



# AVT GigE

## Camera and Driver Attributes

Firmware 1.38  
March 10, 2010

## **Notes**

- 1) This is a master control document for all camera models. Not all attributes are available on all cameras. For PvAPI users, see the *PvAttrExists* call.
- 2) For PvAPI users, attribute type is given: enum, float32, uint32, string, or command. See the corresponding *PvAttrEnum\_\_\_\_*, *PvAttrFloat32\_\_\_\_*, *PvAttrUint32\_\_\_\_*, *PvAttrString\_\_\_\_*, *PvCommandRun* calls.
- 3) R/W = attribute is read/write. R/C = attribute is read only and constant. R/V = attribute is read only and volatile, can change at any time.

# Camera Attributes

## Acquisition

### Trigger

#### AcqEnd

##### AcqEndTriggerEvent – Enum – R/W

What type of external input trigger will end acquisition.

<i>EdgeRising</i>	rising edge trigger
<i>EdgeFalling</i>	falling edge trigger
<i>EdgeAny</i>	rising or falling edge
<i>LevelHigh</i>	active high signal
<i>LevelLow</i>	active low signal

##### AcqEndTriggerMode – Enum – R/W

Selects if the end of acquisition should be stimulated by an external hardware trigger. See the AcquisitionStop command for software triggering.

<i>SyncIn1</i>	trigger at SyncIn1 to be associated with this control
<i>SyncIn2</i>	trigger at SyncIn2 to be associated with this control
<i>Disabled</i>	an external trigger does not control end of acquisition

#### AcqRec

An AcqStart hardware trigger signal, or the AcquisitionStart command, must be received before your AcqRec trigger. See AcquisitionMode = Recorder.

##### AcqRecTriggerEvent – Enum – R/W

What kind of external input trigger will start a recording sequence when AcquisitionMode set to Recorder.

<i>EdgeRising</i>	rising edge trigger
<i>EdgeFalling</i>	falling edge trigger
<i>EdgeAny</i>	rising or falling edge
<i>LevelHigh</i>	active high signal
<i>LevelLow</i>	active low signal

##### AcqRecTriggerMode – Enum – R/W

Selects if the start of a Recorder event should be stimulated by an external hardware trigger. There is no software trigger event capability for this mode.

<i>SyncIn1</i>	trigger at SyncIn1 to be associated with this control
<i>SyncIn2</i>	trigger at SyncIn2 to be associated with this control

*Disabled* an external trigger does not control the start of an event

## AcqStart

### AcqStartTriggerEvent – Enum – R/W

What kind of external input trigger will stimulate the start of an acquisition.

<i>EdgeRising</i>	rising edge trigger
<i>EdgeFalling</i>	falling edge trigger
<i>EdgeAny</i>	rising or falling edge
<i>LevelHigh</i>	active high signal
<i>LevelLow</i>	active low signal

### AcqStartTriggerMode – Enum – R/W

Selects if the start of acquisition should be stimulated by an external hardware trigger. See the AcquisitionStart command for software triggering.

<i>SyncIn1</i>	trigger at SyncIn1 to be associated with this control
<i>SyncIn2</i>	trigger at SyncIn2 to be associated with this control
<i>Disabled</i>	an external trigger does not control start of acquisition

### FrameRate – Float32 – R/W

When FrameStartTriggerMode is set to FixedRate, this control specifies the frame rate.

## FrameStart

Controls relating to the triggering of frames within an acquisition.

### FrameStartTriggerDelay – Uint32 – R/W

Start-of-image is delayed FrameStartTriggerDelay microseconds after receiving an external trigger event. This feature is only valid when FrameStartTriggerMode is set to external trigger (i.e. SyncIn1, SyncIn2). Useful when using a common trigger to synch with a strobe lighting source, which will inherently have some fixed setup time

### FrameStartTriggerEvent – Enum – R/W

The external trigger can be configured to accept various trigger event types:

<i>EdgeRising</i>	rising edge trigger
<i>EdgeFalling</i>	falling edge trigger
<i>EdgeAny</i>	rising or falling edge
<i>LevelHigh</i>	active high signal
<i>LevelLow</i>	active low signal

### FrameStartTriggerMode – Enum – R/W

Determines how an image frame is initiated within an acquisition.

