARTCAMSDK_NOERROR	Normal
ARTCAMSDK_NOT_INITIALIZE	not initialized
ARTCAMSDK_DISABLEDDEVICE	It was going to access disable device
ARTCAMSDK_CREATETHREAD	Failureure of creating thread for capturing
ARTCAMSDK_CREATEWINDOW	Failureure of creating window
ARTCAMSDK_OUTOFMEMORY	No enough memory for transferring image
	Or Failureure of securing memory
ARTCAMSDK_CAMERASET	Error of camera (device) settings
ARTCAMSDK_CAPTURE	Failureure of cap
ARTCAMSDK_PARAM	Wrong argument
ARTCAMSDK_DIRECTSHOW	Error of DirectShow initialization
ARTCAMSDK_UNSUPPORTED	This function is not supported
ARTCAMSDK_UNKNOWN	Unknown error
ARTCAMSDK_CAPTURELOST	Device lost
ARTCAMSDK_FILENOTFOUND	Cannot find specified file
ARTCAMSDK_FPGASET	Error at FPGA setting

## ArtCam\_StartPreview

Definition: **BOOL** ArtCam\_StartPreview(**HACAM** *hACam*)

Function: Start preview

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
--------------	--------------	----------------------------------

### Function Detail:

Start preview of camera image

This function is used internally for [ArtCam\\_Preview](#), [ArtCam\\_Record](#) and [ArtCam\\_CallBackPreview](#).

This function is only used to regenerate image, in which preview is stop, by calling

[ArtCam\\_StopPreview](#).

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

## ArtCam\_StopPreview

Definition: **BOOL** ArtCam\_StopPreview(**HACAM** *hACam*)

Function: Stop preview

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
--------------	--------------	----------------------------------

### Function Detail:

This function stops preview of image.

This function does not release device.

Please use this function only when you need to stop preview temporarily.

To display preview again, use [ArtCam\\_StartPreview](#).

This function is only available when preview is displayed with [ArtCam\\_Preview](#) and [ArtCam\\_CallBackPreview](#).

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

## ArtCam\_SaveImage

Definition: **BOOL** ArtCam\_SaveImage(  
    **HACAM** *hACam*, **LPCTSTR** *lpSaveName*, **LONG** *FileType*)

Function: Save image of camera

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LPCTSTR</b>	<i>lpSaveName</i>	Name of file to be saved
<b>LONG</b>	<i>FileType</i>	Type of save

### Function Detail:

Save camera image in computer files

Image to be saved is the last image obtained by image-capturing functions such as [ArtCam\\_Preview](#), [ArtCam\\_CallBackPreview](#), [ArtCam\\_SnapShot](#) & [ArtCam\\_Trigger](#)

Please note that depending on system environment, speed clock of camera and file types, saved image may deteriorate while real-time image is obtained with [ArtCam\\_Preview](#) and [ArtCam\\_CallBackPreview](#). When this happens, stopping image update temporarily by [ArtCam\\_StopPreview](#) may prevent image deterioration.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

This function is used to save live image obtained by camera.

To save images that are processed by application, save the images at application.

File type can be selected from BMP, binary (RAW), JPEG (high-quality, standard & low-quality), PNG and TIFF.

You cannot save 16 bits image in JPEG

When you save image in JPEG, it is saved as gray scale of 8 bits bit-depth.

BMP and RAW can be saved in 16 bits. However, pallet info is not saved in file. Therefore image may not be display correctly for softwares that do not have special reading routine.

To save images in 16 bits, use of PNG and TIFF are recommended. With these file Initializes, we recommend you to read the images in Artray's Viewer Software or Adobe Photoshop6.

Please note that not every image-processing application is compatible with 16 bits image.

Regarding files to be saved with this function, we only support on reading procedure on BMP and RAW.

We will not provide support on reading procedures of other file Initializes and saving procedure.



## ArtCam\_GetImage

Definition: **BOOL** ArtCam\_GetImage(  
    **HACAM** *hACam*, **LPBYTE** *lpImage*, **LONG** *Size*, **BOOL** *TopDown*)

Function: Obtain image of camera

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LPBYTE</b>	<i>lpImage</i>	Address of arrangement for receiving image data
<b>LONG</b>	<i>Size</i>	Arrangement length of lpImage
<b>BOOL</b>	<i>TopDown</i>	Determination of up and down of image

### Function Detail:

Obtain image of camera.

When function succeeds, previously obtained data is stored in lpImage.

Image will not be obtained unless the size of alignment is equal to or larger than size of image.

If bitmap prepared is DDB (top-down), specify Topdown as True. If bitmap is DIB (bottom-up), specify Topdown as False.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

This function is used to obtain image asynchronously while [ArtCam\\_Preview](#) or [ArtCam\\_CallbackPreview](#) is used.

If you only need to obtain 1 frame, use [ArtCam\\_SnapShot](#).

This function assumes that the PC with low specs is used, or language, which has slow processing speed, is used.

To create with C & C++, receive message of image updates by [WM\\_GRAPHPAINT](#)

## Setting dialog

### ArtCam\_SetCameraDlg

Definition: **BOOL** ArtCam\_SetCameraDlg(**HACAM** *hACam*, **HWND** *hWnd*)

Function: Show dialog of camera settings

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>HWND</b>	<i>hWnd</i>	Parent window for showing dialog

#### Function Detail:

This function displays a dialog box that allows you to alter settings such as size of image and frame rate.

Dialog box displayed varies with the device plugged.

When you call this function while preview is displayed, preview will temporarily stop.

