



Atik Cameras Control

SDK Developer's Guide

Date
14 May 2025

Release
2060



Table of Contents

1	HISTORY	4
2	SCOPE	5
3	INTRODUCTION	6
4	SECTION 1 – PREREQUISITES	7
4.1	WINDOWS	7
4.2	LINUX/ARM.....	7
5	SECTION 2 - DLL METHODS	9
5.1	DEVICE INFO	9
	ArtemisDeviceIsPresent	9
	ArtemisDeviceName.....	9
	ArtemisDeviceSerial	9
5.2	CONNECTING / DISCONNECT TO CAMERAS	10
	ArtemisConnect.....	10
	ArtemisDisconnect.....	10
	ArtemisIsConnected.....	10
	ArtemisRefreshDevicesCount.....	11
	ArtemisDeviceIsCamera.....	11
5.3	CAMERA INFO	11
	ArtemisCameraSerial.....	11
	ArtemisProperties	12
	ArtemisColourProperties.....	13
5.4	EXPOSURES.....	13
	ArtemisStartExposure	13
	ArtemisStartExposureMS	14
	ArtemisStopExposure.....	14
	ArtemisImageReady	14
	ArtemisCameraState	15
	ArtemisExposureTimeRemaining	15
	ArtemisDownloadPercent	16
	ArtemisGetImageData	16
	ArtemisImageBuffer	17
5.5	EXPOSURE SETTINGS	17
	ArtemisBin	17
	ArtemisGetBin	17
	ArtemisGetMaxBin	18
	ArtemisSubframe	18
	ArtemisSubframePos	19
	ArtemisSubframeSize	19
	ArtemisGetSubframe	19
	ArtemisSetPreview	20
5.6	COOLING	20
	ArtemisTemperatureSensorInfo	20
	ArtemisSetCooling	21
	ArtemisCoolingInfo	21
	ArtemisCoolerWarmUp	22
5.7	INTERNAL FILTER WHEEL	22
	ArtemisFilterWheelInfo	22
	ArtemisFilterWheelMove	23
5.8	EXTERNAL FILTER WHEEL (EFW)	23
	ArtemisEFWIsPresent	23
	ArtemisEFWGetDeviceDetails	23
	ArtemisEFWConnect	24



<i>ArtemisEFWDisconnect</i>	24
<i>ArtemisEFWGetDetails</i>	24
<i>ArtemisEFWNmrPosition</i>	25
<i>ArtemisEFWGetPosition</i>	25
<i>ArtemisEFWSetPosition</i>	25
5.9 GUIDING	26
<i>ArtemisGuide</i>	26
<i>ArtemisGuidePort</i>	26
<i>ArtemisPulseGuide</i>	26
<i>ArtemisStopGuiding</i>	27
5.10 ARTEMIS ERROR CODES	27
6 SECTION 3 – EXAMPLES	28
6.1 C++ EXAMPLE	28
6.2 DOTNET EXAMPLE	28
7 TECHNICAL SUPPORT CONTACT	29



1 History

Name	Date	Version	Updates
Pedro Morgado	28/03/2025	1.0.1	Updated .Net example
Diogo Ferreira	05/05/2025	1.0.2	Updated to new template (documentation)
Diogo Ferreira	08/05/2025	1.0.3	Added DLL methods with examples (documentation)



2 Scope

This guide describes how to use the Atik Software Development Kit (SDK) and its Application Programming interface (API) for controlling Atik Cameras. It was designed specifically for Atik camera models. It includes all necessary libraries for Windows and Linux, providing support across these operating systems.



3 Introduction

The SDK is based on the Atik Cameras Dynamic-Link Libraries (DLLs) and allows you to easily integrate a customer-specific application to interact with Atik cameras. The main chapter of this document ([DLL Methods](#)) is organized into sections, with each section representing a specific camera feature or property. Within each section, subsections correspond to individual methods made available from this SDK API.



4 Section 1 – Prerequisites

To use the SDK effectively, several dependencies must be installed. First, the appropriate USB driver (the one provided in the SDK drivers folder) must be installed to ensure proper communication with the hardware. In addition, the SDK relies on C++ dependencies that must be present on the system (standard libraries).

For applications that require integration with .NET, the appropriate .NET runtime and development libraries must also be installed. This allows developers to build and run C# applications that interface with the SDK, providing cross-language flexibility.

4.1 Windows

To ensure the provided dynamic-link libraries (<SDK>\lib\win) are accessible by the system, they should be placed in the appropriate Windows directories depending on system architecture:

For 64-bit systems:

- Place 64-bit DLLs in C:\Windows\System32
- Place 32-bit DLLs in C:\Windows\SysWOW64

For 32-bit systems:

- Place 32-bit DLLs in C:\Windows\System32

Alternatively, if system-wide access is not required, the DLLs can be in the same directory as the executable file (.exe) to allow local loading.