<i>Freerun</i>	camera runs at maximum supported frame rate depending on the exposure time and region of interest size.
<i>SyncIn1</i>	external trigger SyncIn1
<i>SyncIn2</i>	external trigger SyncIn2
<i>FixedRate</i>	camera self-triggers at a fixed frame rate defined by FrameRate.
<i>Software</i>	software initiated image capture.

## FrameStartTriggerSoftware – Command

Triggers a frame within acquisition stream. Valid when:  
FrameStartTriggerMode = Software.

## AcquisitionAbort – Command

Aborts current acquisition.

## AcquisitionFrameCount – Uint32 – R/W

Define the number of frames to capture when capturing a limited sequence of images. Used in combination with MultiFrame and Recorder acquisition modes.

## AcquisitionMode – Enum – R/W

The acquisition modes, which determine how the camera handles frame triggers within the acquisition stream.

<i>Continuous</i>	After an acquisition start event, the camera will continuously receive frame trigger events, or in the case where FrameStartTriggerMode equals Freerun, will continuously stream. This is the normal acquisition mode of the camera.
<i>SingleFrame</i>	After an acquisition start event, the camera will only receive a single frame trigger event. Further trigger events will be ignored until acquisition is stopped and restarted.
<i>MultiFrame</i>	After an acquisition start event, the camera will receive AcquisitionFrameCount number of triggers. Further trigger events will be ignored until acquisition is stopped and restarted.
<i>Recorder</i>	After an acquisition start event, the camera will continuously capture images into the camera on-board memory, but will not send them to the host until an AcqRec trigger signal is received. Further AcqRec trigger events will be ignored until acquisition is stopped and restarted.

Combined with the RecorderPreEventCount control, this feature is useful for returning any number of frames **before** a trigger event.

When AcqRec trigger is received, the currently imaging/aquiring image will complete as normal, and then at least one more image will be taken. See RecorderPreEventCount.

The memory is a circular buffer, that is, once it is full starts rewriting images. Its size is determined by AcquisitionFrameCount.

## AcquisitionStart – Command

Starts acquisition stream.

### AcquisitionStop – Command

Stops acquisition stream.

### RecorderPreEventCount – Uint32 – R/W

The number of images returned before the AcqRec trigger event, with AcquisitionFrameCount minus RecorderPreEventCount images being returned after the trigger event. Valid only when AcquisitionMode equals Recorder.

Note: at least one image must be captured after the AcqRec trigger event. That is, you cannot set RecorderPreEventCount = 1, AcquisitionFrameCount = 1.

## ConfigFile

Prosilica's GigE cameras are capable of storing a number of user-specified configurations within the camera's non-volatile memory. These saved configurations can be used to define the power-up settings of the camera or to quickly switch between a number of predefined settings.

### ConfigFileIndex – Enum – R/W

The index number corresponding to the configuration set that you are currently working with. Possible values: *Factory, 1, 2, 3, 4, 5*.

### ConfigFileLoad – Command

Loads settings saved in camera non-volatile memory indicated by ConfigFileIndex to the current camera settings.

### ConfigFilePowerup – Enum – R/W

The saved configuration loaded when the camera powers up. Possible values: *Factory, 1, 2, 3, 4, 5*.

### ConfigFileSave – Command

Saves the current camera settings to camera non-volatile memory indicated by ConfigFileIndex. The Factory setting cannot be overwritten.

## Controls

### DSP

The automatic exposure, gain, and WhiteBalance features can be configured to respond only to a subregion within the image scene. This feature can be used to choose a subregion that will

'meter' the rest of the image. This feature works like the region metering on a photographic camera.

### DSPSubregionBottom – Uint32 – R/W

Defines the bottom edge of the DSP subregion. Measured in pixels from the top edge of the full image. Note: Currently defaults to a huge number much larger than the maximum number of sensor rows.

### DSPSubregionLeft – Uint32 – R/W

Defines the position of left edge of the DSP subregion. Measured in pixels from the left edge of the full image. Defaults to zero.

### DSPSubregionRight – Uint32 – R/W

Defines the right edge of the DSP subregion. Measured in pixels from the left edge of the full image. Note: Currently defaults to a huge number much larger than the maximum number of sensor columns.

### DSPSubregionTop – Uint32 – R/W

Defines the top edge of the DSP subregion. Measured in pixels from the top edge of the full image. Defaults to zero.