Preview will be displayed again once dialog box is closed.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

### ArtCam\_SetImageDlg

Definition: **BOOL** ArtCam\_SetImageDlg(**HACAM** *hACam*, **HWND** *hWnd*)

Function: Show dialog of filter settings

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>HWND</b>	<i>hWnd</i>	Parent window for showing dialog

#### Function Detail:

This function displays a dialog box that allows you to alter settings such as brightness, contrast and white balance.

Dialog box displayed varies with the device plugged.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

## ArtCam\_SetAnalogDlg

Definition: **BOOL** ArtCam\_SetCameraDlg(**HACAM** *hACam*, **HWND** *hWnd*)

Function: Show dialog of port/camera settings

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>HWND</b>	<i>hWnd</i>	Parent window for showing dialog

### Function Detail:

This function displays a dialog box that allows you to alter settings such as analog port and internal camera device.

Dialog box displayed varies with the device plugged.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

### ArtCam\_SetPreviewWindow

Definition: **BOOL** ArtCam\_SetPreviewWindow(

**HACAM** *hACam*, **HWND** *hWnd*, **LONG** *Left*, **LONG** *Top*, **LONG** *Right*, **LONG** *Bottom*)

Function: Specify window to display image of camera and specify its range

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>HWND</b>	<i>hWnd</i>	Specify handle of window to be displayed
<b>LONG</b>	<i>Left</i>	Specify upper-left X-coordinate of rectangle
<b>LONG</b>	<i>Top</i>	Specify upper-left Y-coordinate of rectangle
<b>LONG</b>	<i>Right</i>	Specify lower-right X-coordinate of rectangle
<b>LONG</b>	<i>Bottom</i>	Specify lower-right Y-coordinate of rectangle

Function Detail:

When Window handle is specified to *hWnd*, create child window in the window and display in the child window.

When **NULL** is specified to *hWnd*, create new window.

Success: Returned **TRUE** or 1

Failure: Returned **FALSE** or 0

## ArtCam\_SetCaptureWindow

**Definition:** **BOOL** ArtCam\_SetCaptureWindow(  
    **HACAM** *hACam*, **LONG** *Width*, **LONG** *Height*, **LONG** *Fps* )

**Function:** Specify image size of camera and frame rate

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>Width</i>	Specify width of image in the unit of pixel
<b>LONG</b>	<i>Height</i>	Specify height of image in the unit of pixel
<b>LONG</b>	<i>Fps</i>	Specify frame rate

### Function Detail:

-For DirectShow camera (ArtCamSdk.dll)-

For Frame, specify frame number calculated by FPS (frame rate per second) \* 10.

If FPS is 30, it will be 300.

-Other-

Frame will be ignored.

Based on Width and Height as operative resolution, most appropriate value among registered will be set.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

This function is as using DirectShow camera to do record setting.

Other than this, please use [ArtCam\\_SetCaptureWindowEx](#)

Set up correct pixel size at the function after Initializing by Initialize, otherwise it would be Failed to CallBackPreview, Snapshot functions.

## ArtCam\_SetCaptureWindowEx

**Definition:** **BOOL** ArtCam\_SetCaptureWindowEx( **HACAM** *hACam*, **LONG** *HTotal*,  
**LONG** *HStart*, **LONG** *HEffective*, **LONG** *VTotat*, **LONG** *VStart*, **LONG** *VEffective*)

**Function:** Specify image size of camera(ROIFunction)

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>HTotal</i>	Specify total horizontal width of camera in the unit of pixel
<b>LONG</b>	<i>HStart</i>	Specify starting point of horizon
<b>LONG</b>	<i>HEffective</i>	Specify effective horizontal width in the unit of pixel
<b>LONG</b>	<i>VTotat</i>	Specify vertical total height in the unit of pixel
<b>LONG</b>	<i>VStart</i>	Specify starting point of vertical
<b>LONG</b>	<i>VEffective</i>	Specify effective vertical height

### Function Detail:

Set up capture image size.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

ROIFunction is a function only for CMOS sensor camera.

You cannot set up image size at CCD sensor camera.

For color image, because of Bayer converting, you need more than 5 pixel active imager size at both horizontal and vertical.

We recommend to set up multiple of 4 for active horizontal pixel and active vertical pixel. Especially, it would not view images properly if you set up other than multiple of 4 to active horizontal pixel.

You cannot use this function for DirectShow camera.

You can change the size at ArtCam\_SetCaptureWindow.

## ArtCam\_GetCaptureWindowEx

Definition: **BOOL** ArtCam\_GetCaptureWindowEx(**HACAM** *hACam*, **LONG\*** *HTotal*, **LONG\*** *HStart*, **LONG\*** *HEffective*, **LONG\*** *VTotat*, **LONG\*** *VStart*, **LONG\*** *VEffective*)

Function: Obtain image size of camera

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG*</b>	<i>HTotal</i>	Returns total width of camera in unit of pixel
<b>LONG*</b>	<i>HStart</i>	Returns starting point of width
<b>LONG*</b>	<i>HEffective</i>	Returns operative width of camera
<b>LONG*</b>	<i>VTotat</i>	Returns total height of camera in unit of pixel
<b>LONG*</b>	<i>VStart</i>	Returns starting point of height
<b>LONG*</b>	<i>VEffective</i>	Returns operative height of camera

Function Detail:

For ArtCamSdk.dll :

Use [ArtCam\\_Width](#) [ArtCam\\_Height](#) [ArtCam\\_Fps](#)

Other :

Obtain each parameter of camera

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

## ArtCam\_SetColorMode

**Definition:** **BOOL** ArtCam\_SetColorMode(**HACAM** *hACam*, **LONG** *ColorMode*)

**Function:** Set color mode for image capturing

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>ColorMode</i>	Number of data bits

**Function Detail:**

Specify number of bits.

