IDS peak 2.17.1 for Windows - Readme

Announcement

Support end for native 32-bit operating systems

For performance, security, and future-proofing reasons, it is generally recommended to use a 64-bit operating system. **IDS peak 2.17 is the last version that supports native 32-bit operating systems.**

What does this mean for you?
Existing installations on 32-bit operating systems will continue to work, but will no longer receive updates or patches. Future versions of IDS peak will require a 64-bit operating system.

For Windows 64-bit setup, there is the option to run 32-bit applications via WoW64 (Windows 32-bit on Windows 64-bit).

System requirements

For installing IDS peak, the following system requirements must be met:

- Disk space:
  - Runtime setup: approx. 26 MB
  - Standard setup: approx. 530 MB
  - Extended setup: approx. 645 MB
- Operating system: Windows 11 (64-bit) / Windows 10 (64/32-bit)
  IDS peak has been tested with the following versions:
  - Edition: Windows 11 Pro
    Version: 24H2
  - Edition: Windows 10 Pro
    Version: 22H2

Note: We recommend to use 64-bit applications. Depending on the use case and camera model, the resources of 32-bit applications may not be sufficient, e.g. for high memory requirements.

Note: In 64-bit compilers, the /LARGEADDRESSAWARE option is enabled by default (this means that an application can handle addresses larger than 2 gigabytes). In 32-bit compilers, the /LARGEADDRESSAWARE option can also be used.

Optional

- CMake (minimum version 3.8) to build the samples yourself (if required)

- C++ 14 compiler (for samples, IDS peak comfortSDK, and IDS peak genericSDK)
- .NET Framework 4.6.1 or .NET 5 and higher
- Python 3.7 and higher (for samples and IDS peak genericSDK binding)
    - For using IDS peak IPL, the NumPy library is additionally required.

IDS peak components

IDS peak applications

- IDS peak Cockpit
  The IDS peak Cockpit is a graphical user interface program for easy camera evaluation. The IDS peak Cockpit allows you to easily manage your cameras with the integrated camera manager, save your workspace settings, save images automatically and much more.
- Tools
    - IDS IP Config - Command line tool for setting camera IP and network settings for GigE Vision cameras
    - IDS Device Update - Command line tool for updating the camera firmware for GigE and USB3 Vision cameras
    - IDS Device Command - Command line tool for manually controlling the camera settings and commands
    - IDS Device Password GUI / IDS Device Password - GUI/command line tool for editing the password of a password-protected camera (certain models only)
    - Support tool - Command line tool for storing information about the system used and the camera configuration. You can send this information to IDS to enable easier error handling.

USB3 Vision Transport Layer (U3VK)

- GenTL Producer for IDS USB3 Vision cameras (necessary for image processing programs based on the GenICam interface) including a driver to improve USB performance.

GigE Vision Transport Layer

- Socket (GEV)
  Installs the GenTL Producer (necessary for image processing programs based on the GenICam interface). This producer can be installed as an alternative to "Kernel" if it is not possible to install a kernel driver in the system.
- Kernel (GEVK) - recommended
  Installs the GenTL Producer (necessary for image processing programs based on the GenICam interface) and a kernel driver to improve Ethernet performance.

uEye Transport Layer

- The uEye Transport Layer allows you to use uEye cameras (UI models) with IDS peak.
    - IDS peak standard and runtime setup: You can install the uEye Transport Layer via the custom installation of IDS peak. Note that you must also install IDS Software Suite 4.94 or higher to operate uEye cameras (UI models) with IDS peak.
    - IDS peak extended setup: This setup includes the uEye camera drivers. Therefore, no additional installation of the IDS Software Suite is required.

IDS peak comfortSDK

- Library
  Installs the IDS peak comfortSDK. Here, you can use numerous simple functions without going deep into the GenICam standard. It is possible to program in C.
- Samples
  Install source code samples for IDS peak comfortSDK.

IDS peak genericSDK

- IDS peak genericAPI
  Installs the IDS peak genericSDK (based on GenICam) for programming.
    - C++ libraries
    - .NET bindings
- IDS peak AFL
  Installs an additional library for auto features.
    - C++ libraries
    - .NET bindings
- IDS peak IPL
  Installs an additional library for basic image processing.
    - C++ libraries
    - .NET bindings
- Sample programs
  Installs source code samples for IDS peak genericSDK.
    - C++
    - C#
    - Python

DirectShow

- Installs the DirectShow interface of IDS peak.