## DefectMask

This feature is only available on the GE4000 and GE4900 cameras. The standard model of these cameras use Class 2 sensors which can have column defects. The DefectMask replaces defective columns with interpolated values based on neighboring columns. Class 1 and Class 0 sensors are available for these cameras which do not require any column masking.

### DefectMaskColumnEnable – Enum – R/W

Possible values: *On, Off.*

## Exposure

### Auto

This group of controls relates to the camera auto-exposure function.

Note: The camera must be acquiring images in order for the auto exposure algorithm to update.

### ExposureAutoAdjustTol – Uint32 – R/W

In percent, from 0 to 50. Sets a tolerance in variation from ExposureAutoTarget in which the auto exposure algorithm will not respond. Can be used to limit exposure setting changes to only larger variations in scene lighting.

## ExposureAutoAlg – Enum – R/W

The following algorithms can be used to calculate auto-exposure:

<i>Mean</i>	The arithmetic mean of the histogram of the current image is compared to ExposureAutoTarget, and the next image adjusted in exposure time to meet this target. Bright areas are allowed to saturate.
<i>FitRange</i>	The histogram of the current image is measured, and the exposure time of the next image is adjusted so bright areas are not saturated. Generally, the Mean setting is preferred.

## ExposureAutoMax – Uint32 – R/W

In microseconds. This sets the upper bound to the exposure setting in autoexposure mode. This is useful in situations where frame rate is important. This value would normally be set to something less than  $1 \times 10^6 / (\text{desired frame rate})$ .

## ExposureAutoMin – Uint32 – R/W

In microseconds. This sets the lower bound to the exposure setting in autoexposure mode.

## ExposureAutoOutliers – Uint32 – R/W

Each unit represents 0.01%. When value is 1000, this equals 10%. The percentage defines the total pixels from top of the distribution that are ignored by the auto exposure algorithm.

## ExposureAutoRate – Uint32 – R/W

In percent. Determines the rate at which the auto exposure function changes the exposure setting.

## ExposureAutoTarget – Uint32 – R/W

In percent. Controls the general lightness or darkness of the auto exposure feature; specifically the target mean histogram level of the image, 0 being black, 100 being white.

## ExposureMode – Enum – R/W

<i>Manual</i>	The camera exposure time is fixed by ExposureValue parameter.
<i>Auto</i>	The exposure time will vary continuously according to the scene illumination. The Auto exposure function operates according to the Auto and DSP controls



<i>AutoOnce</i>	A command. The exposure will be set once according to the scene illumination and then remain at that setting even when the scene illumination changes. The AutoOnce function operates according to the Auto and DSP controls.
<i>External</i>	When ExposureMode is set to External the exposure time will be controlled by an external signal appearing on SyncIn1 or SyncIn2. In order for this feature to work, the parameter FrameStartTriggerMode must be set to SyncIn1 or SyncIn2.

## ExposureValue – Uint32 – R/W

In microseconds. The sensor integration time. 15000 corresponds to 15 ms integration time, 1000 corresponds to 1ms, etc.

## Gain

### Auto

This group of controls relates to the camera auto gain function.

Note: The camera must be acquiring images in order for the auto gain algorithm to update.

## GainAutoAdjustTol – Uint32 – R/W

In percent, from 0 to 50. Sets a tolerance in variation from GainAutoTarget in which the auto exposure algorithm will not respond. Can be used to limit gain setting changes to only larger variations in scene lighting.

## GainAutoMax – Uint32 – R/W

In dB. Sets the upper bound to the gain setting in Auto gain mode.

## GainAutoMin – Uint32 – R/W

In dB. Sets the lower bound to the gain setting in Auto gain mode. Normally this number would be set to zero.

## GainAutoOutliers – Uint32 – R/W

Each unit represents 0.01%. When value is 1000, this equals 10%. The percentage defines the total pixels from top of the distribution that are ignored by the auto gain algorithm.

## GainAutoRate – Uint32 – R/W

In percent. Determines the rate at which the auto gain function changes the gain setting.

## GainAutoTarget – Uint32 – R/W

In percent. Controls the general lightness or darkness of the Auto gain feature. A percentage of the maximum GainValue.