8: 8 bits monochrome image

16 16 bits monochrome image (10 bits for cameras & 16 bits for CNV converters)

24: 24 bits color image (BGR, 8 bits each)

32: 32 bits color image (BGRA, 8 bits, A=invalid)

48: 48 bits color image (BGR, 16 bits each)

64: 64 bits color image (BGRA, 16 bits each, A=invalid)

With 16 (10) bits, numerical values vary slightly depending on the environment.

When you create an application, make sure that the application is compatible with 10, 12, 14, & 16.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

If you use DirectShow camera(ArtCamSdk.dll),

Image is fixed at 24 bits color.

You cannot set up color space at this function.

## ArtCam\_GetColorMode

**Definition:** **LONG** ArtCam\_GetColorMode (**HACAM** *hACam*)

**Function:** Obtain current color mode

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
--------------	--------------	----------------------------------

**Function Detail:**

Success: Returned image bit number as LONG value (8 - 64)

Failure: Returned -1



## ArtCam\_SetCrossbar

Definition: **BOOL** ArtCam\_SetCrossbar(**HACAM** *hACam*, **LONG** *Input*, **LONG** *Output* )

Function: Specify analog port to be plugged

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>Input</i>	Number of input port
<b>LONG</b>	<i>Output</i>	Number of output port

### Function Detail:

You can change connecting analog port.

Specify an integral from 0 to "*Input*" to switch.

"*Output*" is extension function and normally specify 0.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

This function is only for ArtCnv series.

This is only for switching input port of the model in which has several analog connecting ports, such as ArtCnvII-2ch, ArtCnv-HAKO.

Right after switching input port (about 100m/sec), image would be disordered because of obtaining a synchronized analog signal.

## ArtCam\_SetDeviceNumber

**Definition:** **BOOL** ArtCam\_SetDeviceNumber(**HACAM** *hACam*, **LONG** *Number*)

**Function:** Assign number of device to be plugged

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>Number</i>	Assign device number from 0 to 9

**Function Detail:**

After you call this function and initialize with functions such as ArtCam\_Preview, ArtCam\_Record & ArtCam\_CallBackPreview, image of specified device will be displayed. To confirm device number, use ArtCam\_EnumDevice & ArtCam\_GetDeviceName.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

## ArtCam\_GetDeviceName

**Definition:** **BOOL** ArtCam\_GetDeviceName(  
HACAM *hACam*, **LONG** *index*, **LPSTR** *szDeviceName*, **LONG** *nSize*)

**Function:** Obtain name of specified device

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>index</i>	Specify number of device from 0 to 9
<b>LPSTR</b>	<i>szDeviceName</i>	Names of devices are copied, if they are operative
<b>LONG</b>	<i>nSize</i>	Size of szDeviceName

**Function Detail:**

Confirm if device specified by index is operative. If it's operative, store the name of device to szDeviceName.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

## ArtCam\_EnumDevice

**Definition:** `LONG ArtCam_EnumDevice(HACAM hACam, TCHAR szDeviceName[10][256])`

**Function:** Recount names of operative device

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>TCHAR</b>	<i>szDeviceName</i>	Names of operative devices are copied

### Function Detail:

Utilized device name is stored in *szDeviceName*.

For example, if two ARTCAM-130MI cameras are possible to use,

Normaly,

a string ArtCam130MI\_0 is stored in *szDeviceName*[0]

a string ArtCam130MI\_1 is stored in *szDeviceName*[1]

Number specified by [ArtCam\\_SetDeviceNumber](#) is same as alignment number stored in *szDeviceName*.

To use device stored in *szDeviceName*[1], specify `ArtCam_SetDeviceNumber (1)`.

If the function is successfully worked, utilized device number is returned as LONG value.

Please give the strings of [10][256] for second Argument.

If a string is smaller than this, Return value would be returned to 0.

You cannot use this function for VB.NET and C#.NET.

To obtain device name with other languages, please use [ArtCam\\_GetDeviceName](#)

## ArtCam\_SetCameraType

Definition: **BOOL** ArtCam\_SetCameraType(**HACAM** *hACam*, **LONG** *Flg*)

Function: Identified connecting SATA camera

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>Flg</i>	SATA camera type code

Function Detail:

Choose a SATA camera in which specified camera type code in *Flg*.

Call after Initializing ArtCamSdk\_Sata.dll at Initialize function.

Caemra type code is define as below:

MODEL	CAMERA TYPE CODE
ARTSAT-0506LVDS	ARTCAM_CAMERATYPE_SATA_LVDS
ARTCAM-300MI-SATA	ARTCAM_CAMERATYPE_SATA_300MI
ARTCAM-500MI-SATA	ARTCAM_CAMERATYPE_SATA_500MI
ARTCAM-MV413-SATA	ARTCAM_CAMERATYPE_SATA_MV413
ARTCAM-800MI-SATA	ARTCAM_CAMERATYPE_SATA_800MI
ARTCAM-036MI-SATA	ARTCAM_CAMERATYPE_SATA_036MI
ARTCAM-267KY-SATA	ARTCAM_CAMERATYPE_SATA_150P

## ArtCam\_GetCameraType

Definition: **LONG** ArtCam\_GetCameraType(**HACAM** *hACam*, **LPBOOL** *Error*)

Function: Obtain connecting SATA camera

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LPBOOL</b>	<i>Error</i>	Error information

Function Detail:

Obtain connecting SATA camera's camera type code.

