

该内容不适用于您选择的语言

该内容将以另一种语言显示出来。 您可使用浏览器的翻译功能查看内容。

How to Acquire From GigE Vision Cameras With Vision Acquisition Software

Updated Aug 28, 2020

Environment

Software

- LabVIEW

Driver

- Vision Acquisition Software

This tutorial continues the discussion in Part I regarding using GigE Vision cameras with Vision Acquisition Software. In this part, we will discuss the hardware and software requirements for acquisition. Then we review using the LabVIEW API to first acquire images from the camera, control the camera settings, and then setup triggering of the camera.

To understand the basics of the GigE Vision standard, please read the Application Note:

[Acquiring from GigE Vision Cameras with Vision Acquisition Software](#)

Hardware and Software Setup

In order to acquire images from a GigE Vision camera, you need to first make sure that you have all the correct hardware and software. Below is a list of requirements.

Hardware

- GigE Vision camera: The camera must be GigE Vision standard compliant. If you have a camera that has a Gigabit Ethernet port but is not GigE Vision compliant, you cannot acquire images using Vision Acquisition Software. You should find the GigE Vision logo in the camera's user manual or marketing literature.
- Gigabit Ethernet port: While it is possible to acquire images with Ethernet and Fast Ethernet ports, which support 10 MB/s and 100 MB/s respectively, this will only work at very slow frame rates and small resolutions. It is highly recommended that you use a Gigabit Ethernet Network Interface Controller (NIC).

Software

- Vision Acquisition Software: You will need Vision Acquisition Software 8.2.1 or higher, which will install NI-IMAQdx 3.0 or higher. NI-IMAQdx is the driver that includes the functionality to acquire images from USB3 Vision, GigE Vision, and IEEE 1394 cameras.
- Application Development Interface (ADE): You can use LabVIEW, LabWindows/CVI, Visual Studio 6.0, or any ANSI C compiler to acquire images.

Network Configuration

Once you have the hardware and software installed correctly, you must configure the network as well. As discussed in Part I, GigE Vision cameras can obtain an IP address from a DHCP server or select one for itself using Link Local Addressing (LLA). If you connect the camera to a Gigabit Ethernet network with a DHCP server, the camera is automatically detected. If the camera is connected directly to the computer (using either a regular or cross-over cable), you will need to wait about a minute for the camera to timeout on the DHCP request and use LLA. The Windows operating system may display a warning that the network card has only limited operation. You can ignore this warning. **Note**: that the delay only applies to Windows XP and 2000, not the Windows Vista Operating System.

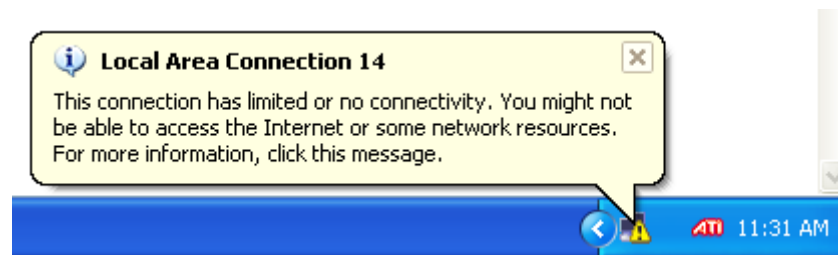


Figure 1. Windows displays a warning when camera is directly connected

Jumbo Packets

Typically, network drivers will split any data larger than 1500 bytes into multiple packets. However, the GigE Vision standard allows packet sizes of up to 9014 bytes. These large packets, also known as Jumbo packets, allow the camera to more efficiently transfer data across the network. You can enable Jumbo packets in many network cards from the Windows Device Manager by right-clicking the network card and selecting **Properties**.