### GainMode – Enum – R/W

<i>Manual</i>	The camera gain is fixed by GainValue parameter.
<i>Auto</i>	The gain will vary continuously according to the scene illumination. The Auto function operates according to the Auto and DSP controls.
<i>AutoOnce</i>	A command. The gain will be set once according to the scene illumination and then remain at that setting even when the scene illumination changes. The AutoOnce function operates according to the Auto and DSP controls.

### GainValue – Uint32 – R/W

In dB.  $G_{dB} = 20 \log_{10}(V_{in}/V_{out})$ . The gain setting applied to the sensor. Default gain is zero, and gives the best image quality. However, in low light situations, it may be necessary to increase the gain setting.

## Iris

All video-type auto iris lenses have a default reference voltage. When a voltage larger than this reference voltage is applied to the lens, the iris closes. When a voltage is applied less than this reference voltage, the iris opens. The auto iris algorithm calculates the appropriate voltage, IrisVideoLevel, to apply to the lens, based on the brightness of the current image vs. the IrisAutoTarget.

Note: The camera must be acquiring images in order for the auto iris algorithm to update.

### IrisAutoTarget – Uint32 – R/W

In percent. Controls the general lightness or darkness of the auto iris feature; specifically the target mean histogram level of the image, 0 being black, 100 being white.

### IrisMode – Enum – R/W

Sets the auto-iris mode.

<i>Disabled</i>	Turn off the video auto-iris function.
<i>Video</i>	Turn on the video auto-iris function.
<i>VideoOpen</i>	Fully open the iris.
<i>VideoClosed</i>	Full close the iris.

### IrisVideoLevel – Uint32 – R/W

In 10 mV units. This attribute reports the strength of the video signal coming from the camera.

### IrisVideoLevelMax – Uint32 – R/W

In 10 mV units. Limits the maximum driving voltage for closing the lens iris. Typically this will be 150, however it may vary dependent on the lens reference voltage.

**IrisVideoLevelMin – Uint32 – R/W**

In 10 mV units. Limits the minimum driving voltage for opening the lens iris. Typically this will be 0.

## SubstrateVoltage

**VsubValue – Uint32 – R/W**

Substrate Voltage value (only for GX cameras) in mV units.

## WhiteBalance

### Auto

Controls the way that the Auto white balance function operates.

**WhitebalAutoAdjustTol – Uint32 – R/W**

A threshold. This parameter sets a range of scene color changes in which the automatic white balance will not respond. This parameter can be used to limit whitebalance setting changes to only larger variations in scene color.

**WhitebalAutoRate – Uint32 – R/W**

In percent. Determines how fast the Auto White balance updates.

**WhitebalMode – Enum – R/W**

<i>Manual</i>	Auto white balance is off. White balance can be adjusted directly by changing the WhitebalValueRed and WhitebalValueBlue parameters.
<i>Auto</i>	White balance will continuously adjust according to the current scene. The Auto function operates according to the Auto and DSP controls
<i>AutoOnce</i>	A command. The white balance will be set once according to the scene illumination and then remain at that setting even when the scene illumination changes. The AutoOnce function operates according to the Auto and DSP controls

**WhitebalValueRed – Uint32 – R/W**

Red gain expressed as a percentage of the camera default setting.

**WhitebalValueBlue – Uint32 – R/W**

Blue gain expressed as a percentage of the camera default setting.

## EventControls

### EventID

All the events supported by the camera:

EventAcquisitionStart – Uint32 – R/C

EventAcquisitionEnd – Uint32 – R/C

EventFrameTrigger – Uint32 – R/C

EventExposureEnd – Uint32 – R/C

EventAcquisitionRecordTrigger – Uint32 – R/C

EventSyncIn1Rise – Uint32 – R/C

EventSyncIn1Fall – Uint32 – R/C

EventSyncIn2Rise – Uint32 – R/C

EventSyncIn2Fall – Uint32 – R/C

EventSyncIn3Rise – Uint32 – R/C

EventSyncIn3Fall – Uint32 – R/C

EventSyncIn4Rise – Uint32 – R/C

EventSyncIn4Fall – Uint32 – R/C

EventNotification – Enum – R/W

Turns the selected event notification *On* or *Off*.