These DLLs require the Microsoft Visual Studio 2017 (v141) C++ Redistributable as a minimum platform requirement. The latest supported redistributable packages are available from Microsoft's official site at <https://learn.microsoft.com/en-us/cpp/windows/latest-supported-vc-redist?view=msvc-170>. Since Visual Studio 2015, Microsoft has maintained ABI (Application Binary Interface) compatibility across toolset versions, making DLLs built with the 2017 toolset compatible with later versions such as Visual Studio 2019. Further information on binary compatibility can be found at <https://docs.microsoft.com/en-us/cpp/porting/binary-compat-2015-2017?view=msvc-160>.

4.2 Linux/ARM

To support Atik Cameras DLL in Linux/ARM, the *libatikcameras.so* must be added to the system and a few other steps may be necessary as well.

The following procedure was tested for Ubuntu 22.04.1

1) Install dependencies: g++, libusb and libudev:

```
sudo apt install g++
sudo apt install libusb-1.0-0-dev
sudo apt install libudev-dev
```

2) Load atik.rules:

Copy atik.rules file (<SDK>/lib/Linux/atik.rules) into /usr/lib/udev/rules.d/ to enable the use of the provided .so shared library without requiring administrative privileges.

```
sudo cp <SDK>/lib/Linux/atik.rules /usr/lib/udev/rules.d/
```

After placing the file, it is recommended to reload the udev rules using the command:

```
sudo udevadm control --reload-rules && sudo udevadm trigger
```

to apply the changes without needing to restart the system.



3) Copy the dynamically linked library to usr/lib folder

For 64b version:

```
sudo cp <SDK>/lib/Linux/64b/libatikcameras.so /usr/lib
```

4) Include the files from <SDK>/include into your SW.

5) Use the methods listed in [Section 2 - DLL Methods](#) to control the Atik Camera. Compilation should now be successful.

5 Section 2 - DLL Methods

This section provides an overview of all the methods defined in the SDK header file *AtikCameras.h*. Each subsection focuses on a specific topic and includes the corresponding methods related to that topic. For every method, a table is provided detailing its syntax, description, input parameters, and output.

5.1 Device Info

This section describes the methods related to the USB device.

ArtemisDeviceIsPresent

Method Syntax	<code>BOOL ArtemisDeviceIsPresent(int iDevice)</code>						
Description	Retrieves a bool value to indicate whether the given device is present						
Input Parameters	<table> <thead> <tr> <th>Name</th> <th>Type</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>iDevice</td> <td>int</td> <td>Device index</td> </tr> </tbody> </table>	Name	Type	Description	iDevice	int	Device index
Name	Type	Description					
iDevice	int	Device index					
Output	<table> <thead> <tr> <th>Type</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>bool</td> <td>True if the device is present, False otherwise</td> </tr> </tbody> </table>	Type	Description	bool	True if the device is present, False otherwise		
Type	Description						
bool	True if the device is present, False otherwise						

Example usage:

```
while (!ArtemisDeviceIsPresent(0))
    SleepMS(100);
```

ArtemisDeviceName

Method Syntax	<code>BOOL ArtemisDeviceName(int iDevice, char *pName)</code>									
Description	Retrieves the device's printable name									
Input Parameters	<table> <thead> <tr> <th>Name</th> <th>Type</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>iDevice</td> <td>int</td> <td>Device index</td> </tr> <tr> <td>pName</td> <td>char *</td> <td>Set as name of the correspondent device</td> </tr> </tbody> </table>	Name	Type	Description	iDevice	int	Device index	pName	char *	Set as name of the correspondent device
Name	Type	Description								
iDevice	int	Device index								
pName	char *	Set as name of the correspondent device								
Output	<table> <thead> <tr> <th>Type</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>bool</td> <td>True if iDevice is found, False otherwise</td> </tr> </tbody> </table>	Type	Description	bool	True if iDevice is found, False otherwise					
Type	Description									
bool	True if iDevice is found, False otherwise									

Example usage:

```
char name[100];
BOOL result = ArtemisDeviceName(iDevice, name);
```

ArtemisDeviceSerial

Method Syntax	<code>BOOL ArtemisDeviceSerial(int iDevice, char *pSerial)</code>									
Description	Retrieves the device's serial number									
Input Parameters	<table> <thead> <tr> <th>Name</th> <th>Type</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>iDevice</td> <td>int</td> <td>Device index</td> </tr> <tr> <td>pSerial</td> <td>char *</td> <td>Set as serial number of the correspondent device</td> </tr> </tbody> </table>	Name	Type	Description	iDevice	int	Device index	pSerial	char *	Set as serial number of the correspondent device
Name	Type	Description								
iDevice	int	Device index								
pSerial	char *	Set as serial number of the correspondent device								
Output	<table> <thead> <tr> <th>Type</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>bool</td> <td>True if iDevice is found, False otherwise</td> </tr> </tbody> </table>	Type	Description	bool	True if iDevice is found, False otherwise					
Type	Description									
bool	True if iDevice is found, False otherwise									



Example usage:

```
char serial[100];
BOOL result = ArtemisDeviceSerial(iDevice, serial);
```

5.2 Connecting / Disconnect to Cameras

This chapter includes methods used to establish and terminate connections with the cameras.