Third Party Samples

- Installs source code samples for HALCON and MIL (Matrox Imaging Library)

Installation

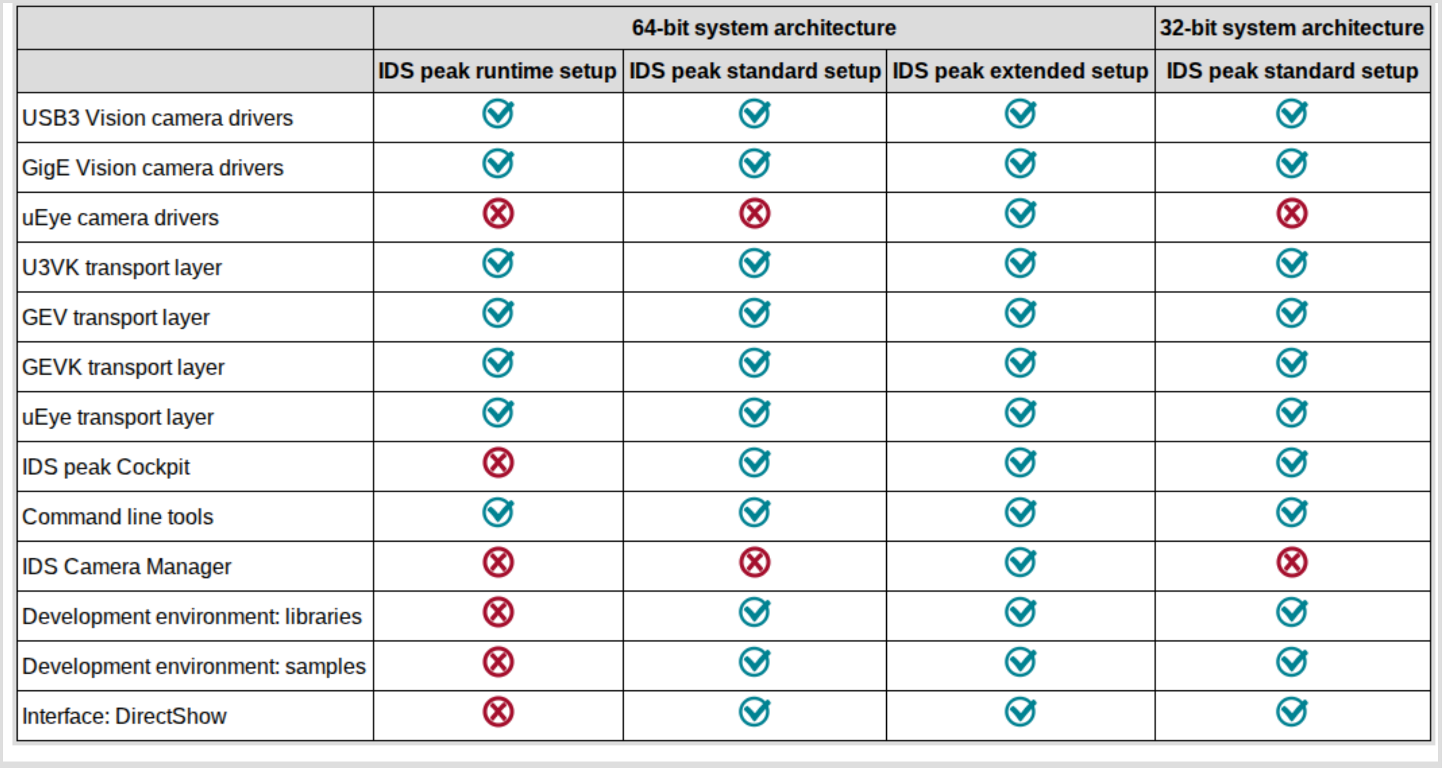
Administrator privileges are required to install the software.

Which setup to choose?