EventSelector – Enum – R/W

Select a specific event to be enabled or disabled using EventNotification

*AcquisitionStart*

*AcquisitionEnd*

*FrameTrigger*

*ExposureEnd*  
*AcquisitionRecordTrigger*  
*SyncIn1Rise*  
*SyncIn1Fall*  
*SyncIn2Rise*  
*SyncIn2Fal*  
*SyncIn3Ris*  
*SyncIn3Fall*  
*SyncIn4Rise*  
*SyncIn4Fall*

## EventsEnable1 – Uint32 – R/W

Bitmask of events currently enabled. Bit 1 is *EventAcquisitionStart*, Bit 2 is *EventAcquisitionEnd*, Bit 3 is *FrameTrigger*, and so on. This is an alternative to setting each of the event individually using the *EventNotification* and *EventSelector* method.

# GigE

## BandwidthCtrlMode – Enum32 – R/W

Select the desired mode of bandwidth control.

<i>StreamBytesPerSecond</i>	The default mode of bandwidth control. See the <i>StreamBytesPerSecond</i> control for more information.
<i>SCPD</i>	Stream channel packet delay expressed in timestamp counter units. This mode is not recommended.
<i>Both</i>	Implements a combination of control modes. This mode is not recommended.

## PacketSize – Uint32 – R/W

In Bytes. Determines the Ethernet packet size. Generally speaking this number should be set to as large as the network adaptor will allow. If this number is reduced, then CPU loading will increase. These large packet sizes are called Jumbo Packets/Frames in Ethernet terminology. If your GigE network adaptor does not support Jumbo Packets/Frames of at least 8228 Bytes (the camera default on power up), then you will need to reduce *PacketSize* parameter to match the maximum supported by your network adaptor.

A *PacketSize* of 1500 is a safe setting which all GigEthernet network cards support.

Note: If you are seeing all "black images", or all frames reported as *StatFramesDropped* and zero images reported as *StatFramesCompleted*, you will likely need to decrease this parameter.

## StreamBytesPerSecond – Uint32 – R/W

In Bytes/Sec. Used to moderate the data rate of the camera. This is particularly useful for slowing the camera down so that it can operate over slower links such as Fast Ethernet (100-speed), or wireless networks. It is also an important control for multi-camera situations. When multiple cameras are connected to a single Gigabit Ethernet port (usually through a switch), StreamBytesPerSecond for each camera needs to be set to a value so that the sum of each camera's StreamBytesPerSecond parameter does not exceed the data rate of the GigE port. Setting the parameter in this way will ensure that multiple camera situations work without packet collisions, i.e. data loss.

115,000,000 is the typical data maximum data rate for a GigE port.

To calculate the required minimum StreamBytesPerSecond setting for a camera in any image mode, use the following formula:

**Height x Width x FrameRate x Bytes per Pixel** (see ImageFormat)

NOTE: If you are seeing occasional "black images", or occasional frames/packets reported as StatFramesDropped/StatPacketsMissed you will likely need to decrease this parameter.

## StreamHold

For controlling when the camera sends data to the host computer. Normally the camera sends data to the host computer immediately after completion of exposure. Enabling StreamHold delays the transmission of data, storing it in on-camera memory, until StreamHold is disabled.

This feature can be useful to prevent GigE network flooding in situations where a large number of cameras connected to a single host computer are capturing a single event. Using the StreamHold function, each camera will hold the event image data until the host computer disables StreamHold for each camera in turn.

**StreamHoldCapacity – Uint32 – R/V**

The total number of image frames that can be stored in the camera memory. Dependent on the camera internal memory size and TotalBytesPerFrame.

**StreamHoldEnable – Enum – R/W**

Enables StreamHold functionality. When disabled, the image data will be released to the host computer. Possible values: *On*, *Off*.

## Timestamp

**TimeStampFrequency – Uint32 – R/C**

In Hz. All images returned from the camera are marked with a timestamp. TimeStampFrequency is the time base for the Timestamp function. The image timestamp can be useful for determining whether images are missing from a sequence due to missing trigger events.

**TimeStampReset – Command**

Reset the camera's time stamp to 0.

**TimeStampValueHi – Uint32 – R/V**

Time stamp, upper 32-bit.

## TimeStampValueLatch – Command

Command. Latch the value of the timestamp on the camera. Both *TimeStampValueHi* and *TimeStampValueLo* are updated with the value read from the camera.