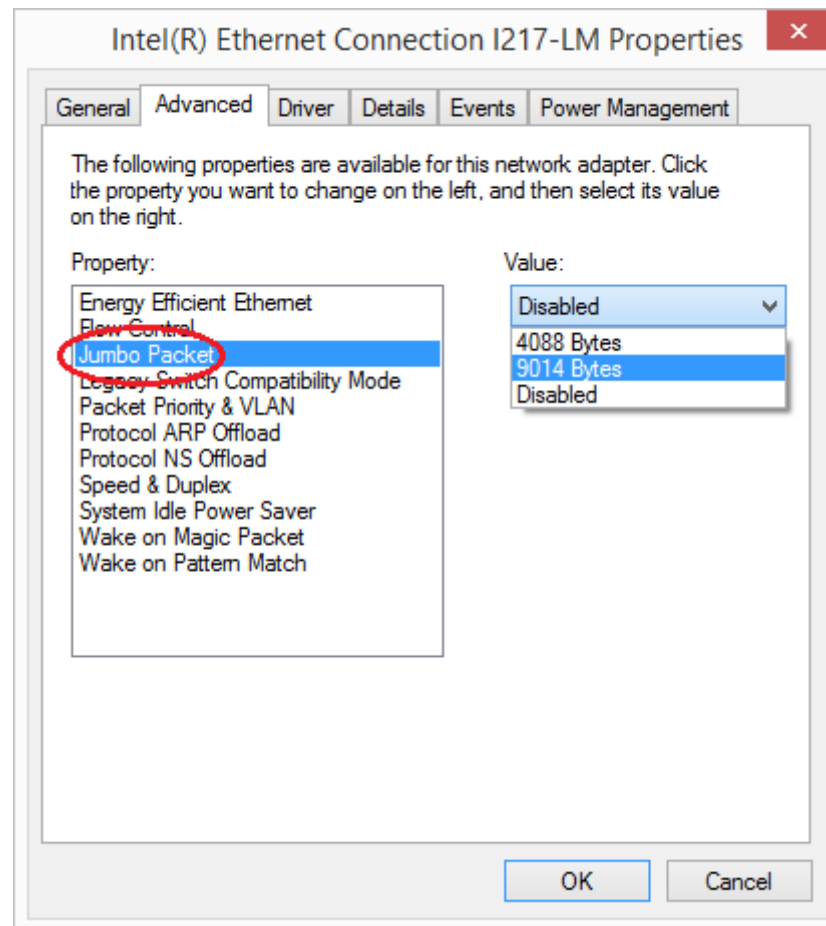


Figure 2. Example of setting Jumbo packets on the Intel PRO/1000 Adapter

Network Firewalls

When a camera acquires an image, it immediately streams those data packets to the host. However, network firewalls will not allow the packets to reach their destination because firewalls typically block uninitiated incoming traffic. Therefore you will need to disable your firewall in order to acquire images from a GigE Vision camera. You can disable the Windows Firewall from the Control Panel (Start»Control Panel). However, if you have a network card with an Intel PRO/1000 chipset and you are using the High Performance driver, you will not need to disable the firewall. Since the High Performance driver redirects incoming GigE Vision packets to the NI-IMAQdx kernel driver before it reaches the firewall, your firewall settings will not affect image acquisition.

Acquiring Images in MAX

Measurement and Automation Explorer (MAX) is used to verify that you have discovered the camera and can acquire images. Since the NI-IMAQdx driver supports Plug and Play (PnP), any GigE Vision cameras on the same subnet as the host should automatically appear in the Devices and Interfaces sub tree. GigE Vision cameras are enumerated under the NI-IMAQdx sub-tree and are identified by a special icon. If you are using NI IMAQdx 4.3.5 or later, GigE Vision cameras will appear in the Network Devices sub tree.