The Windows setup of IDS peak is available in different versions:

- IDS peak standard setup: This setup has the same scope as in the previous version. Using the uEye Transport Layer, you can operate uEye cameras (UI models) if you have installed the IDS Software Suite.
- IDS peak runtime setup: This is a greatly reduced setup, which contains only the drivers without development environment or IDS peak Cockpit. Using the uEye Transport Layer, you can operate uEye cameras (UI models) if you have installed the IDS Software Suite.
- IDS peak extended setup: This setup also contains the uEye camera drivers for UI models. With this setup, you can use UI camera models without installing the IDS Software Suite. The setup also contains the IDS Camera Manager, which you can use to configure the IP addresses of the GigE uEye cameras (UI-5xxx), for example.
  Note that this setup automatically uninstalls an existing version of the IDS Software Suite.

The following overview shows you which components are included in the individual setups.



	64-bit system architecture			32-bit system architecture
	IDS peak runtime setup	IDS peak standard setup	IDS peak extended setup	IDS peak standard setup
USB3 Vision camera drivers	✓	✓	✓	✓
GigE Vision camera drivers	✓	✓	✓	✓
uEye camera drivers	✗	✗	✓	✗
U3VK transport layer	✓	✓	✓	✓
GEV transport layer	✓	✓	✓	✓
GEVK transport layer	✓	✓	✓	✓
uEye transport layer	✓	✓	✓	✓
IDS peak Cockpit	✗	✓	✓	✓
Command line tools	✓	✓	✓	✓
IDS Camera Manager	✗	✗	✓	✗
Development environment: libraries	✗	✓	✓	✓
Development environment: samples	✗	✓	✓	✓
Interface: DirectShow	✗	✓	✓	✓

Installation of IDS peak

Note: If you change the setup, components are automatically uninstalled during a downgrade. Example: You have previously installed the standard setup and would now like to install the runtime setup. The runtime setup will automatically uninstall components that are not included in the setup. In the case of installation via the GUI, you will be notified. In the case of a silent installation, this takes place without any further notification.

Standard installation via setup file

Double-click on `ids_peak_<version>.exe` to run the setup. You have the following options for installation:

1. Typical
   Installation of recommended components.
2. Custom
   For user-defined installation. All components to be installed must be selected individually.
   In order to support uEye cameras (UI models), choose "Custom" and enable the installation of the uEye Transport Layer.

After the installation is finished, the PC must be restarted.

Note: If changes are required to an existing installation, you can use the "Modify" option of the corresponding setup.

Installation via silent setup

Prerequisites

The silent setup runs in the background and does not prompt the user for input. It gets its input from a silent response file (*.iss file) which you need to create in advance:

1. Extract the IDS peak setup package to a local directory. The required silent setup command line parameters can only be processed by the `ids_peak_<version>.exe`.
2. Record a setup response file. Run the setup with the /r command line parameter to have the setup record and create the setup response file, for example:
   ```
   ids_peak_<version>.exe /r
   ```
   → The setup will record all your setup choices in a file named "setup.iss" which will be placed in the Windows folder c:\Windows i.e. %WINDIR%.
3. If necessary, you may rename the setup.iss file and locate it in a different directory. Do not use space characters.

Running the silent installation

You have two options:

1. The "setup.iss" file is located in the setup directory. This is the default option.
   Enter `ids_peak_<version>.exe /s /f1".\setup.iss"` via command line.
2. The "setup.iss" file has a different name and is located in a different directory.
   Enter the appropriate file name and location behind the f1 command, see example. Do not use space characters within the command line option /f1 command.
   Example: `ids_peak_<version>.exe /s /f1"d:\mySetup.iss"`

Note: When running a setup in silent mode no messages are displayed at all. You can review the Setup.log file for getting an installation summary including error log.

Installation of Python

Note: It is necessary to reinstall the Python bindings after updating/reinstalling IDS peak.

To use the Python binding of IDS peak make sure that Python is installed on your PC. Note that Python version **3.7 and higher** are supported.

You can use `pip` to install the IDS peak Python bindings from PyPI.