Caemra type code is define as below:

MODEL	CAMERA TYPE CODE
ARTSAT-0506LVDS	ARTCAM_CAMERATYPE_SATA_LVDS
ARTCAM-300MI-SATA	ARTCAM_CAMERATYPE_SATA_300MI
ARTCAM-500MI-SATA	ARTCAM_CAMERATYPE_SATA_500MI
ARTCAM-MV413-SATA	ARTCAM_CAMERATYPE_SATA_MV413
ARTCAM-800MI-SATA	ARTCAM_CAMERATYPE_SATA_800MI
ARTCAM-036MI-SATA	ARTCAM_CAMERATYPE_SATA_036MI
ARTCAM-267KY-SATA	ARTCAM_CAMERATYPE_SATA_150P

## ArtCam\_Width

Definition: **LONG** ArtCam\_Width(**HACAM** *hACam*)

Function: Obtain width of camera image

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
--------------	--------------	----------------------------------

Function Detail:

DirectShow camera (ArtCamSdk.dll) :

Assigned Width value is returned at [ArtCam\\_SetCaptureWindow](#)

Cameras that capture size is fixed :

Standard size set within SDK is returned by LONG value.

(ArtCnvSdk.dll, ArtCamSdk\_150P3.dll, ArtCamSdk\_500P.dll etc...)

Cameras that capture size is flexible :

Assigned Heffective value at [ArtCam\\_SetCaptureWindowEx](#) is returned by LONG value.

## ArtCam\_Height

Definition: **LONG** ArtCam\_Height(**HACAM** *hACam*)

Function: Obtain height of camera image

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle of camera identification
--------------	--------------	------------------------------------

Function Detail:

DirectShow camera (ArtCamSdk.dll) :

Assigned Height value is returned at [ArtCam\\_SetCaptureWindow](#) by LONG value.

Cameras that capture size is fixed :

Standard size set within SDK is returned by LONG value.

(ArtCnvSdk.dll, ArtCamSdk\_150P3.dll, ArtCamSdk\_500P.dll etc...)

Cameras that capture size is flexible :

Assigned Veffective value at [ArtCam\\_SetCaptureWindowEx](#) is returned by LONG value.

## ArtCam\_Fps

Definition: **LONG** ArtCam\_Fps(**HACAM** *hACam*)

Function: Obtain frame rate of camera

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
--------------	--------------	----------------------------------

Function Detail:

Obtained setting frame rate by LONG value.

Frame rate is obtained by following: FPS \* 10

If FPS is 30, it will be 300.

This function is only for ArtCamSdk.dll (DirectShow camera)

## ArtCam\_GetCameraInfo

Definition: **BOOL** ArtCam\_GetCameraInfo(**HACAM** *hACam*, **LPCAMERAINFO** *pInfo*)

Function: Obtain camera information

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LPCAMERAINFO</b>	<i>pInfo</i>	Camera information

Function Detail:

Obtaining the information of connecting camera's setting possible value.

CAMERAINFO type structure is defined as below:

```
CAMERAINFO {
    LONG    ISize;           // structure's size
    LONG    IWidth;          // Camera's effective maximum width
    LONG    IHeight;         // Camera's effective maximum height
    LONG    IGlobalGainMin;  // Lowest value of global gain(cannot use -1 camera)
    LONG    IGlobalGainMax; // Maximum value of global gain(cannot use -1 camera)
    LONG    IColorGainMin;  // Lowest value of color gain(cannot use -1 camera)
    LONG    IColorGainMax;  // Maximum value of color gain(cannot use -1 camera)
    LONG    IExposureMin;   // Lowest value of exposure time(cannot use -1 camera)
    LONG    IExposureMax;   // Maximum value of exposure time(cannot use -1 camera)
} *LPCAMERAINFO;
```

## ArtCam\_SetIOPort

**Definition:** **BOOL** ArtCam\_SetIOPort(  
    **HACAM** *hACam*, **BYTE** *byteData*, **LONG** *longData*, **DWORD** *Reserve*)

**Function:** Write data to IO.

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>BYTE</b>	<i>byteData</i>	Data written in IO (byte data)
<b>LONG</b>	<i>longData</i>	Not in use. Please specify 0.
<b>DWORD</b>	<i>Reserve</i>	Not in use. Please specify 0.

**Function Detail:**

Write data (8 bit) into I/O port.

Port will be initialized at low level when loading device driver (i.e. loading operating system or plugging USB)

e.g. When "0x0C" is saved into "byteData", both OUT0 and OUT1 ports will be at Hi level.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

## ArtCam\_GetIOPort

**Definition:** **BOOL** ArtCam\_GetIOPort(  
    **HACAM** *hACam*, **LPBYTE** *byteData*, **LPLONG** *longData*, **DWORD** *Reserve*)

**Function:** Read data from IO

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LPBYTE</b>	<i>byteData</i>	Data read from IO (byte data)
<b>LPLONG</b>	<i>longData</i>	Not in use. Please specify NULL.
<b>DWORD</b>	<i>Reserve</i>	Not in use. Please specify 0.

**Function Detail:**

Read data (8 bit) from I/O port. At the newest driver (July 11th. 2007) Low level (LSB)'s 1and 2 bits are IN0, and IN1 port.

Input level of port returns 1 when it is Hi in byteData.

e.g. When IN0 level is Low and IN1 level is Hi, "0x02" will be in byteData.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

This function is effective only for corresponded to I/O customized camera.