ArtemisConnect

Method Syntax	<code>ArtemisHandle ArtemisConnect (int iDevice)</code>		
Description	Connect to an Atik device by index and obtain an ArtemisHandle to it for further use		
Input Parameters	Name	Type	Description
	iDevice	int	Device index
Output	Type	Description	
	ArtemisHandle	The ArtemisHandle is a <i>void</i> and will be needed for all camera specific methods. It will return 0 if this method fails.	

Example usage:

```
auto handle = ArtemisConnect(0);
```

ArtemisDisconnect

Method Syntax	<code>BOOL ArtemisDisconnect (ArtemisHandle hCam)</code>		
Description	Disconnects from the device with that handle and invalidates the handle		
Input Parameters	Name	Type	Description
	hCam	ArtemisHandle	Artemis handle given by connect method
Output	Type	Description	
	bool	True if the disconnection was successful, False otherwise	

Example usage:

```
ArtemisDisconnect(handle);
```

ArtemisIsConnected

Method Syntax	<code>BOOL ArtemisIsConnected (ArtemisHandle hCam)</code>		
Description	Returns whether the handle is currently connected to a camera		
Input Parameters	Name	Type	Description
	hCam	ArtemisHandle	Artemis handle given by connect method
Output	Type	Description	
	bool	True if the camera is still connected, False otherwise	



Example usage:

```
ArtemisHandle hcam = ArtemisConnect(0); //Get hcm
if (ArtemisCameraState(hcam) == 0 && ArtemisIsConnected(hcam) != 0)
{
    //Start an exposure
```

ArtemisRefreshDevicesCount

Method Syntax	int ArtemisRefreshDevicesCount()	
Description	Updates the available device count	
Input Parameters	None	
Output	Type	Description
	int	Number of devices

Example usage:

```
int numberdevices = ArtemisRefreshDevicesCount();
```

ArtemisDeviceIsCamera

Method Syntax	BOOL ArtemisDeviceIsCamera(int iDevice)	
Description	Return whether the device at the specified index is a camera (and not E.G. a filter wheel device)	
Input Parameters	Name	Type
	iDevice	int
	Type	Description
	bool	True if the device is a camera, False otherwise

Example usage:

```
deviceInfo.deviceIsCamera = ArtemisDeviceIsCamera(i);
```

5.3 Camera Info

This section focusses on methods to retrieve information about the camera properties.

ArtemisCameraSerial

Method Syntax	int ArtemisCameraSerial(ArtemisHandle hCam, int* flags, int* serial)	
Description	Retrieves the serial number of the connected device	
Input Parameters	Name	Type
	hCam	ArtemisHandle
	flags	int *
	serial	int*
	Type	Description
	int	This method returns an ArtemisErrorCode (5.10)



Example usage:

```
int flags, serial;
int result = ArtemisCameraSerial(handle, &flags, &serial);
```

ArtemisProperties

Method Syntax	int ArtemisProperties(ArtemisHandle hCam, struct ARTEMISPROPERTIES *pProp)		
Description	Gets the connected camera's physical properties		
Input Parameters	Name	Type	Description
	hCam	ArtemisHandle	Artemis handle given by connect method
Output	Type	Description	
	int	This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
struct ARTEMISPROPERTIES properties;
int result = ArtemisProperties(handle, &properties);
```

ARTEMISCOLOURTYPE type contains the following properties:

Value	Description
int Protocol	The firmware version of the camera
int nPixelsX	The width of the sensor in pixels
int nPixelsY	The height of the sensor in pixels
float PixelMicronsX	The width of each pixel in microns
float PixelMicronsY	The height of each pixel in microns
int ccdflags	The value is '1' if the sensor is interlaced. 0 otherwise
int cameraflags	Represents the properties of the camera: 1 = Has FIFO 2 = Has External Trigger 4 = Can return preview data 8 = Camera can subsample 16 = Has Mechanical Shutter 32 = Has Guide Port 64 = Has GPIO capabilities 128 = Has Window Heater 256 = Can download 8-bit image 512 = Can Overlap exposure 1024 = Has Filter Wheel
char Description[40]	The name of the type of camera
char Manufacturer[40]	The manufacturer of the camera. Usually Atik Cameras

ArtemisColourProperties

Method Syntax	<code>int ArtemisColourProperties(ArtemisHandle hCam, ARTEMISCOLOURTYPE *colourType, int *normalOffsetX, int *normalOffsetY, int *previewOffsetX, int *previewOffsetY)</code>		
Description	Retrieves the colour properties of the connected device		
Input Parameters	Name	Type	Description
	hCam	ArtemisHandle	Artemis handle given by connect method
	*colourType	ARTEMISCOLOURTYPE	int pointer to get flags
	*normalOffsetX	int	Normal offset over the X axis
	*normalOffsetY	int	Normal offset over the Y axis
	*previewOffsetX	int	Preview offset over the X axis
Output	Type	Description	
	int	This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
ARTEMISCOLOURTYPE colourType;
int normalOffsetX, normalOffsetY, previewOffsetX, previewOffsetY;
int result = ArtemisColourProperties(handle, &colourType, &normalOffsetX, &normalOffsetY, &previewOffsetX, &previewOffsetY);
```