Installing the Python bindings from PyPI

There are three packages to be installed:

- ids_peak
  - Version 1.11.1.0.5
  - Provides the bindings for IDS peak genericAPI
  - https://pypi.org/project/ids-peak/
- ids_peak_ipl
  - Version 1.16.1.0.4
  - Provides the bindings for IDS peak IPL
  - For using IDS peak IPL, the NumPy library is additionally required.
  - https://pypi.org/project/ids-peak-ipl/
- ids_peak_afl
  - Version 1.7.0.0.2
  - Provides the bindings for IDS peak AFL
  - https://pypi.org/project/ids-peak-afl/

To install the bindings, follow this order:

```
python -m pip install ids_peak_ipl
python -m pip install ids_peak
python -m pip install ids_peak_afl
```

Note: If you use a proxy, you must specify the proxy settings in the commands above directly after "pip install" with
`--proxy <IP address>:<port number>`

First start

To open the IDS peak Cockpit, click on the icon of the IDS peak Cockpit on the desktop or click on "Start > Programs > IDS peak > IDS peak Cockpit".

The camera manager in IDS peak Cockpit shows you the IDS producers available on the host system. The available IDS cameras are listed below their interfaces. If the camera entry is grayed out, this camera is already open.

Opening a camera

To open a camera in IDS peak Cockpit, you can select it via the camera manager or open the first available camera directly with the icon in the main toolbar.

For operating a GigE Vision or USB3 Vision camera, refer also to the Notes for GigE Vision cameras and Notes for USB3 Vision cameras.

Samples

The samples are available for IDS peak comfortSDK, IDS peak genericSDK, or for third parties, e.g. for HALCON.

You can find the binaries of the samples under "Start > Programs > IDS peak > comfortSDK Sample" or "Start > Programs > IDS peak > genericSDK Sample".

You can find the source code of the samples under "Start > Programs > IDS peak > comfortSDK Sample Source Code" or "Start > Programs > IDS peak > genericSDK Sample Source Code".

Note on Qt/QML versions

The samples were built and tested with the following Qt versions:

- Windows (32/64-bit): Qt 5.15.1

For the QML samples, it is recommended to use the latest Qt5 version. Qt6 is currently only supported for the IDS peak genericSDK samples.

Note on C# versions

For the C# samples, the following versions are required

- C# 6.0 or higher
- .NET Framework 4.6.1 or .NET 5 and higher
- For executing the binary samples: .NET 6 Runtime

When building the samples using the dotnet build command: The platform property must be set according to your target platform, e.g. when building on x86_64:

```
dotnet build -p:"Platform=x64" <.csproj file>
```

Note on Python versions

Python samples providing a GUI depend on the following packages:

- pyside
  - Python 3.10 or earlier: pyside2
  - Python 3.11 or later: pyside6
- pyqt5

Samples for IDS peak comfortSDK

IDS peak comfortC

Sample	Content
chunks	Demonstrates how to use chunks which provide additional information about a frame.
configure_camera_gfa	Demonstrates the "generic feature access" which allows direct access to the camera's NodeMap.
firmware_update	Demonstrates how to update cameras using a GUF file and how to be informed about the update progress. To use this example, download a GUF file for your specific camera model. Start the example and pass the path to the GUF file as first argument. Then a list of updateable devices will be displayed. After a confirmation prompt, the compatible devices will be updated. The update progress is displayed in the terminal.
i2c	Demonstrates the use of I2C feature. To use this sample, you need a camera which supports I2C: ● U3 models (requires USB3 Vision firmware 3.20 or higher) ○ PCB version uEye+ LE USB 3.1 Rev. 1.2 ○ uEye+ LE USB 3.1 Rev. 1.2 AF ○ USB 3 uEye+ ACP Rev. 1.2 ● GV models (requires GigE Vision firmware 3.20 or higher) ○ GigE uEye+ ACP Rev. 1.2 ● UI models (requires IDS peak 2.7 or higher) ○ PCB version uEye LE USB 3.1 Gen 1 ○ PCB version USB 3 uEye LE ○ PCB version USB uEye LE ○ GigE uEye LE