Port is Initialized to Low level at the time device drive is loading (Start OS or connecting USB). Voltage is unstable till driver is loaded.

## ArtCam\_SetSubSample

**Definition:** **BOOL** ArtCam\_SetSubSample(**HACAM** *hACam*, **LONG** *SubSampleMode*)

**Function:** Set sub-sampling mode

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>SubSampleMode</i>	Sub-sampling mode

**Function Detail:**

This function sets sub-sampling transfer mode.

Thinning out image is transferred. Image is thinned out by value set in *SubSampleMode*.

SUBSAMPLE\_1 All data

SUBSAMPLE\_2 Data equals to half of matrix

SUBSAMPLE\_4 Data equals to quarter of matrix

SUBSAMPLE\_8 Data equals to eighth of matrix

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

When [ArtCam\\_CallBackPreview](#) is used in this function, data less than assigned image size is transferred.

Transfer mode is different at each models.

There is not this function at CCD camera.

## ArtCam\_GetSubSample

**Definition:** **LONG** ArtCam\_GetSubSample(**HACAM** *hACam*)

**Function:** Obtain current sub-sampling mode

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
--------------	--------------	----------------------------------

**Function Detail:**

Obtain current pixel skipping transfer mode.

SUBSAMPLE\_1 All data

SUBSAMPLE\_2 Data equals to half of matrix

SUBSAMPLE\_4 Data equals to quarter of matrix

SUBSAMPLE\_8 Data equals to eighth of matrix

Return -1 if the function is Failed.



## ArtCam\_SetWaitTime

Definition: **BOOL** ArtCam\_SetWaitTime(**HACAM** *hACam*, **LONG** *WaitTime*)

Function: Assign WaitTime

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>WaitTime</i>	WaitTime

Function Detail:

This function assigns waiting time for obtaining video from [ArtCam\\_Preview](#) and [ArtCam\\_CallBackPreview](#).

Specified wait time between frame by mm/sec. Default is 10.

Success: Returned TRUE or 1.

Failure: Returned FALSE or 0.

Frame rate will be increased when you assign a small value in Wait Time. Missing will be decreased.  
CPU's using rate will be increased.

Please specify between 5 to 20 as average number.

Frame rate will decrease when you assign a large value for Wait Time

## ArtCam\_GetWaitTime

Definition: **LONG** ArtCam\_GetWaitTime(**HACAM** *hACam*)

Function: Obtain WaitTime

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
--------------	--------------	-------------------------------------

Function Detail:

Success: Obtain current Wait Time by LONG value to RETURN value.

Failure: Return -1 to RETURN value.

## ArtCam\_SetMirrorV

**Definition:** **BOOL** ArtCam\_SetMirrorV(**HACAM** *hACam*, **BOOL** *Flg*)

**Function:** Set flip vertical mirroring function

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>BOOL</b>	<i>Flg</i>	Reverse flag

**Function Detail:**

With camera's hardware function, you can transfer data in flip vertical. Setting *Flg* to True will enable mirroring function, while false will disable the function.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

Each models has different default flg.

## ArtCam\_GetMirrorV

**Definition:** **BOOL** ArtCam\_GetMirrorV(**HACAM** *hACam*)

**Function:** Obtain conditions of flip vertical mirroring function

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
--------------	--------------	----------------------------------

**Function Detail:**

Confirm if flip vertical mirroring function is enabled or not.

Mirroring function enabled: True

Mirroring function disabled: False

Each models has different default flg.

## ArtCam\_SetMirrorH

Definition: **BOOL** ArtCam\_SetMirrorH(**HACAM** *hACam*, **BOOL** *Flg*)

Function: Set flip horizontal mirroring function

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>BOOL</b>	<i>Flg</i>	Reverse flag

Function Detail:

With camera's hardware function, you can transfer data in flip horizontal. Setting Flg to True will enable mirroring function, while false will disable the function.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

Each models has different default flg.

## ArtCam\_GetMirrorH

Definition: **BOOL** ArtCam\_GetMirrorH(**HACAM** *hACam*)

Function: Obtain current conditions of flip horizontal mirroring function

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
--------------	--------------	----------------------------------

Function Detail:

With camera's hardware function, you can transfer data in flip horizontal.

Mirroring function enabled: True

Mirroring function disabled: False

Each models has different default flg.

## ArtCam\_SetHalfClock

**Definition:** **BOOL** ArtCam\_SetHalfClock(**HACAM** *hACam*, **LONG** *Value*)

**Function:** Halve clock speed of camera

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>Value</i>	Flag for half clock

**Function Detail:**

Clock speed of camera can be halved. Halving clock speed will halve frame rate.

Halve clock is effective when specified 1 on *Value*. In validate at 0.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

Use this function when:

Does not require high frame rate

Connect several camera

Use low spec PC

USB transferring speed is not enough and an image is disordered.

Use 10 bits transfer mode

This function does not allow direct switching.

Having set this function in advance, clock switching will be reflected when you initialize using image capturing functions.

Depending on models, clock switching may take up to several seconds.

## ArtCam\_GetHalfClock

**Definition:** **LONG** ArtCam\_GetHalfClock(**HACAM** *hACam*, **LPBOOL** *Error*)

**Function:** Obtain condition of camera clock

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LPBOOL</b>	<i>Error</i>	Error information

**Function Detail:**

Obtain current condition of camera clock speed by LONG value

Effective half clock: 1

Invalid half clock: 0.