## TimeStampValueLo – Uint32 – R/V

Time stamp, lower 32-bit.

# ImageFormat

## MirrorX – Enum – R/W

Enable horizontal mirroring of the image. Possible Values: *On*, *Off*.

## ROI

Region of Interest. Defines a rectangular sub-region of the image. Selecting an ROI that is small can increase the maximum frame rate and reduce the amount of image data. The following parameters define the size and location of the ROI sub-region:

## Height – Uint32 – R/W

In rows. The vertical size of the rectangle that defines the ROI.

## RegionX – Uint32 – R/W

In pixels. The X position of the top-left corner of the ROI.

## RegionY – Uint32 – R/W

In pixels. The Y position of the top-left corner of the ROI.

## Width – Uint32 – R/W

In columns. The horizontal size of the rectangle that defines the ROI.

## PixelFormat – Enum – R/W

The various pixel data formats the camera can output. Not all cameras have every mode:

<i>Mono8</i>	8 bits per pixel, monochrome. On camera interpolation, with luminance (Y) channel returned. For 10 bit (CMOS) or 12 bit (CCD) sensors, the most significant 8 bits are returned.
--------------	--

<i>Mono16</i>	16 bits per pixel, monochrome. On camera interpolation, with luminance (Y) channel returned. For 10 bit (CMOS) or 12 bit (CCD) sensors, the data is least significant bit aligned within the 16 bit word. That is: 0000xxxx xxxxxxxx.
<i>Bayer8</i>	8 bits per pixel, raw un-interpolated data from camera. SampleViewer interpolates the data in software. For 10 bit (CMOS) or 12 bit (CCD) sensors, the most significant 8 bits are returned.
<i>Bayer16</i>	16 bits per pixel, raw un-interpolated data from camera. SampleViewer interpolates the data in software. For 10 bit (CMOS) or 12 bit (CCD) sensors, the data is least significant bit aligned within the 16 bit word. That is: 0000xxxx xxxxxxxx.
<i>RGB24</i>	24 bits per pixel, on-camera interpolated color.
<i>YUV411</i>	12 bits per pixel, on-camera interpolated color.
<i>YUV422</i>	16 bits per pixel, on-camera interpolated color.
<i>YUV444</i>	24 bits per pixel, on-camera interpolated color.
<i>BGR24</i>	24 bits per pixel, on-camera interpolated color.
<i>RGBA24</i>	24 bits per pixel, support for post overlay, on-camera interpolated color.
<i>Mono12Packed</i>	12 bits per pixel, monochrome.
<i>Bayer12Packed</i>	12 bits per pixel, raw un-interpolated data from camera.

## TotalBytesPerFrame – Uint32 – R/V

Read only. The total number of bytes per image frame. Dependant on ROI, PixelFormat, and Binning.

## ImageMode

Binning is the summing of charge of adjacent pixels on a sensor, to give a lower resolution but more sensitive image.

## BinningX – Uint32 – R/W

The horizontal binning factor.

## BinningY – Uint32 – R/W

The vertical binning factor. In most cases BinningX and BinningY would be set to equal values.

## Info

## CameraName – String – R/W

Human readable camera name. E.g. "EngineRoomCam1".

## DeviceFirmwareVersion – String – R/C

Version of the Firmware the camera is running.



### DeviceModelName – String – R/W

Human readable model name, such as "GE650". Software should use the *PartNumber* and *PartVersion* to distinguish between models.

### DevicePartNumber – String – R/C

Manufacturer's part number

### DeviceSerialNumber – String – R/C

The Serial Number is not a unique identifier across models; software should use *UniqueId* instead.

### DeviceVendorName – String – R/C

Manufacturer's name.

## Firmware

Read only. What firmware is currently loaded on the camera.

### FirmwareVerBuild – Uint32 – R/C

Build number.

### FirmwareVerMajor – Uint32 – R/C

The major part of the Firmware version number (part before the decimal).

### FirmwareVerMinor – Uint32 – R/C

The minor part of Firmware version number (part after the decimal).

## Part

### PartClass – Uint32 – R/C

Camera part class (manufacturer dependant).

### PartNumber – Uint32 – R/C

Camera part number. Manufacturer part number for the camera model.

### PartRevision – String – R/C

Camera revision. Part number revision level.

PartVersion – String – R/C

Camera version. Part number version level.