inference	Demonstrates the inference feature in IDS peak: - Loading a CNN (convolutional neural network) with IDS peak. - Running the CNN on a captured image. - Readout the results. **Note:** a CNN especially trained for IDS peak is required. Use IDS lighthouse to generate such a CNN or use one of the sample neural networks in the directory `<IDS peak installation>/cnn`.
ipl_features_live_qtwidgets	Shows the use of IDS peak IPL functions for image manipulation in IDS peak comfortSDK. The example uses QtWidgets for this.
message_queue	Demonstrates the use of camera-based and host-based events in a simple GUI. The example uses QtWidgets for this. 1. Click on "Open camera". 2. Select a camera. 3. Active at least one event. 4. Start image acquisition.
reconnect	Opens the first available uEye+ camera and starts image acquisition. If you disconnect the camera, this is detected and signaled. If you reconnect the camera, this will also be detected and image acquisition will start again. This example can also be used in combination with the uEye transport layer for uEye cameras (UI models).
record_video	Gets the camera list and opens a camera after user selection. After the user enters the path and the name of the video file, 100 images are acquired and the video is saved. Afterwards, the camera is closed again.
sequencer_qtwidgets	This example requires a camera that supports the Sequencer feature. The example allows to parameterize 4 sequencer sets and to execute them in trigger mode. The following parameters can be used: - Exposure time - Gains (master gain, red gain, green gain, and blue gain) - Height, width, offsetX, and offsetY The example uses QtWidgets for this.
simple_live_mfc	Opens a camera and displays the live image. The example uses MFC for this.
simple_live_qtwidgets	Opens a camera and displays the live image. The example uses QtWidgets for this.
trigger_live_qtwidgets	Shows the use of triggered acquisition and setting trigger parameters like trigger delay etc. The example uses QtWidgets for this.
walkthrough	Gets the camera list and opens the first available camera. The frame rate is set to

	maximum and 100 images are acquired. Afterwards, the camera is closed again.

Samples for IDS peak genericSDK

IDS peak genericC++

Sample	Content
afl_features_live_qtwidgets	Opens a camera and demonstrates the usage of the IDS peak AFL to control the host-sided autofocus features. This sample uses a QtWidgets GUI, which also provides a method of setting the autofocus ROI by drawing a bouding box on the image.
chunks_live_qml	Opens a camera and shows the use of chunk data via the IDS peak API. The example uses QML (Qt Meta-object Language) for this.
chunks_live_qtwidgets	Opens a camera and shows the use of chunk data via the IDS peak API. The example uses QtWidgets for this.
firmware_update	Demonstrates how to update cameras using a GUF file and how to be informed about the update progress. To use this example, download a GUF file for your specific camera model. Start the example and pass the path to the GUF file as first argument. Then a list of updateable devices will be displayed. After a confirmation prompt, the compatible devices will be updated. The update progress is displayed in the terminal.
device_tree	Create and update module tree and open a camera.
get_first_pixel	Starting image acquisition and accessing image pixel data.
host_auto_features_live_qtwidgets	Opens a camera and shows the use of automatic exposure, gain and white balance algorithms (host based). The example uses QtWidgets for this.
lego_trigger	Using triggered acquisition and setting trigger parameters. **Note:** When exiting the sample, the camera remains in trigger mode. If you want to use the camera in freerun mode afterwards, load the "Default" UserSet or set all TriggerMode entries to "Off".
linescan_live_qtwidgets	Opens a camera and configures it for linescan applications. The example uses QtWidgets for this. **Note:** When exiting the sample, the camera remains in linescan mode. If you want to use the camera in default mode afterwards, load the "Default" UserSet.
multi_camera_live_qtwidgets	Opens several cameras and displays the live image. In addition, different information per camera is displayed, e.g. number of acquired images.
open_camera	Simple creation and opening of a camera object.
open_camera_by_serno	Create and open a camera object by serial number.
open_camera_load_user	Opens a camera. Then, the "Default" user set is loaded and activated.