Success: TRUE on *Error*

Failure: FALSE on *Error*

## ArtCam\_SetAutoIris

**Definition:** **BOOL** ArtCam\_SetAutoIris(**HACAM** *hACam*, **LONG** *Value*)

**Function:** Set condition of auto-iris

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>Value</i>	Flag for auto-iris

**Function Detail:**

Set up Auto Iris (Auto brightness revision)'s effective/invalid.  
specify in *Value*:

*Value*=0    Invalid *Auto Iris*  
*Value*=1    Effective Auto Iris by Shutter Speed  
*Value*=2    Effective Auto Iris by Global Gain

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

There are some difference depend on using environment by each model.

We recommend to avoid using this function with filter because if filter of Sharpness or Brightness is effective, this function would not work properly.

## ArtCam\_GetAutoIris

**Definition:** **LONG** ArtCam\_GetAutoIris(**HACAM** *hACam*, **LPBOOL** *Error*)

**Function:** Obtain condition of auto-iris

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LPBOOL</b>	<i>Error</i>	Error information

**Function Detail:**

Obtain current condition of auto-iris (Auto brightness revision) by LONG value.

Success: Invalid=0

Exposure time settings=1

Gain settings=2

Success: TRUE on *Error*

Failure: FALSE on *Error*

## ArtCam\_SetSamplingRate

Definition: **BOOL** ArtCam\_SetSamplingRate(**HACAM** *hACam*, **LONG** *Value*)

Function: Set frame rate and capturing size

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>Value</i>	Capturing mode

Function Detail:

Set analog signal's frame rate and capturing size

Set 0-3 on *Value*. Set up capture mode as below(NTSC)

*Value* =0    720 \* 470    30 frame

*Value* =1    720 \* 480    15 frame

*Value* =2    640 \* 470    30 frame

*Value* =3    640 \* 480    15 frame

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

This function is only for NTSC-USB2.0 converter ARTCNVII (ArtCnvSdk.dll).

Does not stretch an image.

If specified size signal is not input, there might be black or green line on four corners.

## ArtCam\_GetSamplingRate

Definition: **LONG** ArtCam\_GetSamplingRate(**HACAM** *hACam*, **LPBOOL** *Error*)

Function: Obtain frame rate and capturing size.

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LPBOOL</b>	<i>Error</i>	Error information

Function Detail:

Obtain frame rate and capturing size

RETURN value would be obtained by 0-3 LONG value.

Obtained value is same value as [ArtCam\\_SetSamplingRate](#)Value

Success: TRUE on *Error*

Failure: FALSE on *Error*

## ArtCam\_GetVideoFormat

Definition: **LONG** ArtCam\_GetVideoFormat(**HACAM** *hACam*, **LPBOOL** *Error*)

Function: Obtain type of image signal of plugged camera

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LPBOOL</b>	<i>Error</i>	Error information

Function Detail:

Obtain type of image signal of plugged camera

RETURN value is obtained by 0-3 LONG value.

Initialized obtained value and image signal is as below:

- 0 NTSC
- 1 PAL
- 2 PALM
- 3 SECAM

Success: TRUE on *Error*

Failure: FALSE on *Error*

## ArtCam\_WriteSromID

Definition: **BOOL** ArtCam\_WriteSromID(**HACAM** *hACam*, **LONG** *Address*, **LONG** *Value*)

Function: Register sub-code

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>Address</i>	Address to be written
<b>LONG</b>	<i>Value</i>	Data to be written

Function Detail:

Write data in 8 Byte memory space (EEPROM) of a camera.

Please use this function to distinguish several cameras.

It is possible to set *Address* at 0-255.

It is possible to set *value* at 0-255.

Please note that the number more than above data will be rounded down.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

## ArtCam\_ReadSromID

Definition: **LONG** ArtCam\_ReadSromID(**HACAM** *hACam*, **LONG** *Address*, **LPBOOL** *Error*)

Function: Read sub-code

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>Address</i>	Address to be read
<b>LPBOOL</b>	<i>Error</i>	Error information

Function Detail:

Read data from 8 Byte memory space (EEPROM) of a camera.

Set 0-7 on *Address*, and this value would be obtained by LONG value.

Once register ID in camera with [ArtCam\\_WriteSromID](#), it is possible to manage more than one camera separately if you compare the number.

Success: TRUE on *Error*

Failure: FALSE on *Error*



## ArtCam\_WriteRegister

Definition: **BOOL** ArtCam\_WriteRegister(**HACAM** *hACam*, **BYTE** *Address*, **DWORD** *Value*)

Function: Write to a sensor register

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>BYTE</b>	<i>Address</i>	Writing address
<b>DWORD</b>	<i>Value</i>	Writing data

Function Detail:

Writing data on camera sensor's register

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

We do not open the detail of register setting on public.

This function is for some customized cameras.

Please avoid to change a register value on normal camera. Unexpected trouble would be occurred.

## ArtCam\_ReadRegister

Definition: **DWORD** ArtCam\_ReadRegister(**HACAM** *hACam*, **BYTE** *Address*, **LPBOOL** *Error*)

Function: Read sensor register value

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>BYTE</b>	<i>Address</i>	Reading address
<b>LPBOOL</b>	<i>Error</i>	Error information

Function Detail:

Read register value or a camera sensor's specified address.

Success: TRUE on *Error*

Failure: FALSE on *Error*

We do not open the register setting detail on public.

This function is only for some customized cameras.

## ArtCam\_SetFilterValue

**Definition:** **BOOL** ArtCam\_SetFilterValue(**HACAM** *hACam*, **LONG** *FilterType*, **LONG** *Value*)

**Function:** Set image filter information

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>FilterType</i>	Type of filter to be set
<b>LONG</b>	<i>Value</i>	Number to be set

**Function Detail:**

This function allows you to directly set values, which can be set with

[ArtCam\\_SetImageDlg](#).