SerialNumber – String – R/C

Camera serial number.

## Sensor

SensorBits – Uint32 – R/C

The sensor digitization bit depth.

SensorHeight – Uint32 – R/C

The total number of pixel rows on the sensor.

SensorType – Enum – R/C

Monochrome or Bayer-pattern color sensor type.

SensorWidth – Uint32 – R/C

The total number of pixel columns on the sensor.

UniqueID – Uint32 – R/C

The unique camera ID that differentiates the current camera from all other cameras.

## IO

The control and readout of all camera inputs and outputs. The number of inputs and outputs will depend on your camera model.

### Strobe

Valid when any of the SyncOut modes are set to Strobe1. Strobe allows the added functionality of duration and delay, useful when trying to sync a camera exposure to an external strobe.

#### 1

Strobe1ControlledDuration – Enum – R/W

When enabled, the Strobe1Duration control is valid. Possible Values: *On*, *Off*.

## Strobe1Delay – Uint32 – R/W

In microseconds. Delay of start of strobe signal.

## Strobe1Duration – Uint32 – R/W

In microseconds. Duration of strobe signal.

## Strobe1Mode – Enum – R/W

Associates the start of strobe signal with one of the following image capture signals:

<i>AcquisitionTriggerReady</i>	Active once the camera has been recognized by the host PC and is ready to start acquisition.
<i>FrameTriggerReady</i>	Active when the camera is in a state that will accept the next frame trigger.
<i>FrameTrigger</i>	Active when an image has been initiated to start. This is a logic trigger internal to the camera, which is initiated by an external trigger or software trigger event.
<i>Exposing</i>	Active for the duration of sensor exposure.
<i>FrameReadout</i>	Active at during frame readout, i.e. the transferring of image data from the CCD to camera memory.
<i>Imaging</i>	Active during exposure and readout.
<i>Acquiring</i>	Active during an acquisition stream.
<i>SyncIn1</i>	Active when there is an external trigger at SyncIn1
<i>SyncIn2</i>	Active when there is an external trigger at SyncIn2

NOTE: Please refer to camera waveform diagrams provided in the camera manuals for more detail information

## SyncIn

### SyncInLevels – Uint32 – R/V

A bitmask, each bit corresponding to a specific SyncIn input. For example: 2 equals (0010) which means SyncIn2 is high and all other Sync input signals are low.

## SyncOut

Controls the camera outputs. Can be used for synchronization with other cameras/devices or general purpose outputs.

## 1

### SyncOut1Invert – Enum – R/W

When enabled, reverses the polarity of the signal output by SyncOut1.  
Possible values: *On, Off*.

## SyncOut1Mode – Enum – R/W

Determines the type of output defined by SyncOut1:

<i>GPO</i>	configured to be a general purpose output, control of which is assigned to SyncOutGpoLevels
<i>AcquisitionTriggerReady</i>	Active once the camera has been recognized by the host PC and is ready to start acquisition.
<i>FrameTriggerReady</i>	Active when the camera is in a state that will accept the next frame trigger.
<i>FrameTrigger</i>	Active when an image has been initiated to start. This is a logic trigger internal to the camera, which is initiated by an external trigger or software trigger event.
<i>Exposing</i>	Active for the duration of sensor exposure.
<i>FrameReadout</i>	Active at during frame readout, i.e. the transferring of image data from the CCD to camera memory.
<i>Acquiring</i>	Active during a acquisition stream.
<i>SyncIn1</i>	Active when there is an external trigger at SyncIn1
<i>SyncIn2</i>	Active when there is an external trigger at SyncIn2
<i>Strobe1</i>	The output signal is controlled according to Strobe1 settings.

Note: Refer to camera waveform diagrams provided in the camera manual for more detailed information.

## 2 / 3 / 4

Same as SyncOut1.

## SyncOutGpoLevels – Uint32 – R/W

GPO output levels. A bitfield. Bit 0 is sync-out 0, bit 1 is sync-out 1, etc.

## **Driver Attributes**

### **GigE**

#### **Ethernet**

DeviceEthAddress – String – R/C

The physical MAC address of the camera.

HostEthAddress – String – R/C

The physical MAC address of the host network card on which the camera is reached.

#### **IP**

DeviceIPAddress – String – R/C

The current IP address of the camera.