ARTEMISCOLOURTYPE type contains the following properties:

Value	Title	Description
0	ARTEMIS_COLOUR_UNKNOWN	Unknown colour type
1	ARTEMIS_COLOUR_NONE	No colour (Mono camera)
2	ARTEMIS_COLOUR_RGGB	Colour Camera (Bayer Matrix)

5.4 Exposures

This chapter describes the methods that can be used to control the exposure process as well as the methods implemented to get the result image.

ArtemisStartExposure

Method Syntax	<code>int ArtemisStartExposure(ArtemisHandle hCam, float Seconds)</code>		
Description	Begin an exposure for the device, specifying a duration as floating point seconds		
Input Parameters	Name	Type	Description
	hCam	ArtemisHandle	Artemis handle given by connect method
Output	Seconds	float	Seconds to perform the exposure
	int	This method returns an ArtemisErrorCode (5.10)	



Example usage:

```
ArtemisHandle handle = ArtemisConnect(0);

int tempResult = ArtemisStartExposure(handle, 0.02f);
```

ArtemisStartExposureMS

Method Syntax	int ArtemisStartExposureMS(ArtemisHandle hCam, int ms)		
Description Begin an exposure for the device, specifying a duration as milliseconds			
Input Parameters	Name	Type	
	hCam	ArtemisHandle	Artemis handle given by connect method
	ms	int	Miliseconds to perform the exposure
Output	Type	Description	
	int	This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
ArtemisStartExposureMS(handle, 50);
```

ArtemisStopExposure

Method Syntax	int ArtemisStopExposure(ArtemisHandle hCam)	
Description Stop the current exposure for the device		
Input Parameters	Name	Type
	hCam	ArtemisHandle
Output	Type	Description
	int	This method returns an ArtemisErrorCode (5.10)

Example usage:

```
tempResult = ArtemisStartExposure(handle, 1);
SleepMS(500);
ArtemisStopExposure(handle);
```

ArtemisImageReady

Method Syntax	bool ArtemisImageReady(ArtemisHandle hCam)	
Description Returns whether the image has been fully downloaded and is ready to access		
Input Parameters	Name	Type
	hCam	ArtemisHandle
Output	Type	Description
	bool	True if the image is ready, False otherwise



Example usage:

```
tempResult = ArtemisStartExposure(handle, 0.02f);
while (!ArtemisImageReady(handle))
{
    WriteLine("TimeRem: %.2f", ArtemisExposureTimeRemaining(handle));
    SleepMS(100);
}
```

ArtemisCameraState

Method Syntax	int ArtemisCameraState(ArtemisHandle hCam)	
Description Returns the device's state as an ARTEMISCAMERASTATE enumeration		
Input Parameters	Name hCam	Type ArtemisHandle Description Artemis handle given by connect method
Output	Type int	Description Camera state (values are listed below)

Example usage:

```
while (ArtemisCameraState(handle) != CAMERA_IDLE)
    SleepMS(100);
tempResult = ArtemisStartExposure(handle, 2);
```

The camera state values are present in the table below:

Value	Title	Description
0	CAMERA_IDLE	The default state of the camera. Indicates that the camera is ready to start and exposure
1	CAMERA_WAITING	This is a temporary state between receiving a start exposure command, and the exposure actually starting.
2	CAMERA_EXPOSING	The camera is exposing
4	CAMERA_DOWNLOADING	The service is reading the image off the sensor
5	CAMERA_FLUSHING	Stop Exposure has been called, and we are waiting for everything to stop. Some cameras will still need to clear the sensor, so this can take a while.
-1	CAMERA_ERROR	Something has gone wrong with the camera

ArtemisExposureTimeRemaining

Method Syntax	float ArtemisExposureTimeRemaining(ArtemisHandle hCam)	
Description Returns how much time is left in the exposure as a floating point number		
Input Parameters	Name hCam	Type ArtemisHandle Description Artemis handle given by connect method
Output	Type float	Description Time in seconds



Example usage:

```
while (!ArtemisImageReady(handle))
{
    WriteLine("TimeRem: %.2f", ArtemisExposureTimeRemaining(handle));
    SleepMS(100);
}
```

ArtemisDownloadPercent

Method Syntax	int ArtemisDownloadPercent(ArtemisHandle hCam)	
Description	Returns the download progress in percent	
Input Parameters	Name	Type
	hCam	ArtemisHandle
		Artemis handle given by connect method
Output	Type	Description
	int	This method returns an ArtemisErrorCode (5.10)

Example usage:

```
int downloadpercent = ArtemisDownloadPercent(handle);
```

ArtemisGetImageData

Method Syntax	int ArtemisGetImageData(ArtemisHandle hCam, int *x, int *y, int *w, int *h, int *binx, int *biny)	
Description	Returns a pointer to the image buffer which contains the latest captured image	
Input Parameters	Name	Type
	hCam	ArtemisHandle
		Artemis handle given by connect method
	*x	int
		The x start position of the image. Default value is 0
	*y	int
		The y start position of the image. Default value is 0
	*w	int
		The width of the image
	*h	int
		The height of the image
	*binx	int
		The x-bin value
	*biny	int
		The y-Bin value.
Output	Type	Description
	int	This method returns an ArtemisErrorCode (5.10)

Example usage:

```
while (!ArtemisImageReady(handle))
    SleepMS(100);

int x, y, w, h, binx, biny;
ArtemisGetImageData(handle, &x, &y, &w, &h, &binx, &biny);
```



ArtemisImageBuffer

Method Syntax	void* ArtemisImageBuffer(ArtemisHandle hCam)		
Description	Returns a pointer to the latest captured image buffer		
Input Parameters	Name hCam	Type ArtemisHandle	Description Artemis handle given by connect method
Output	Type void*	Description Pointer to the image buffer; returns 0 if no image is available	

Example usage:

```
int x, y, w, h, binx, biny;  
ArtemisGetImageData(handle, &x, &y, &w, &h, &binx, &biny);  
  
unsigned short* rawData = new unsigned short[w * h];  
  
auto imageBuffer = (unsigned short*)ArtemisImageBuffer(handle);
```

5.5 Exposure Settings

This section focusses on the exposures features like binning and sub frame.

ArtemisBin

Method Syntax	int ArtemisBin(ArtemisHandle hCam, int x, int y)		
Description	Sets the binning for the device		
Input Parameters	Name hCam	Type ArtemisHandle	Description Artemis handle given by connect method
	x	int	X binning value
	y	int	Y binning value
Output	Type int	Description This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
ArtemisBin(hcam, 1, 1);
```

ArtemisGetBin

Method Syntax	int ArtemisGetBin(ArtemisHandle hCam, int *x, int *y)		
Description	Gets the current binning values of the device		
Input Parameters	Name hCam	Type ArtemisHandle	Description Artemis handle given by connect method
	*x	int	Current X binning value
	*y	int	Current Y binning value
Output	Type int	Description This method returns an ArtemisErrorCode (5.10)	



Example usage:

```
int x, y;  
int result = ArtemisGetBin(handle, &x, &y);
```

ArtemisGetMaxBin

Method Syntax	int ArtemisGetMaxBin(ArtemisHandle hCam, int *x, int *y)		
Description	Gets the current maximum binning values of the device		
Input Parameters	Name	Type	Description
	hCam	ArtemisHandle	Artemis handle given by connect method
	*x	int	Current maximum X binning value
Output	Type	Description	
	int	This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
int x, y;  
int result = ArtemisGetMaxBin(handle, &x, &y);
```

ArtemisSubframe

Method Syntax	int ArtemisSubframe(ArtemisHandle hCam, int x, int y, int w, int h)		
Description	Sets the device subframe position and size		
Input Parameters	Name	Type	Description
	hCam	ArtemisHandle	Artemis handle given by connect method
	x	int	X offset of the subframe region
	y	int	Y offset of the subframe region
	w	int	Width of the subframe region
Output	Type	Description	
	int	This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
ArtemisHandle handle = ArtemisConnect(0);  
  
ArtemisSubframe(handle, 0, 0, 3508, 2200);  
  
int tempResult = ArtemisStartExposure(handle, 300);
```

ArtemisSubframePos

Method Syntax	int ArtemisSubframePos(ArtemisHandle hCam, int x, int y)	
Description	Sets the X and Y position of the subframe	
Input Parameters	Name	Type
	hCam	ArtemisHandle
	x	int
Output	Type	Description
	int	This method returns an ArtemisErrorCode (5.10)

Example usage:

```
int x = 200, y = 300;
ArtemisSubframePos(handle, x, y);
```

ArtemisSubframeSize

Method Syntax	int ArtemisSubframeSize(ArtemisHandle hCam, int w, int h)	
Description	Sets the width and height of the subframe	
Input Parameters	Name	Type
	hCam	ArtemisHandle
	w	int
Output	Type	Description
	int	This method returns an ArtemisErrorCode (5.10)

Example usage:

```
w = 2000; h = 1500;
ArtemisSubframeSize(handle, w, h);
```

ArtemisGetSubframe

Method Syntax	int ArtemisGetSubframe(ArtemisHandle hCam, int *x, int *y, int *w, int *h)	
Description	Gets the current sub framing setting for the device	
Input Parameters	Name	Type
	hCam	ArtemisHandle
	x	int*
	y	int*
	w	int*
Output	Type	Description
	int	This method returns an ArtemisErrorCode (5.10)

Example usage:

```
int x, y, w, h;
int result = ArtemisGetSubframe(handle, &x, &y, &w, &h);
```