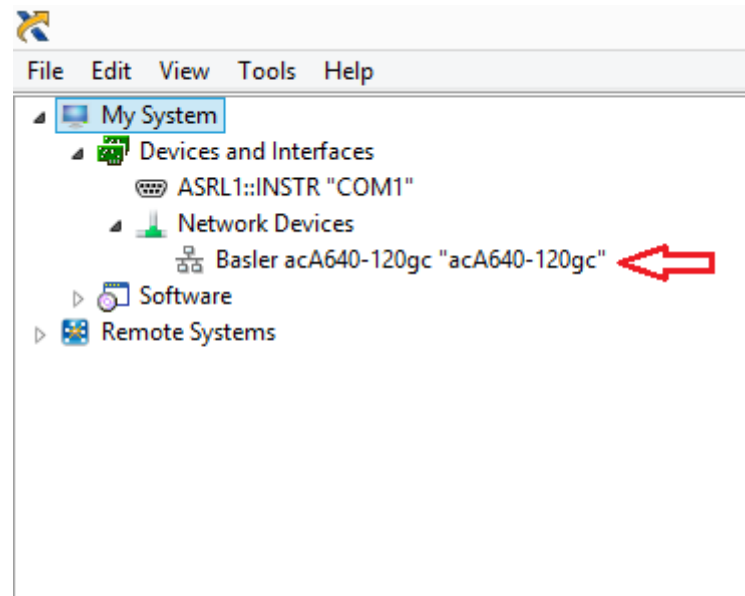


Figure 3. MAX automatically detects any GigE Vision camera on the same subnet

MAX will display any GigE Vision camera on the same subnet as the host. However, NI-IMAQdx allows you to acquire images from cameras on remote subnets too. You can discover cameras on remote subnets by calling the appropriate function in the NI-IMAQdx API. For example, the C function `IMAQdxDiscoverEthernetCameras()` has a parameter to specify the subnet to discover cameras on.

Once you are able to discover the camera in MAX, the next step is to acquire images from it. Select the camera from the sub tree to open it in the main window. Below are the various parts of the Acquisition tab and their descriptions

Figure 4. The Acquisition Attributes page

- **Video Mode:** This attribute is only valid for IEEE 1394 cameras. It is grayed out for GigE Vision cameras.
- **Pixel Format:** Displays a list of available pixel format. Typical formats are Mono8, Mono16 and YUV422Packed.
- **Region of Interest:** Specifies the left and top offsets and the width and height of the acquisition window.
- **Timeout:** Sets the number of milliseconds the driver will wait for an acquisition to complete before returning a timeout error.
- **Packet Size:** Specifies the number of bytes transferred in each data packet. This number must be less than the maximum packet size allowed by the network card (1500 if Jumbo packets are disabled and 9014 if Jumbo packets are enabled).

Once you have set the acquisition parameters correctly, click the **Snap** to acquire one image or click the **Grab** to acquire images continuously.

Acquiring Images in LabVIEW

NI-IMAQdx provides a unified API for acquiring images from both IEEE 1394, USB3 Vision, and GigE Vision cameras. While some functionality is specific to one type of bus, most functions and VIs can be used with both types of cameras. This enables a more bus agnostic development for image acquisition. You can replace your IEEE 1394 camera with a GigE Vision camera, or vice versa, with little or no change to your code.

The NI-IMAQdx LabVIEW API is divided into high-level and low-level VIs. Using the high-level VIs, you can program a simple snap, grab or sequence operation. The low-level VIs allow you to perform the same tasks as the high-level VIs but give you greater control over execution details. Look at the examples that ship with LabVIEW to understand how to program image acquisition using NI-IMAQdx.