Regarding FilterType, please refer to defined file of each language.

Success: Returned TRUE or 1

Failure: Returned FALSE or 0

## ArtCam\_GetFilterValue

**Definition:** **LONG** ArtCam\_GetFilterValue(  
**HACAM** *hACam*, **LONG** *FilterType*, **LPBOOL** *Error*)

**Function:** Obtain image filter information

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>FilterType</i>	Type of filter to be set
<b>LPBOOL</b>	<i>Error</i>	Error information

**Function Detail:**

This function allows you to obtain value of parameter that can be set

[ArtCam\\_SetImageDlg](#) and [ArtCam\\_SetAnalogDlg](#).

When you set NULL for Error, error info will not be obtained.

Regarding FilterType, please refer to defined file of each language.

Success: TRUE on *Error* and returned setting number on *Filter Type* to LONG value.

Failure: FALSE on *Error*

## ArtCam\_Set\*\*\*

Definition: **BOOL** ArtCam\_Set\*\*\*(**HACAM** *hACam*, **LONG** *Value*)

Function: Set image filter information

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LONG</b>	<i>Value</i>	Value to be set

Function Detail:

Wrapper function of [ArtCam\\_SetFilterValue](#)

This function will be called when second argument of [ArtCam\\_SetFilterValue](#) is set

For example, if you want to change Global Gain to 30,

Instead of doing ArtCam\_SetFilterValue  
(hACam, ARTCAM\_FILTERTYPE\_GLOBAL\_GAIN, 30),  
set to ArtCam\_SetGlobalGain(hACam,30).

## ArtCam\_Get\*\*\*

Definition: **LONG** ArtCam\_Get\*\*\*(**HACAM** *hACam*, **LPBOOL** *Error*)

Function: Obtain image filter information

Argument:

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LPBOOL</b>	<i>Error</i>	Error information

Function Detail:

Wrapper function of [ArtCam\\_GetFilterValue](#)

This function will be called when second argument of [ArtCam\\_GetFilterValue](#) is set

## ArtCam\_GetRealExposureTime

**Definition:** **DWORD** ArtCam\_GetRealExposureTime(**HACAM** *hACam*, **LPBOOL** *Error*)

**Function:** Get the real exposure time

**Argument:**

<b>HACAM</b>	<i>hACam</i>	Handle for camera identification
<b>LPBOOL</b>	<i>Error</i>	Error information

### Function Detail:

Obtain the real exposure time by LONG value.

The unit of exposure time is microsecond.

Success: TRUE on Error.

Failure: FALSE on Error.

### Notice:

Please note that the unit of a related function ArtCam\_SetExposureTime is in H because ArtCam\_SetExposureTime sets the exposure time on the sensors. H is a unit of shutter speed calculation, not a time unit. To obtain the shutter speed in time units, please use ArtCam\_GetExposureTime.

The exposure time is calculated by the following formulas.

$$1H = (\text{Effective Horizontal Pixels} + \text{Horizontal Blank Pixels}) * \text{Pixel Clock}$$
$$\text{Exposure Time} = \text{Shutter Setting Value} * 1H$$

This function calculates internally on the software.

Please note that the pixel clock varies with the model. For example, the clock parameter for 130MI is 1/24000000. Also, if the hardware has been updated or the clock is set to half, this function may not return the correct value.

## Image Filter Setting Possible Value

### For All Cameras

\*Exclude ARTCNVII and ARTUST series

ARTCAM_FILTERTYPE_BRIGHTNESS			
Set Brightness	Min:	-255	Max: 255 Def: 0
Setter: ArtCam_SetBrightness	Getter:		

ARTCAM_FILTERTYPE_CONTRAST			
Set Contrast	Min:	-127	Max: 127 Def: 0
Setter: ArtCam_SetContrast	Getter: ArtCam_GetContrast		

ARTCAM_FILTERTYPE_HUE			
Set Hue	Min:	-360	Max: 360 Def: 0
Setter: ArtCam_SetHue	Getter: ArtCam_GetHue		

ARTCAM_FILTERTYPE_SATURATION			
Set Saturation	Min:	-255	Max: 255 Def: 0
Setter: ArtCam_SetSaturation	Getter: ArtCam_GetSaturation		

ARTCAM_FILTERTYPE_SHARPNESS			
Set Sharpness	Min:	0	Max: 30 Def: 0
Setter: ArtCam_SetSharpness	Getter: ArtCam_GetSharpness		
*Frame rate might be down because it would use more CPU performance.			

ARTCAM_FILTERTYPE_BAYER_GAIN_R / BAYER_GAIN_G / BAYER_GAIN_B			
Set Red/Green/Blue Bayer	Min:	0	Max:400(200) Def: 100
Setter: ArtCam_SetBayerGainRed	Getter:	ArtCam_GetBayerGainRed	
Setter: ArtCam_SetBayerGainGreen	Getter:	ArtCam_GetBayerGainGreen	
Setter: ArtCam_SetBayerGainBlue	Getter:	ArtCam_GetBayerGainBlue	
Bayer value is for change the color balance by a software. Normally you cannot set up green Bayer. If you set up auto ehite balance(BAYER_GAIN_AUTO), this value would be changed. * There are 200 and 400 for Max depends on models. If moving is unstable when you set up more than 201 because of PC's spec, please use the value up to 200.			