ArtemisSetPreview

Method Syntax	int ArtemisSetPreview(ArtemisHandle hCam, bool bPrev)		
Description	Preview mode will produce images at a faster rate, but at a cost of quality. This method is used to set the camera into normal / preview mode.		
Input Parameters	Name	Type	
	hCam	ArtemisHandle	Artemis handle given by connect method
	bPrev	bool	If True sets preview mode, otherwise sets normal mode
Output	Type	Description	
	int	This method returns an ArtemisErrorCode (5.10)	

Example:

```
ArtemisHandle hcam = ArtemisConnect(0); //Get hcam
if (ArtemisCameraState(hcam) == 0 && ArtemisIsConnected(hcam) != 0)
{
    //Start an exposure
    ArtemisSubFrame(hcam, 0, 0, 658, 492);
    ArtemisSetPreview(hcam, 0);
    ArtemisBin(hcam, 1, 1);
```

5.6 Cooling

This section describes methods related to the temperature sensor (cooling). The methods below can be used with every Atik camera model except Infinity model.

ArtemisTemperatureSensorInfo

Method Syntax	int ArtemisTemperatureSensorInfo(ArtemisHandle hCam, int sensor, int* temperature)		
Description	Gets temperature of the device for the specific sensor index, in hundreds of degrees centigrade. Note: If the method is called with sensor parameter set to 0, then temperature will actually be set to the number of sensors		
Input Parameters	Name	Type	
	hCam	ArtemisHandle	Artemis handle given by connect method
	sensor	int	The index of the sensor to obtain information from
	temperature	int*	Set to the temperature in tenths of degrees celsius, or the number of temperature sensors on the device
Output	Type	Description	
	int	This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
for (int i = 0; i < 300; i++)
{
    int tempInt;
    ArtemisTemperatureSensorInfo(handle, 1, &tempInt);

    double temp = 0.01 * tempInt;
    WriteLine("Temp: %.2f", temp);

    if (temp < 0)
        break;

    SleepMS(1000);
}
```

ArtemisSetCooling

Method Syntax	int ArtemisSetCooling(ArtemisHandle hCam, int setpoint)	
Description	Sets the cooling target temperature of the device. To set the cooling to -10C, you need to call the function with setpoint = -1000	
Input Parameters	Name	Type
	hCam	ArtemisHandle
Output	Type	Description
	int	This method returns an ArtemisErrorCode (5.10)

Example usage:

```
ArtemisSetCooling(handle, -10);
```

ArtemisCoolingInfo

Method Syntax	int ArtemisCoolingInfo(ArtemisHandle hCam, int* flags, int* level, int* minlvl, int* maxlvl, int* setpoint)		
Description	Gives the current state of the cooling		
Input Parameters	Name	Type	
	hCam	ArtemisHandle	Artemis handle given by connect method
	flags	int*	internal flags (cooling state)
	level	int*	The power level of the cooler, usually from 0 to 255
	minlvl	int*	Minimum cooling power level
	maxlvl	int*	Maximum cooling power level
Output	Type	Description	
	int	This method returns an ArtemisErrorCode (5.10)	



Example usage:

```
int level, minlvl, maxlvl, setpoint;
auto result = ArtemisCoolingInfo(handle, &flags, &level, &minlvl, &maxlvl, &setpoint);
```

ArtemisCoolerWarmUp

Method Syntax	int ArtemisCoolerWarmUp(ArtemisHandle hCam)		
Description	Tells the camera to start warming up. Note: It is very important that this function is called at the end of operation. Letting the sensor warm up naturally can cause damage to the sensor. It's not unusual for the temperature to go further down before going up.		
Input Parameters	Name hCam	Type ArtemisHandle	Description Artemis handle given by connect method
Output	Type int	Description This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
int result = ArtemisCoolerWarmUp(handle);
```

5.7 internal Filter Wheel

This section focuses on the methods used to control the camera internal filter wheel. The methods below can only be used by certain Atik camera models: Atik One, Artemis VS with the suffix "F" (e.g., VS60-M-F), QSI 700 and ChemiMOS 9f.

ArtemisFilterWheelInfo

Method Syntax	int ArtemisFilterWheelInfo(ArtemisHandle hCam, int *numFilters, int *moving, int *currentPos, int *targetPos)		
Description	Retrieve the state of the filter wheel		
Input Parameters	Name hCam	Type ArtemisHandle	Description Artemis handle given by connect method
	numFilters	Type int*	Number of filters in the filter wheel
	moving	Type int*	Set to 1 if the filter wheel is moving, 0 otherwise
	currentPos	Type int*	Set to the current position
	targetPos	Type int*	Set to the target position
Output	Type int	Description This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
int numFilters, moving, currentPos, targetPos;
int result = ArtemisFilterWheelInfo(handle, &numFilters, &moving, &currentPos, &targetPos);
```

ArtemisFilterWheelMove

Method Syntax	<code>int ArtemisFilterWheelMove(ArtemisHandle hCam, int targetPos)</code>		
Description	Move the filter wheel to the desired location		
Input Parameters	Name	Type	Description
	hCam	ArtemisHandle	Artemis handle given by connect method
Output	Type	Description	
	int	This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
targetPosition = 2;
ArtemisFilterWheelMove(handle, targetPosition);
```

5.8 External Filter Wheel (EFW)

As the filter wheel can be external to the camera. This section focusses on methods that can control an external filter wheel.