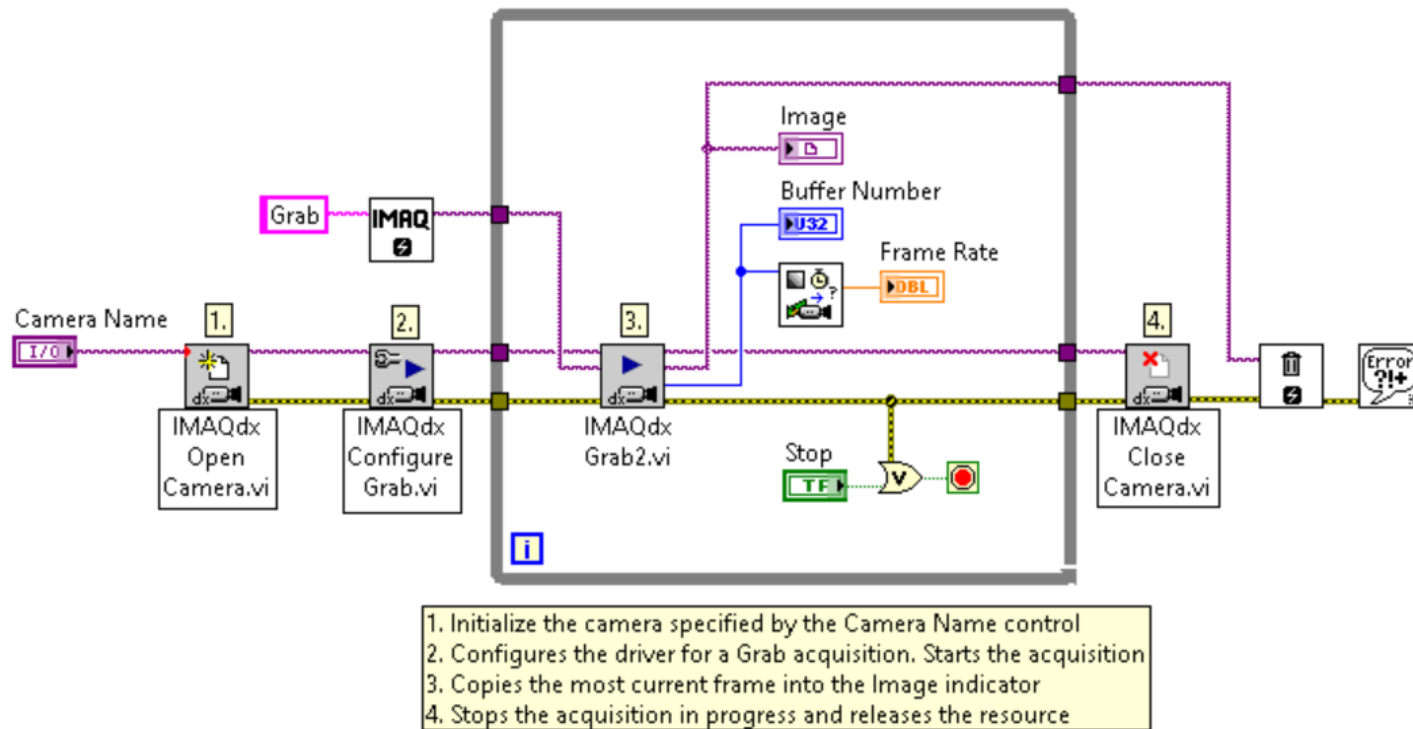


Figure 5. A simple grab example

The above example illustrates a simple Grab acquisition in LabVIEW. The acquired images are displayed in the Image indicator. This example shows the buffer number in the Buffer Number Indicator. You may lose buffers if the loop rate of the while loop is not higher than the frame rate of the camera. In such cases, an image copied into a buffer in memory is overwritten with another image before the original image can be processed. In most machine vision cases, it is important to notify the user if any frames have been missed.

Understanding Camera Attributes

Cameras typically support several settable attributes that enable the camera to be flexible enough to work in different environments with varied constraints. While most machine vision cameras support some typical attributes, such as gain, shutter speed, or bit depth, many cameras have a unique attribute subset that is specific only to that camera or family of cameras.

| | | | | |
|------------------------------|--|--------------------------|--------------|-----------|
| Expand All | | Refresh | View Options | Hide Help |
| Camera Attributes | | | | |
| + Analog Controls | | | | |
| + Image Format Controls | | | | |
| - AOI Controls | | | | |
| Width | | 658 | | |
| Height | | 492 | | |
| X Offset | | 0 | | |
| Y Offset | | 0 | | |
| Center X | | <input type="checkbox"/> | | |
| Center Y | | <input type="checkbox"/> | | |
| + Color Improvements Control | | | | |
| - Acquisition Controls | | | | |
| Acquisition Mode | | Continuous | | |
| AcquisitionFrameCount | | 1 | | |
| Trigger Selector | | Frame Start | | |
| Trigger Mode | | Off | | |
| Generate Software Trigger | | | | |

Figure 6. The Camera Attributes tab in MAX displays all attributes

The GigE Vision standard defines a minimal set of attributes that are required to capture an image. These attributes, such as image width, height, pixel format, etc., must be supported by every GigE Vision camera. However, additional attributes supported by a camera can be exposed using the GenICam standard.