set_default	
open_camera_select_cti	Using a specific CTI to create and open a camera object.
reconnect_callbacks	Opens the first available uEye+ camera and starts image acquisition. If you disconnect the camera, this is detected and signaled. If you reconnect the camera, this will also be detected and image acquisition will start again. This example can also be used in combination with the uEye transport layer for uEye cameras (UI models).
remote_device_events	Demonstrates the usage of "RemoteDeviceEvents" by activating and displaying "ExposureStart" event during continuous image acquisition. **Note:** This sample requires firmware version 2.8 or higher.
save_images_live_qtwidgets	Simple GUI example using Qt to show images and save them to disk using the IDS peak IPL.
sequencer_live_qml	This example requires a camera that supports the Sequencer feature. The example allows to parameterize 4 sequencer sets and to execute them in trigger mode. The following parameters can be used: ● Exposure time ● Gains (AnalogAll, DigitalAll, DigitalRed, DigitalGreen, DigitalBlue) ● Height, width, offsetX, and offsetY **Note:** When exiting the sample, the camera remains in sequencer mode. If you want to use the camera in default mode afterwards, load the "Default" user set.
simple_live_qml	Simple example for usage with QML GUI to display images.
simple_live_qtwidgets	Simple example for usage with Qt widgets GUI to display images.
unicast	Detecting cameras across subnet boundaries: This example demonstrates how you can identify and operate cameras across subnet boundaries. You will be prompted to select a network interface and enter the unicast IP addresses of the target cameras. The transport layer is configured to use only unicast discovery. Broadcast discovery is explicitly disabled. The identified cameras are listed with detailed information.
walkthrough	Guided tour through module tree and opening of the first available camera.

IDS peak genericC#

Sample	**Content**
firmware_update	Demonstrates how to update cameras using a GUF file and how to be informed about the update progress. To use this example, download a GUF file for your specific camera model. Start the example and pass the path to the GUF file as first argument. Then a list of updateable devices will be displayed. After a confirmation prompt, the compatible devices will be updated. The update progress is displayed in the terminal.

open_camera	Simple creation and opening of a camera object.
reconnect_callbacks	Opens the first available uEye+ camera and starts image acquisition. If you disconnect the camera, this is detected and signaled. If you reconnect the camera, this will also be detected and image acquisition will start again. This example can also be used in combination with the uEye transport layer for uEye cameras (UI models). This sample is provided as Microsoft Visual Studio project file (*.csproj).
simple_live_windows_forms	Simple example for usage with Windows Forms to display images.
simple_live_wpf	Simple example for usage with Windows Presentation Foundation (WPF) to display images.
unicast	Detecting cameras across subnet boundaries: This example demonstrates how you can identify and operate cameras across subnet boundaries. You will be prompted to select a network interface and enter the unicast IP addresses of the target cameras. The transport layer is configured to use only unicast discovery. Broadcast discovery is explicitly disabled. The identified cameras are listed with detailed information.

IDS peak genericPython

Sample	Content
firmware_update	Demonstrates how to update cameras using a GUF file and how to be informed about the update progress. To use this example, download a GUF file for your specific camera model. Start the example and pass the path to the GUF file as first argument. Then a list of updateable devices will be displayed. After a confirmation prompt, the compatible devices will be updated. The update progress is displayed in the terminal.
open_camera	Simple creation and opening of a camera object.
reconnect_callbacks	Opens the first available uEye+ camera and starts image acquisition. If you disconnect the camera, this is detected and signaled. If you reconnect the camera, this will also be detected and image acquisition will start again. This example can also be used in combination with the uEye transport layer for uEye cameras (UI models).
record_video_and_change_parameter	Uses the IDS peak genericAPI to show you how to set camera parameters, start/stop image acquisition and capture a video with IDS peak IPL.
simple_live_qtwidgets	Simple example for usage with Qt widgets GUI to display images.
start_stop_acquisition_software_trigger	Shows how to start/stop image acquisition and how to capture images using a software trigger.

unicast	Detecting cameras across subnet boundaries: This example demonstrates how you can identify and operate cameras across subnet boundaries. You will be prompted to select a network interface and enter the unicast IP addresses of the target cameras. The transport layer is configured to use only unicast discovery. Broadcast discovery is explicitly disabled. The identified cameras are listed with detailed information.

Third party samples

HALCON

Sample	Content
simple_live_halcon	Opening a camera and grabbing images.
triggered_live_halcon	Configuring and using triggered image acquisition.

MIL

Sample	Content
simple_live_mil	Opening a camera and grabbing images.
triggered_live_mil	Configuring and using triggered image acquisition.

Notes for GigE Vision cameras

Setting the IP address for GigE Vision cameras

A valid IP address must be assigned to GigE Vision cameras before they can be used.