ArtemisEFWIsPresent

Method Syntax	<code>bool ArtemisEFWIsPresent(int i)</code>		
Description	Checks whether a filter wheel is present at the index		
Input Parameters	Name	Type	Description
	i	ArtemisHandle	The index to check
Output	Type	Description	
	bool	TRUE if present, FALSE if not	

ArtemisEFWGetDeviceDetails

Method Syntax	<code>int ArtemisEFWGetDeviceDetails(int i, ARTEMISEFWTYPE * type, char * serialNumber)</code>		
Description	Get information on the filter wheel device at the specified index		
Input Parameters	Name	Type	Description
	i	int	Index to get details for
	type	ARTEMISEFWTYPE *	Set to the type of the filter wheel device
Output	Type	Description	
	int	TRUE if present, FALSE if not	



ArtemisEFWConnect

Method Syntax	<code>ArtemisHandle ArtemisEFWConnect(int i)</code>		
Description	Connects to the given filter wheel. The ArtemisHandle is a void * and will be needed for all filter wheel specific methods. This method can be call with i = -1, in this case you will get the first filter wheel that is not currently in use (by any application). Note: It is possible for different applications to connect to the same filter wheel.		
Input Parameters	Name i	Type int	Description Index of the filter wheel
Output	Type ArtemisHandle	Description It will return 0 if this method fails.	

Example usage:

```
while (!ArtemisEFWIsPresent(0))
{
    char* serial1 = (char*)calloc(16, sizeof(char));
    ARTEMISEFWTYPE fw1Type;
    ArtemisEFWGetDeviceDetails(0, &fw1Type, serial1);
    SleepMS(100);

}
ArtemisHandle handle = ArtemisEFWConnect(0);
```

ArtemisEFWDisconnect

Method Syntax	<code>int ArtemisEFWDisconnect(ArtemisHandle handle)</code>		
Description	This method is used to release the filter wheel when done. It will allow other applications to connect to the filter wheel (using -1) as listed above.		
Input Parameters	Name handle	Type ArtemisHandle	Description Filter wheel handle
Output	Type int	Description This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
ArtemisEFWDisconnect(handle);
```

ArtemisEFWGetDetails

Method Syntax	<code>int ArtemisEFWGetDetails(ArtemisHandle handle, ARTEMISEFWTYPE * type, char * serialNumber)</code>		
Description	Get information on the connected filter wheel device		
Input Parameters	Name handle	Type ArtemisHandle	Description Filter wheel handle
	type	Type ARTEMISEFWTYPE*	Set to type of filter wheel device
	serialNumber	Type char*	Sets serial number of the filter wheel
Output	Type int	Description This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
ARTEMISEFWTYPE type;
char serialNumber[100];
int result = ArtemisEFWGetDetails(handle, &type, serialNumber);
```

ArtemisEFWNmrPosition

Method Syntax	<code>int ArtemisEFWNmrPosition(ArtemisHandle handle, int * nPosition)</code>	
Description	Gets the number of filters inside the filter wheel	
Input Parameters	Name	Type
	handle	ArtemisHandle
Output	Type	Description
	int	This method returns an ArtemisErrorCode (5.10)

Example usage:

```
int nPosition;
int result = ArtemisEFWNmrPosition(handle, &nPosition);
```

ArtemisEFWGetPosition

Method Syntax	<code>int ArtemisEFWGetPosition(ArtemisHandle handle, int * iPosition, bool * isMoving)</code>		
Description	Gets the device's filter wheel position		
Input Parameters	Name	Type	
	handle	ArtemisHandle	Filter wheel handle
	iPosition	int*	Set to the filter wheel's current position
Output	Type	Description	
	bool*	Set to whether the filter wheel is currently moving	
	int	This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
int nPosition;
bool isMoving;
int result = ArtemisEFWGetPosition(handle, &nPosition, &isMoving);
```

ArtemisEFWSetPosition

Method Syntax	<code>int ArtemisEFWSetPosition(ArtemisHandle handle, int iPosition)</code>	
Description	Sets the device's filter wheel position	
Input Parameters	Name	Type
	handle	ArtemisHandle
Output	Type	Description
	int	Target position
	int	This method returns an ArtemisErrorCode (5.10)



Example usage:

```
int iPosition = 2;  
int result = ArtemisEFWSetPosition(handle, iPosition);
```

5.9 Guiding

This section describes the functions that can control the guiding feature. The methods below can only be applied to certain Atik camera models: Atik One, Atik Infinity, Atik Series 3, Artemis VS.

ArtemisGuide

Method Syntax	int ArtemisGuide(ArtemisHandle hCam, int axis)	
Description	Allows you to set the guiding for a specific direction	
Input Parameters	Name	Type
	hCam	ArtemisHandle
Output	Type	Description
	int	1=North, 2=South, 3=East, 4=West
Output	Type	Description
	int	This method returns an ArtemisErrorCode (5.10)

Example usage:

```
axis = 1;  
int result = ArtemisGuide(handle, axis);
```

ArtemisGuidePort

Method Syntax	int ArtemisGuidePort(ArtemisHandle hCam, int nibble)	
Description	Allows you to set the guiding in multiple directions at the same time	
Input Parameters	Name	Type
	hCam	ArtemisHandle
Output	Type	Description
	int	1=North, 2=South, 3=East, 4=West
Output	Type	Description
	int	This method returns an ArtemisErrorCode (5.10)

Example usage:

```
int nibble = 1;  
int result = ArtemisGuidePort(handle, nibble);
```

ArtemisPulseGuide

Method Syntax	int ArtemisPulseGuide(ArtemisHandle hCam, int axis, int milli)	
Description	Allows you to set the guiding for a specific direction for a set amount of time	
Input Parameters	Name	Type
	hCam	ArtemisHandle
Output	Type	Description
	int	1=North, 2=South, 3=East, 4=West
Output	Type	Description
	int	Pulse duration (ms) for the selected axis
Output	Type	Description
	int	This method returns an ArtemisErrorCode (5.10)

Example usage:

```
int axis = 1, milli = 50;
int result = ArtemisPulseGuide(handle, axis, milli);
```

ArtemisStopGuiding

Method Syntax	int ArtemisStopGuiding(ArtemisHandle hCam)	
Description	Stops the guiding	
Input Parameters	Name	Type
	hCam	ArtemisHandle
Output	Type	
	int	
	This method returns an ArtemisErrorCode (5.10)	

Example usage:

```
int result = ArtemisStopGuiding(handle);
```

5.10 Artemis Error Codes

This section lists all Artemis error codes:

Value	Title	Description
0	ARTEMIS_OK	The function call has been successful
1	ARTEMIS_INVALID_PARAMETER	One or more of the parameters supplied are inconsistent with what was expected. One example of this would be calling any camera function with an invalid ArtemisHandle. Another might be to set a subframe to an unreachable area
2	ARTEMIS_NOT_CONNECTED	Returned when calling a camera function on a camera which is no longer connected.
3	ARTEMIS_NOT_IMPLEMENTED	Returned for functions that are no longer used.
4	ARTEMIS_NO_RESPONSE	Returned if a function times out for any reason
5	ARTEMIS_INVALID_FUNCTION	Returned when trying to call a camera specific function on a camera which doesn't have that feature. (Such as the cooling functions on cameras without cooling).
6	ARTEMIS_NOT_INITIALISED	Returned if trying to call a function on something that hasn't been initialised. The only current example is the lens control
7	ARTEMIS_OPERATION_FAILED	Returned if a function couldn't complete for any other reason.



6 Section 3 – Examples

The examples folder included in the SDK is designed to help developers quickly understand and implement the Atik Cameras SDK in their own applications. It provides two reference implementations (one in C++ and one in .NET) demonstrating practical usage of the SDK's core functionality. These examples provide a foundational reference for integrating Atik Cameras control into both native (C++) and managed (.NET) environments, helping to simplify development and accelerate the integration process.

6.1 C++ Example

The provided C++ example is a console application designed to demonstrate how Atik Cameras methods can be utilized within a C++ project. The main source file, `AtikCamerasSDKApp.cpp`, illustrates the procedures required to interface with the Atik Cameras SDK. This example covers the following key operations:

- Connect/disconnect
- Start exposure
- Retrieving image data
- Configuring binning settings
- Getting and setting subframe parameters, including size and position
- Controlling camera cooling (e.g., setting the target temperature, warming up, and reading the current temperature)
- Operating the filter wheel
- Executing guiding commands

This example serves as a practical reference for implementing Atik Cameras functionality in your own C++ applications.

i) To build it:

```
cd <SDK>/example/C++ Example/AtikCamerasSDKExample  
make
```

ii) To execute it:

```
bin/atiksdkexample.out
```

6.2 DotNet Example

The provided .NET example is a Windows Forms application that demonstrates how to integrate the Atik Cameras SDK into .NET-based software. The main interface logic is implemented in the `Form1.cs` file, which acts as the bridge between the user interface and the Atik Cameras SDK. This application showcases the following key functionalities:

- Connecting to and disconnecting from the camera
- Start/stop exposures
- Retrieving image data
- Accessing temperature sensor information
- Enabling and using Fast Mode
- Executing cooling-related operations

This example serves as a useful reference for developers looking to incorporate Atik Camera support into their .NET applications.



7 Technical Support Contact

For any issues encountered with the SDK, please contact Atik technical support at:

support@atik-cameras.com