The GenICam Standard

The GigE Vision specification relies on GenICam, which is a standard of the European Machine Vision Association (EMVA) to describe the features (attributes) supported by a camera. Every GigE Vision Camera must provide an XML device description file conforming to the GenICam syntax. When a camera is connected and first selected in MAX, this XML file is retrieved and interpreted by NI-IMAQdx to enumerate the attributes supported by the camera. Since each camera vendor provides an XML file specific to each camera, NI-IMAQdx can auto-populate the attributes specific to that camera.

```

<Integer Name="Gain" Namespace="Standard">
  <ToolTip>Analog Gain setting for camera</ToolTip>
  <DisplayName>Gain</DisplayName>
  <pValue>GainReg</pValue>
  <Min>0</Min>
  <Max>128</Max>
  <Inc>1</Inc>
  <Representation>Logarithmic</Representation>
</Integer>

<IntReg Name="GainReg">
  <ToolTip>Access node for the camera's Gain feature</ToolTip>
  <Address>0x0815</Address>
  <Length>2</Length>
  <AccessMode>RW</AccessMode>
  <pPort>Device</pPort>
  <Sign Unsigned </Sign>
  <Endianess>BigEndian</Endianess>
</IntReg>

```

Figure 7. Snippet of an XML file describing the gain attribute

Figure 7 shows a very simple example of the gain attribute being described in an XML file. Upon parsing this snippet of XML code, NI-IMAQdx determines the following:

- The feature name is Gain.
- It has a minimum and maximum value of 0 and 128 respectively and has a minimum increment of 1.
- The gain value is represented in logarithmic form.
- The current gain value is stored in a register with address 0x0815 in BigEndian Unsigned form and is 2 bytes long.
- The driver has both read and write access to this register.

Every such attribute supported by the camera will have a section of code in the XML file that defines the attribute parameters. You can examine the XML file manually by opening it from the <Program Files>\National Instruments\NI-IMAQdx\Data\XML directory. **Note:** The above example is a very simplistic representation of an

attribute and is provided as an academic exercise. Typical XML files are far more complicated and involve many cross references.

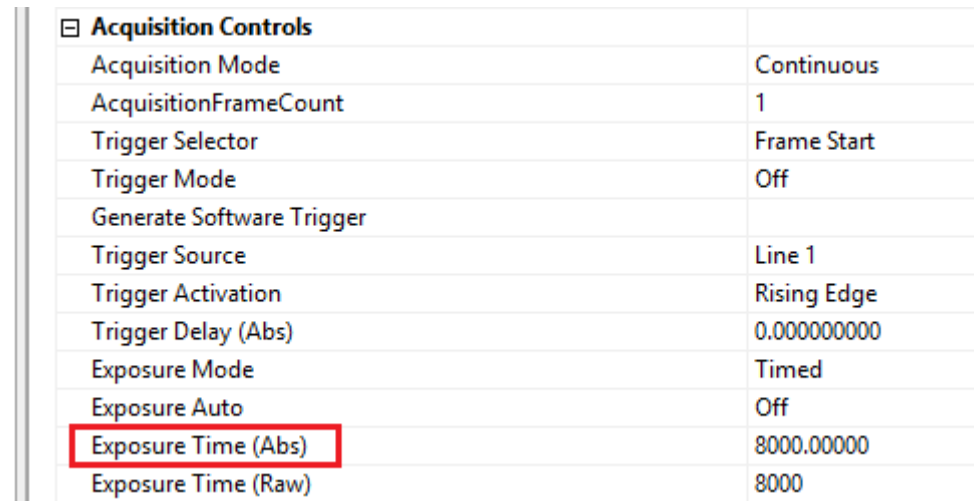
Programmatically Controlling Camera Settings

Camera settings can be controlled by using MAX (see Figure 6) to set the values of attributes exposed in the XML file. However, many applications need the ability to programmatically change camera attributes. The NI-IMAQdx API provides methods to change the value of any attribute exposed in the XML file. While camera attributes can be set in any supported API, we shall discuss its implementation in LabVIEW.