HostIPAddress – String – R/C

The current IP address of the host network interface.

GvcpRetries – Uint32 – R/W

Gvcp = GigE Vision Control Protocol. The maximum number of resend requests that the host will attempt when trying to recover a lost control packet.

#### **Gvsp**

Gvsp = GigE Vision Streaming Protocol.

GvspLookbackWindow – Uint32 – R/W

Size of the look back window, in packets, when determining if a stream packet is missing.

GvspResentPercent – Float32 – R/W

Maximum of missing stream packets that will be requested from the camera if they are detected missing.

GvspRetries – Uint32 – R/W

The maximum number of resend requests that the host will attempt when trying to recover a lost stream packet.

### GvspSocketBufferCount – Enum – R/W

Number of buffers to be used by the network socket. Only applicable when not using the Filter Driver.

Possible values: 256,512,1024,2048,4096,8192

### GvspTimeout – Uint32 – R/W

End of stream timeout, in milliseconds.

### HeartbeatInterval – Uint32 – R/W

In milliseconds. The interval at which the API sends a heartbeat command to the camera. Normally this parameter does not require adjustment.

### HeartbeatTimeout – Uint32 – R/W

In milliseconds. The maximum amount of time the camera will wait for a heartbeat command before timing out.

## Multicast

Multicast mode allows the camera to send image data to all hosts on the same subnet as the camera. The host computer that first enables multicast mode is the *master*, and controls all camera parameters. All other hosts / instances are the *monitors*, and can view image data only.

NOTE: Most GigE switches support a maximum PacketSize of 1500 in Multicast mode.

### MulticastEnable – Enum – R/W

Enables Multicast mode. Possible values: *On*, *Off*.

Note: In order to enable this, the camera must not be streaming.

### MulticastIPAddress – String – R/W

Set the multicast IP address.

## Stats

### StatDriverType – Enum – R/W

*Standard*  
*Filter*

The default network card driver is being used only.  
The AVT filter driver is being used in conjunction with the default network card driver. Using the Filter driver will reduce the load on the host CPU.

### StatFilterVersion – String – R/C

Version of the filter driver.

### StatFrameRate – Float32 – R/V

Frame rate of the camera.

### StatFramesCompleted – Uint32 – R/V

The number of frames captured since the start of imaging.

### StatFramesDropped – Uint32 – R/V

The number of frames dropped during transmission since the start of imaging.

With proper configuration and hardware, this number should be zero. See StreamBytesPerSecond, PacketSize, and refer to the Host Computer Optimizations note.

### StatPacketsErroneous – Uint32 – R/V

The number of improperly formed packets. If this number is non-zero, it suggests a possible camera hardware failure.

### StatPacketsMissed – Uint32 – R/V

The number of packets missed since the start of imaging.

### StatPacketsReceived – Uint32 – R/V

The number of packets received since the start of imaging.

### StatPacketsRequested – Uint32 – R/V

The number of resend requests since the start of imaging. When an expected packet is not received by the driver, it is recognized as missing and the driver requests the camera to resend it.

### StatPacketResent – Uint32 – R/V

The number of packets resent by the camera and received by the host, since the start of imaging.

## Contacting Allied Vision Technologies

- **Technical information:**

<http://www.alliedvisiontec.com>

- **Support:**

[support@alliedvisiontec.com](mailto:support@alliedvisiontec.com)

**Allied Vision Technologies GmbH (Headquarters)**

Taschenweg 2a  
07646 Stadtroda, Germany  
Tel.: +49.36428.677-0  
Fax.: +49.36428.677-28  
e-mail: [info@alliedvisiontec.com](mailto:info@alliedvisiontec.com)

**Allied Vision Technologies Canada Inc.**

101-3750 North Fraser Way  
Burnaby, BC, V5J 5E9, Canada  
Tel: +1 604-875-8855  
Fax: +1 604-875-8856  
e-mail: [info@alliedvisiontec.com](mailto:info@alliedvisiontec.com)

**Allied Vision Technologies Inc.**

38 Washington Street  
Newburyport, MA 01950, USA  
Toll Free number +1-877-USA-1394  
Tel.: +1 978-225-2030  
Fax: +1 978-225-2029  
e-mail: [info@alliedvisiontec.com](mailto:info@alliedvisiontec.com)