1. Open the IDS peak Cockpit.
2. Open the camera manager.
3. Select the camera in the camera manager.
4. Open the "IP configuration" dialog in the camera manager.
5. Assign a valid IP address to the camera.
    1. Select "New network configuration".
    2. Either enter an IP address and subnet mask for the camera or click on the "Find IP address and subnet mask" button.
    3. Confirm your settings with "Apply new settings".
6. The new settings are applied and the camera is rebooted.

Enabling Jumbo Frames for GigE Vision cameras

For optimal performance it is recommended to use Ethernet package sizes which are larger than 1500 bytes. Recommended package size is ~9000 bytes or higher depending on the support of the used network controller. Your whole network infrastructure e.g. switches should support this Ethernet package size if not the GenTL will use the largest possible size.

Maximizing the "receive buffer" size

For operating GigE cameras, it is recommended to set the receive buffer size (so-called receive descriptors) of the network card to its maximum value. Note that not all network cards provide this option.

Disabling interrupt throttling rate (ITR) for 10GigE cameras

Modern network interface drivers try to avoid high CPU load by reducing the number of interrupts they generate. Although this is generally good, the default settings of this option are not optimal for fast 10GigE cameras. In this case, the "interrupt throttling rate" (depending on the card manufacturer also: interrupt moderation rate) should be disabled.

NOTICE! If you disable the interrupt throttling rate, the CPU load increases.

Notes for USB3 Vision cameras

USB3 Vision cameras under USB 2.0

USB3 Vision cameras are limited usable under USB 2.0. Depending on the camera model, not all camera functions are available in USB 2.0 mode. USB3 cameras are optimized for USB 3.0 ports and are not tested by IDS Imaging Development Systems GmbH under USB 2.0.

Note that due to the high performance of modern sensors, some USB3 models are not supported in USB 2.0 mode anymore, as the USB 2.0 interface does not provide enough power.

Integrating cameras into own applications

How to integrate IDS industrial cameras with IDS peak in your own application is described in the further documentation of IDS peak which can be found in:

- IDS peak manual on our website
- IDS peak comfortSDK: "C:\Program Files\IDS\ids_peak\comfort_sdk\api\doc"
- IDS peak genericSDK: "C:\Program Files\IDS\ids_peak\generic_sdk\api\doc"
- IDS peak IPL: "C:\Program Files\IDS\ids_peak\generic_sdk\ipl\doc"
- IDS peak AFL: "C:\Program Files\IDS\ids_peak\generic_sdk\afl\doc"

List of contained files / dependencies

The software includes some parts that are copyright-protected from access by third parties, and which were published under Open Source licensing conditions, see either list at "C:\Program Files\IDS\ids_peak\licenses". The Open Source parts may be used under the terms and conditions of their corresponding Open Source licenses. You will find the Open Source licenses in the "thirdparty_licenses.txt" file.

Uninstallation ⇧

Use the control panel function of Windows to uninstall IDS peak. After uninstalling, the PC must be restarted. It is recommended to remove existing IDS peak installations before installing a new version.

Known issues ⇑

Issue	Details	Solution
During runtime, changes to the network configuration are not detected by the GenTL.	The GenTL cannot handle changes to the system's network configuration during runtime.	If you want to change the network configuration, first close the application and restart the application after you changed the network configuration. This behavior also affects the IDS peak cockpit.
It is not possible to operate a uEye XLE USB 3 camera on a USB 2.0 port.	USB 3 uEye cameras are optimized for use on USB 3.0 ports. In some cases, certain models can be used with restrictions on USB 2.0 ports.	uEye XLE USB 3 camera models does not support USB 2.0 operation.
LabVIEW and template functions	LabVIEW does not support template functions as they are used for `FindNode()` for example (IDS peak genericSDK).	As a workaround, all possible templates of `FindNode`, `FindInvalidatedNode`, `FindInvalidatingNode`, `FindSelectedNode`, and `FindSelectingNode` are provided as separate functions in addition to the regular generic function.
Performance issue with Windows update KB5043145 and WindowsForms with C#	A performance issue was noticed when you use WindowsForms with C# after the Windows update KB5043145 was installed.	We recommend to install the latest Windows update KB5046633 from November 2024 to resolve this performance issue: Windows Update KB5046633.

Contact

IDS Imaging Development Systems GmbH
Dimbacher Str. 10

74182 Obersulm/Germany