Every attribute supported by a camera is defined by these (non-exhaustive) properties:

- Name: Unique name of the attribute
- Representation: Can be an integer, float, boolean, enumeration, string or command
- Unit of Measurement: Unit which the value represents (e.g. microseconds(μ s))
- Access: Read-only, Write-only, or Read/Write

The camera manufacturer can provide you with documentation detailing the properties for each attribute. If the documentation is not available, you can use MAX to determine the properties of a certain attribute. To do so, simply select the desired attribute from the Camera Attributes tab. For example, we shall examine the ExposureTimeAbs attribute of a Basler Scout scA640-70gm camera. From Figure 8, we can determine that the attribute ExposureTimeAbs is a floating point number with microseconds as its unit.



| | |
|---------------------------|-------------|
| [-] Acquisition Controls | |
| Acquisition Mode | Continuous |
| AcquisitionFrameCount | 1 |
| Trigger Selector | Frame Start |
| Trigger Mode | Off |
| Generate Software Trigger | |
| Trigger Source | Line 1 |
| Trigger Activation | Rising Edge |
| Trigger Delay (Abs) | 0.000000000 |
| Exposure Mode | Timed |
| Exposure Auto | Off |
| Exposure Time (Abs) | 8000.00000 |
| Exposure Time (Raw) | 8000 |

Figure 8. ExposureTimeAbs Attribute in MAX

In LabVIEW you can set the attribute values using a Property Node. However, at development time, LabVIEW cannot know the attribute's name or representation. Therefore, you will need to provide the attribute name and call the appropriate function depending on the attribute representation (integer, string, boolean, etc.).

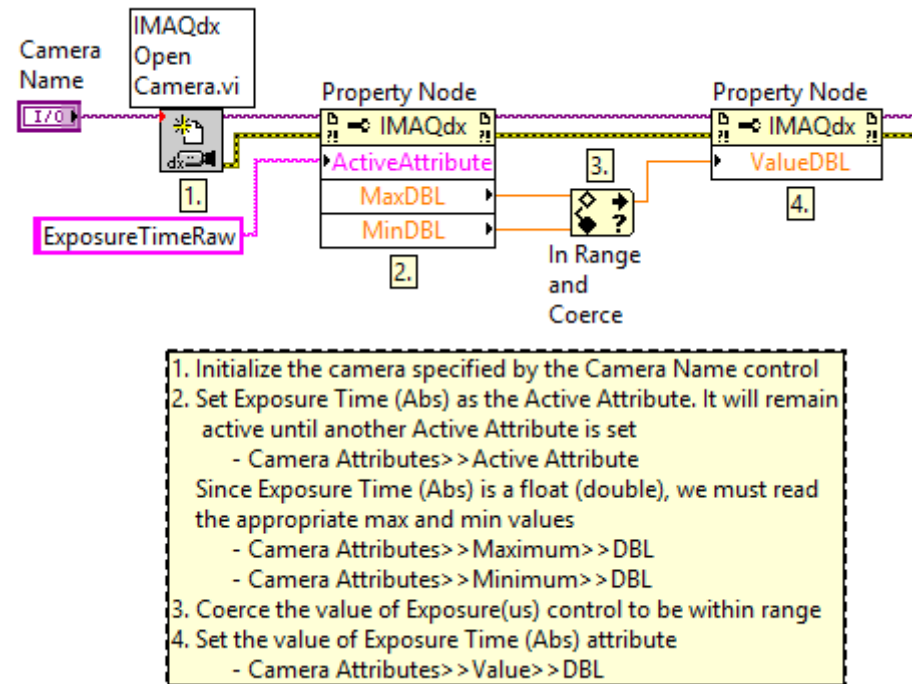


Figure 9. Setting the ExposureTimeAbs attribute of a Basler scA640-70gm

While GenICam provides a flexible method to control cameras, the standard is not enough to guarantee interoperability. Interoperability is the ability to switch between different cameras and still maintain the functionality of the application software. For example, if we replace the Basler camera in the above example with a camera from another manufacturer. This camera might have the attribute ExposureTimeAbs represented in nanoseconds or as an integer or may even refer to the attribute by a different name. Clearly, each camera will produce different acquisition results for the same inputs.