ARTCAM_FILTERTYPE_BAYER_GAIN_AUTO			
Set ON/OFF of Auto White Balance	Data:	BOOL	Def: FALSE
Setter: ArtCam_SetBayerGainAuto	Getter: ArtCam_GetBayerGainAuto		

ARTCAM_FILTERTYPE_BAYER_GAIN_RGB			
Set RGB Bayer Value at once		Min: 0	Max: 0xffffffff
Def:0x646464			
Setter:	ArtCam_SetBayerGainRGB	Getter:	ArtCam_GetBayerGainRGB
* Set 0-255 value one time to R/G/B Bayer bvalue. This function is normally not in use.			

ARTCAM_FILTERTYPE_GAMMA			
Set Gamma Value (gamma1.0=100)	Min:	0	Max: 200 Def: 100
Setter: ArtCam_SetGamma	Getter:	ArtCam_GetGamma	
*Frame rate might be down because it would use more CPU performance.			

ARTCAM_FILTERTYPE_GLOBAL_GAIN			
Set Global Gain	Min: ****	Max: ****	Def: ****
Setter: ArtCam_SetGlobalGain	Getter: ArtCam_GetGlobalGain		
*Setting value and default value is different at each sensor			

ARTCAM_FILTERTYPE_COLOR_GAIN_R / GAIN_G1 / GAIN_G2 / GAIN_B	
Set Gain by Each Color	Setting Range Is Same as Global Gain
Setter: ArtCam_SetColorGainRed	Getter: ArtCam_GetColorGainRed
Setter: ArtCam_SetColorGainGreen1	Getter: ArtCam_GetColorGainGreen1
Setter: ArtCam_SetColorGainGreen2	Getter: ArtCam_GetColorGainGreen2
Setter: ArtCam_SetColorGainBlue	Getter: ArtCam_GetColorGainBlue
*This is special function for CMOS camera. Normally this is as same as global gain's value.	

ARTCAM_FILTERTYPE_EXPOSURETIME			
Set Shutter Speed	Min: ****	Max: ****	Def: ****
Setter: ArtCam_SetExposureTime	Getter: ArtCam_GetExposureTime		
*Setting value and default value is different at each sensor			

ARTCAM_FILTERTYPE_BAYERMODE				
Change Bayer Pattern	Min:	0	Max: 3	Def: ****
Setter: ArtCam_SetBayerMode	Getter:	ArtCam_GetGlobalGain		
*Def to output correct color is different as each sensor.				

## ARTCNVII

ARTCAM_FILTERTYPE_BRIGHTNESS			
Set Brightness	Min:	0	Max: 255 Def: 128
Setter: ArtCam_SetBrightness	Getter: ArtCam_GetBrightness		

ARTCAM_FILTERTYPE_CONTRAST			
Set Contrast	Min:	0	Max: 255 Def: 128
Setter: ArtCam_SetContrast	Getter: ArtCam_GetContrast		

ARTCAM_FILTERTYPE_HUE			
Set Hue	Min:	0	Max: 255 Def: 0
Setter: ArtCam_SetHue	Getter: ArtCam_GetHue		

ARTCAM_FILTERTYPE_SATURATION			
Set Saturation	Min:	0	Max: 255 Def: 128
Setter: ArtCam_SetSaturation	Getter: ArtCam_GetSaturation		

## Grayscale Filter Setting Possible Value

(DLL Ver.1280 or Up)

ARTCAM_FILTERTYPE_GRAY_MODE			
Set Grayscale Mode	Min:	0	Max: 2 Def: 0
Setter: ArtCam_SetGrayMode	Getter:	ArtCam_GetGrayMode	
0 = GRAY_NONE // Invalid Still Bayer arrangement			
1 = GRAY_BAYERCONVERT // To Bayer Arranement Add calculation by GRAY GAIN and GRAY OFFSET			
2 = GRAY_GRAYSCALE // After change color, leave only luminance information.			

ARTCAM_FILTERTYPE_GRAY_GAIN_R / GAIN_G1 / GAIN_G2 / GAIN_B			
Control each color's gain by a software	Min:	0	Max: 400 Def: 128
Setter: ArtCam_SetGrayGainRed	Getter: ArtCam_GetGrayGainRed		
Setter: ArtCam_SetGrayGainGreen1	Getter: ArtCam_GetGrayGainGreen1		
Setter: ArtCam_SetGrayGainGreen2	Getter: ArtCam_GetGrayGainGreen2		
Setter: ArtCam_SetGrayGainBlue	Getter: ArtCam_GetGrayGainBlue		

ARTCAM_FILTERTYPE_GRAY_OFFSET_R / OFFSET_G1 / OFFSET_G2 / OFFSET_B			
Control each color's offset by a software	Min:	-255	Max: 255 Def: 0
Setter: ArtCam_SetGrayOffsetRed	Getter: ArtCam_GetGrayOffsetRed		
Setter: ArtCam_SetGrayOffsetGreen1	Getter: ArtCam_GetGrayOffsetGreen1		
Setter: ArtCam_SetGrayOffsetGreen2	Getter: ArtCam_GetGrayOffsetGreen2		
Setter: ArtCam_SetGrayOffsetBlue	Getter: ArtCam_GetGrayOffsetBlue		



\*\*\*\*\*  
\*\*\*\*\*  
ARTRAY Camera / Capture Module Software Developer Kit  
Dynamic Link Library for Windows,2000,XP,Vista,7

ARTRAY CO., LTD.  
4F Ueno Bldg,1-17-5 Kouenjikota,Suginami-ku, Tokyo 166-0002 Japan  
TEL: 03-3389-5488  
FAX: 03-3389-5486  
E-mail: sales@artray.us  
URL: www.artray.us