In order to improve interoperability, EMVA, in partnership with camera manufacturers, created the GenICam Standard Features Naming Convention. The goal of this document is to standardize the name, representation, access, unit, and function of many of the attributes common to most cameras. By using this naming convention in unison with the GenICam standard, camera manufacturers can promote interoperability with other cameras for standard features while still giving users access to unique features in their cameras.

Triggering in GigE Vision

In most Machine Vision applications, the camera needs to take images based on real-world events. For example, a bottle inspection system must capture each image when a bottle is in the exact same position on the conveyer belt with respect to the camera. This will make the bottle appear in exactly the same location in the image and hence will simplify the image processing. You can achieve such control using hardware triggers.

In typical hardware triggered systems, a proximity sensor or an encoder sends pulses to trigger an acquisition. In many cases, the trigger is connected to a framegrabber which initiates an acquisition. However, due to the potentially long distances possible with GigE Vision cameras (up to 100 Meters), triggering the framegrabber is not feasible. Therefore all trigger signals must be directly connected to the camera.

In GenICam, selecting trigger modes works just like setting camera attributes. In fact, trigger modes are attributes in GenICam. You can use the same API discussed in the previous section to set trigger modes. The GenICam Standard Features Naming Convention defines several trigger control features that let you customize the behavior of your triggered action. While camera manufacturers are not required to implement all triggering modes, the most commonly used modes are described below.

- AcquisitionStart: Trigger starts the acquisition on the camera
- AcquisitionActive: Camera acquires images as long as the trigger is active
- FrameStart: Camera captures one image for each trigger
- LineStart: Camera captures one line for each trigger (used in line scan cameras)
- ExposureStart: Trigger begins exposure of one frame (or line)
- ExposureActive: Frame (or line) is exposed as long as the trigger is active

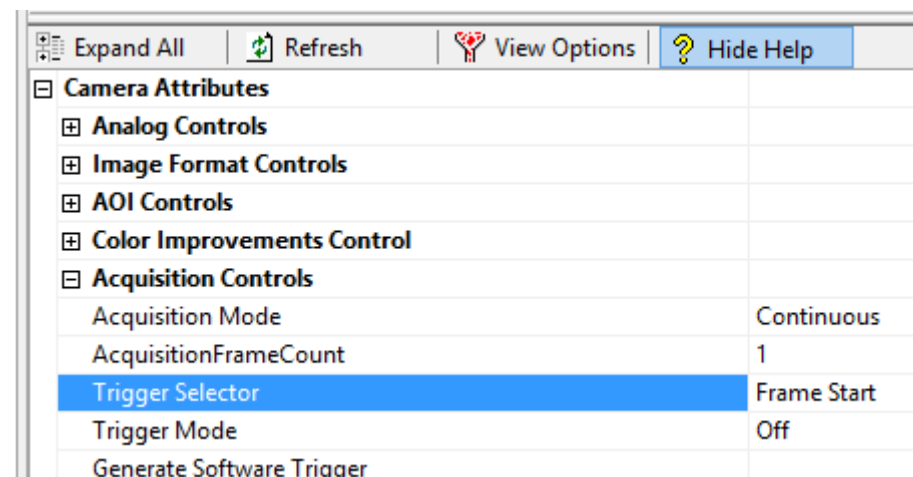


Figure 10. Trigger attributes for a Prosilica GE650 camera

As an example, let us look at the triggering modes of a Prosilica GE650 GigE Vision camera. The Camera Attributes tab in MAX can be seen in Figure 10. Notice that we have selected that the camera should capture one image for each rising edge of a trigger connected to Line1. These settings, once saved in MAX, are automatically loaded when the camera is opened in any API. However you may want to programmatically set the settings. Figure 11 uses LabVIEW to programmatically configure the camera to the same settings as Figure 10.

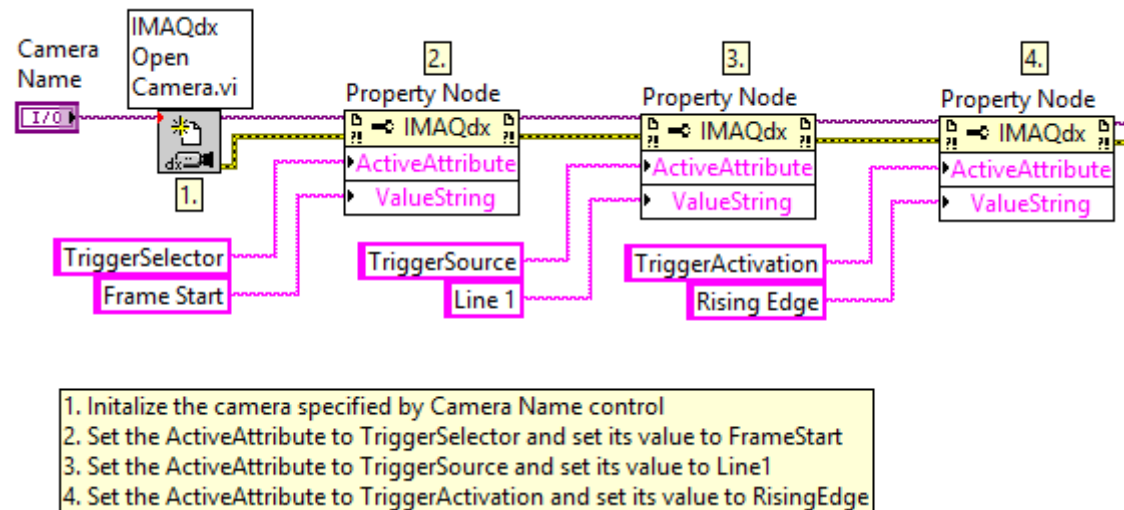


Figure 11. A simple triggering example in LabVIEW

Caveats and Pitfalls

Jumbo Packets: If your NIC device, or any intermediate network hardware (switch, router, etc.), does not support Jumbo packets, you will be limited to a packet size of less than 1500 Bytes. The GigE Vision packet size cannot be greater than the maximum packet size allowed by the NIC.

Firewalls: Many corporate networks employ firewalls for network security. However, you cannot acquire from GigE Vision cameras with the firewall enabled, unless you use the High Performance driver. If your company's network policy does not allow you to disable the firewall or use a different network driver, you will need to use a system dedicated to image acquisition, that is not part of the corporate network.

Corrupt XML files: As with any new standard, camera manufacturing companies routinely release new revision of their firmware. If you get an error stating that the XML file is corrupt, please contact the camera manufacturer for the latest revision of their firmware.

Interoperability: While GenICam gives camera manufacturers the flexibility of creating a custom attribute set, it makes it difficult to easily switch between cameras without modifying your code. While the GenICam Standard Features Naming Convention alleviates this problem to a certain extent, most of the conventions are only recommendations and not requirements. So a camera manufacturer may deviate from the convention, in which case, the application software will need to be modified to be interoperable with other cameras.

Related Links

- [Acquiring from GigE Vision cameras using Vision Acquisition Software](#)
- [External Link: European Machine Vision Association: The GenICam Standard](#)
- [List of GigE Vision Cameras on NI Camera Advisor](#)
- [Troubleshooting GigE Vision Cameras with Vision Acquisition Software](#)

Other Support Options

Ask the NI Community

Collaborate with other users in our discussion forums

- [Search the NI Community for a solution](#)

Request Support from an Engineer

A valid service agreement may be required, and support options vary by country.

- [Open a service request](#)
- [Purchase or renew support services](#)

本信息是否对您有帮助?

有帮助



没有帮助

